Доля П.Г.
Харківський національний університет імені В. Н. Каразіна
факультет математики і інформатики
кафедра теоретичної і прикладної інформатики
2020 р.

Основи моделювання в COMSOL Multiphysics.
Глави 4 – 8.

Алгебраїчні і диференціальні рівняння. Крайові задачі для диференціальних рівнянь. Функції. Параметри і змінні. Комплексні величини.

Зміст

4. Алгебраїчні і диференціальні рівняння. 4
4.1. Використання Global ODE and DAEs інтерфейсу. 5
4.1.1. Просторово незалежні алгебраїчні рівняння і системи. 5
4.1.2. Задача Коші для звичайних диференціальних рівнянь. 22
4.1.3. Диференціально-алгебраїчні системи рівнянь. 54
4.1.4. Використання в рівняннях операторів. 64
4.1.5. Наближення інтегральних і інтегро-диференціальних рівнянь. 74
4.1.6. Приклади допоміжних просторово незалежних рівнянь. 86
4.2. Використання Mathematical Particle Tracing інтерфейсу. 96
4.3. Використання Domain ODEs and DAEs інтерфейсу. 107
4.3.1. Розподілені алгебраїчні задачі. 108
4.3.2. Розподілені диференціальні рівняння. 119
4.4. Демонстраційні приклади. 124
4.4.1. Триланковий механізм по Чебишеву. 124
4.4.2. Рух маятника. 131
4.4.3. Пружина з відривним вантажем. 136
4.4.4. Рух тіла, кинутого під кутом до горизонту. 146
4.4.5. Рух пружного м’яча. 156
4.4.6. Модель Лотки – Вольтерри. 159
4.4.7. Рух планети навкруги сонця. 164
4.4.8. Математичний більярд в еліпсі. 167
4.4.9. Поперечні коливання струни з двома вантажами. 169
4.4.10 Одновимірна модель кристала. 173
Література. 183
5. Крайові задачі для диференціальних рівнянь. 184
5.1. Граничні умови для одновимірних задач. 185
5.1.1. Гранична умова Flux/Source. 188
5.1.2. Гранична умова Constraint. 193
5.1.3. Гранична умова Periodic Condition. 195

5.1.4. Умова Continuity. 197
5.1.5. Внутрішні граничні умови. 203
5.2. Приклади крайових задач на відрізку. 224
5.2.1. Крайова задача $u'' + (1+x^2)\,u = -1,\ u(-1) = 0,\ u(1) = 0$ 224
5.2.2. Крайова задача з кусковим вільним членом. 225
5.2.3. Статичний прогин струни під дією зосереджених сил. 232
5.2.4. Крайова задача $((1+u)\,u')' = 0,\ u(0) = 0,\ u(1) = 1$. 235
5.2.5. Задача на власні значення. 241
5.2.6. Поліноми Лежандра. 245
5.3. Нетипові крайові задачі. 247
5.3.1. Використання Global ODEs and DAEs інтерфейсу. 247
5.3.2. Використання в рівняннях операторів. 254
5.4. Демонстраційні приклади. 260
5.4.1. Теплопровідність в багатошаровій плоскій стінці. 260
5.4.2. Теплопровідність в багатошаровій циліндричній стінці. 264
5.4.3. Задача з розривним розв'язком. 271
5.4.4. Прогин вільно опертої балки. 282
5.4.5. Затиснена на краях балка під дією зосереджених сил. 289
5.4.6. Струна з пружним упором 295
5.4.7. Статичний прогин двох перетинних струн. 303
5.4.8. Стаціонарна хімічна реакція. 308
5.4.9. Електричне поле двошарового сферичного конденсатора. 311
5.4.10. Еластіка Ейлера. 314
5.4.11. Рівняння Шредінгера для гармонічного осцилятора. 318
5.4.12. Дифузія крізь напівнепроникливу перегородку. 323
5.4.13. Ланцюгова лінія. 331
5.4.14. Скін ефект. 339
5.4.15. Осесиметричний нагрів і деформація дроту. 345
Література. 353
6. Функції. 354
6.1. Аналітичні функції. 355
6.2. Кускові функції і приклади їх застосування. 358
6.3. Інтерполяційні функції. 389
6.4. Функції Matlab. 401
6.5. Генерування випадкових даних. 405
6.6. Прогонка по функціям (Function Sweep). 428
6.7. Демонстраційні приклади. 433
6.7.1. Генерування випадкової хвильоподібної поверхні. 433
6.7.2. Сферично-симетричні коливання газу. 436
6.7.3. Поперечні коливання нескінченного пружного стрижня. 440
Література. 444
7. Параметри і змінні. 445
7.1. Загальні відомості про параметри і змінні. 445
7.2. Геометричні і службові змінні. 447

7.3. Зонди, фізичні властивості матеріалів і змінні маси. 475
7.4. Робота з розривними величинами. 493
7.5. Полеві змінні на межі розрахункових областей..... 506
7.6. Змінні в моделях менших розмірностей (Lower Dimensions)..... 517
7.7. Системи координат користувача і їх змінні. 529
Література. 544
8. Комплексні величини..... 544
8.1. Використання комплексних змінних. 545
8.2. Обчислення інтегралів. 556
8.3. Графічні зображення комплексних функцій. 578
8.4. Конформні відображення. 603
8.5. Приклади застосування комплексних потенціалів. 614
8.5.1. Обтікання вертикальної перешкоди над плоским дном. 621
8.5.2. Набігання потоку рідини на параболу..... 624
8.5.3. Течія ідеальної рідини в каналі з перешкодою. 628
8.5.4. Електростатичне поле біля краю конденсатора..... 630
8.5.5. Зовнішнє електростатичне поле двох циліндрів..... 633
8.5.6. Теплове поле між двома ексцентричними трубами..... 637
8.5.7. Теплове поле навколо труби в грунті. 640
8.5.8. Обтікання кола. 644
Література. 648

Ця частина посібника призначена для знайомства з допоміжними інструментами системи COMSOL Multiphysics: змінними, функціями, алгебраїчними і звичайними диференціальними рівняннями, та їх системами. Вони докладно описуються в довідковій документації. Але без прикладів застосування такий опис часто не сприймається. Щоб познайомитися з особливостями цих елементів треба лише навести приклади їх використання. В документації вони є, але часто відносяться до фізичних моделей, які зрозумілі не всім користувачам. В нашому посібнику ми спробували навести достатньо прості, зрозумілі більшості читачів, приклади.

Подання матеріалу по главам зроблено відносно самостійним, тому можете не хвилюватися через зайву залежність розділів один від одного. Але знайомство з першою главою посібника потрібно. Особливо це стосується домовленостей скороченого позначення інструкцій, які наведено в п. 1.4 [http://geometry.karazin.ua/~Dolya/documents: Доля П. Г. Основи моделювання в COMSOL Multiphysics. Глави 1 – 3. – Х.: ХНУ ім. В.Н. Каразіна, 2019. – 529 с.]

4. Алгебраїчні і диференціальні рівняння.

При побудові моделей в COMSOL Multiphysics часто застосовуються змінні. Коли вони задаються явно, то для них в ДМ використовуються вузли **Variables**. Однак інколи змінні визначаються зі співвідношень, які можуть бути представлені тільки алгебраїчними або диференціальними рівняннями. В таких випадках замість змінних ми отримуємо невідомі, які треба розшукувати з цих додаткових рівнянь, включаючи їх до «основної» моделі. Звісно, що алгебраїчні і звичайні диференціальні рівняння можуть розв'язуватися і як самостійні задачі.

Метою цієї глави є опис інструментів COMSOL Multiphysics, використовуваних при розв'язанні алгебраїчних (АР) та звичайних диференціальних рівнянь (ЗДР), і їх систем, незалежно від того, є вони «основними» чи додатковими рівняннями моделі.

Спочатку буде розглянуто методику розв'язання алгебраїчних систем та задач Коші для звичайних диференціальних рівнянь, які не містять просторових координат. Потім буде розглянуто розподілені (distributed, просторово залежні) алгебраїчні та звичайні диференціальні рівняння, які описують неперервні системи без просторової взаємодії. В такі рівняння не входить диференціювання по просторовим координатам, але від координат можуть залежати коефіцієнти рівнянь або початкові умови.

В термінології COMSOL Multiphysics просторово незалежні задачі відносяться до типу нульвимірних, бо їх можна інтерпретувати як обчислювання значень фізичних величин в точках. Навпаки, для розв'язання розподілених рівнянь потрібно будувати просторові моделі.

Для розв'язання просторово незалежних і розподілених рівнянь використовуються різні інтерфейси:

- Global ODEs and DAEs інтерфейс використовується при розв'язанні АР та задачі Коші для просторово незалежних ЗДР;
- Domain ODEs and DAEs інтерфейс використовується при розв'язанні розподілених рівнянь;

Крайові задачі для ЗДР, які моделюють, наприклад, фізичні процеси в одновимірних середовищах, будуть розглянуті в наступній главі. Основою їх розв'язання є математичні PDE інтерфейси, використовувані в стаціонарних одновимірних моделях.

Отже коли ви маєте справу зі звичайними диференціальними рівняннями, то спосіб їх розв'язання в COMSOL Multiphysics залежить від додаткових умов.

	Задача Коші, нема залежності від просторових координат.	Задача Коші, нема диференціювання по просторовим координатам	Крайова задача, нема залежності від часу.
Приклад рівняння	$u''_{tt} + \alpha u'_t + \beta u = f(t)$	$u''_{tt} + a(x,y,z)\,u'_t +$ $b(x,y,z)\,u = f(t,x,y,z)$	$u''_{xx} + a(x)\,u'_x +$ $b(x)\,u(x) = g(x)$
Приклад додаткових умов	$u(0) = u_0,\ u'(0) = u_1$	$u(0) = u_0(x,y,z)$ $u'(0) = u_1(x,y,z)$	$u(0) = u_0,\ u(L) = u_1$
Інтерфейс	Global ODEs and DAEs інтерфейс	Domain ODEs and DAEs інтерфейс	Математичні PDE інтерфейси

Окрім `Global ODEs and DAEs` і `Domain ODEs and DAEs` інтерфейсів, використовуваних при розв'язанні задач Коші для ЗДР, в COMSOL Multiphysics існують інші спеціалізовані інтерфейси, в яких зручно розв'язувати окремі типи систем ЗДР. Одним з них є `Mathematical Particle Tracing` інтерфейс. Приклади його застосування будуть тут наведені.

Більшість розглянутих в цій главі задач може бути з успіхом розв'язана в інших програмних середовищах. Однак, щоб звикнути до інструментів системи потрібно навчитися досліджувати різноманітні моделі.

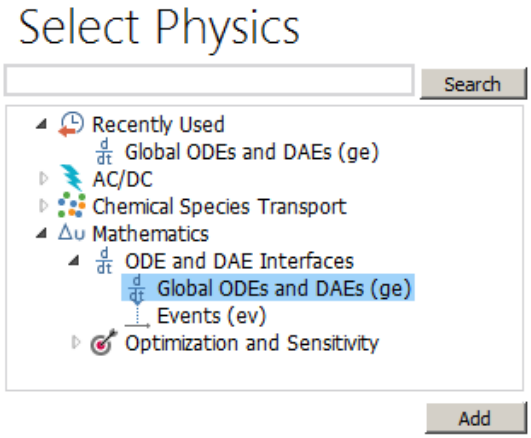
4.1. Використання Global ODE and DAEs інтерфейсу.

В цьому розділі ми покажемо, як можна розв'язувати просторово незалежні системи алгебраїчних і звичайних диференціальних рівнянь. Слід розуміти, що система COMSOL Multiphysics здатна поєднувати задачі різної фізичної природи і різної розмірності, при чому основні досліджувані моделі зводяться до диференціальних рівнянь в частинних похідних (ДРЧП). Задачі, що описуються системами алгебраїчних або звичайних диференціальних рівнянь (ЗДР), є допоміжними. Зокрема, такі задачі можуть виникнути в просторових моделях, в яких зовнішній вплив здійснюється в точках і визначається, наприклад, із ЗДР. Самостійні математичні моделі, які зводяться до систем алгебраїчних або диференціальних рівнянь, краще розв'язувати в інших пакетах, наприклад, в MatLab, з яким у COMSOL Multiphysics є тісна інтеграція. Однак, для повноти картини і для тренування користувача на відносно простих прикладах, ми тут наводимо опис шляхів розв'язання таких «нульвимірних» задач.

4.1.1. Просторово незалежні алгебраїчні рівняння і системи.

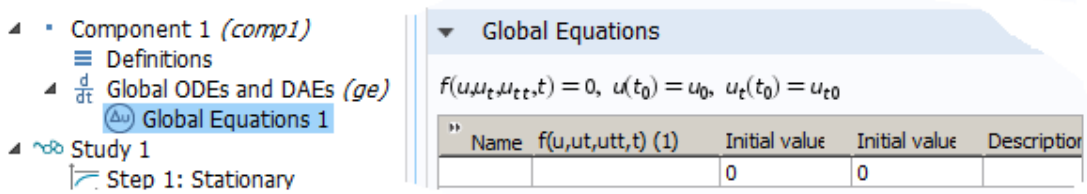
Розв'язання алгебраїчного рівняння (або системи АР) починається зі створення ДМ задачі. Виберіть пункт меню File –> New, а потім варіант створення моделі за допомогою майстра (Model Wizard). Відкривається вікно, в якому обирається просторова модель задачі.

Варіант 0D означає, що задача буде містити значення фізичних величин в точках. Вони можуть бути незмінними або функціями часу, в залежності від того, яке дослідження (study) ми виберемо (стаціонарне або динамічне). Тиснемо мишею на кнопку 0D. Відкривається вікно вибору моделі (Select Physics). Серед можливих варіантів, які надає дерево інтерфейсів, розкриваємо гілку Mathematics. Потім розкриваємо підгілку ODE and DAE Interfaces і вибираємо вузол Global ODEs and DAEs (ge). Тут ODEs означає Ordinary Differential Equations (звичайні диференціальні рівняння) і DAEs - Differential Algebraic Equations (диференціально алгебраїчні рівняння). Інтерфейс Global ODEs and DAEs використовується для створення просторово незалежних моделей з системами алгебраїчних або ЗД рівнянь.



Клацніть мишею по кнопці Add. В поле Added physics interfaces, яке розташовано нижче, додається назва моделі. Натисніть мишею кнопку Study. Відкривається вікно Select Study. Виберіть варіант Stationary, який означає, що шукані величини не залежать від часу (будемо розв'язувати алгебраїчні рівняння). Завершуємо все натисканням на кнопку Done, розташовану внизу панелі Select Study. Створюється вікно задачі з панеллю Model Builder в його лівій частині. В дереві моделі (в панелі Model Builder) з'явиться гілка **Component 1**, підгілка **Global ODEs and DAEs (ge)** і вузол **Global Equations 1**. Також з'являється гілка **Study 1** з вузлом **Solver Configurations**. На стрічці з'являється закладка Global Equations, яку в попередніх прикладах ми не бачили.

В ПВ вузла **Global Equations 1** розташована таблиця, яка використовується для введення рівнянь. Ми будемо називати її таблицею рівнянь (ТР).



Приклад 1. Розв'яжемо алгебраїчне рівняння $f(u)=0$, де $f(u)=u^3+3u-4$. Ми знаємо, що воно має єдиний дійсний корінь $u=1$.

Створіть модель задачі послідовністю наведених вище інструкцій, які запишемо за допомогою стислих позначень.

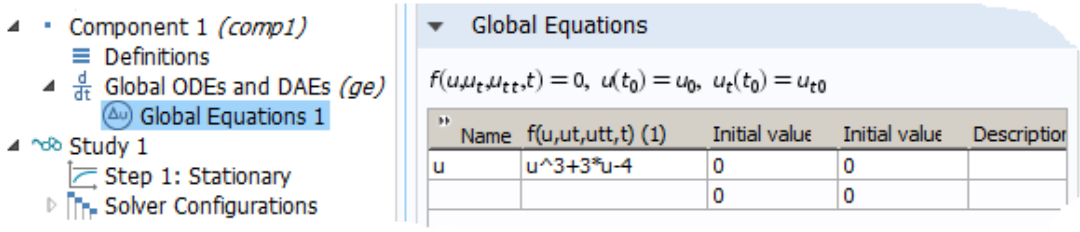
File –> New –> Model Wizard –> 0D –> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs(ge) –> Stationary

Зверніть увагу, що у вікні `Select Physics` ім'я шуканої змінної не вказується.

В ПВ вузла **Global Equations 1** в таблиці рівнянь в першому рядку в першій колонці (з назвою **`Name`**) введіть ім'я невідомої `u`, а в другій колонці (вона має назву **`f(u,ut,utt,t)`**) введіть вираз `u^3+3*u-4`. Третя колонка призначена для введення початкового наближення. В разі ЗДР третя і четверта колонки використовуються для введення початкових значень. Залишимо їх без змін. Ці кроки в подальшому будемо записувати стисло наступним чином:

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name** = `u`
– **f(u, ut, utt, t)** = `u^3+3*u-4`



Для утворення рівняння вираз `u^3+3*u-4` прирівнюється нулю.

Тепер знаходимо корінь.

`Home` –> Compute

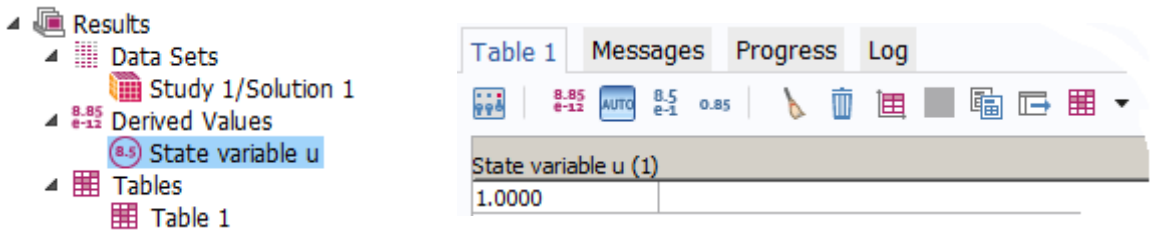
Здійснюються розрахунки і в дереві моделі з'являється вузол **Results - Derived Values - State variable u**. Виконайте обчислення цього вузла.

Results - Derived Values - State variable u:

`Expression` – **Expression** = `comp1.u` (*вираз за замовчуванням*)

LMB > Evaluate

В гілці **Results - Tables** з'являється вузол **Table 1**, який представляє таблицю результатів. В правій нижній частині вікна програми з'являється панель з вмістом цієї таблиці. В нашому випадку таблиця буде містити корінь рівняння `1.0`.



Приклад 2. Розв'язати трансцендентне рівняння $f(u)=0$, де $f(u)=e^{-u}-u$.

Створіть модель задачі так, як описано раніше.

File –> New –> Model Wizard –> 0D –> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs(ge) –> Stationary

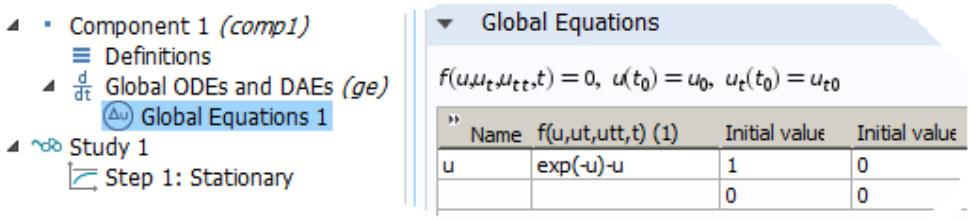
Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – Name = u

– f(u, ut, utt, t) = exp(-u)-u

– Initial value = 1



Тут в третій колонці ми ввели початкове наближення кореня.

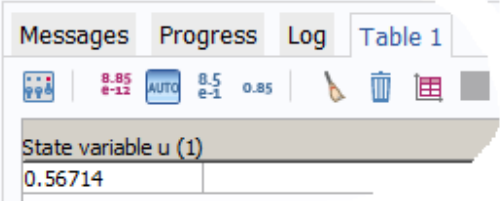
Виконайте обчислення.

Home -> Compute

Здійснюються обчислення, і в дереві моделі з'являється вузол **Results - Derived Values - State variable u**. Виконайте обчислення цього вузла.

Results - Derived Values - State variable u: LMB > Evaluate

В правій нижній частині вікна програми з'являється таблиця зі значенням кореня, який дорівнює 0.56714.



Збережіть модель в файл, але не закривайте його.

Приклад 3 . Розв'язати рівняння $f(u)=0$, де $f(u)=u^3-6u^2+11u-6$. Відомо, що воно має три дійсних кореня: $u=1, u=2$ і $u=3$. Відшукаємо їх.

Будемо працювати в файлі, створеному в попередньому прикладі. В ДМ можна мати кілька вузлів типу Global Equations, в кожному з яких розв'язувати власну задачу. Для додавання нового вузла Global Equations виконайте інструкцію:

Global Equations -> **Global** – Global Equations

В дереві моделі в гілці **Component 1 - Global ODEs and DAEs (ge)** з'явиться новий вузол **Global Equations 2**. В його ПВ в ТР введіть:

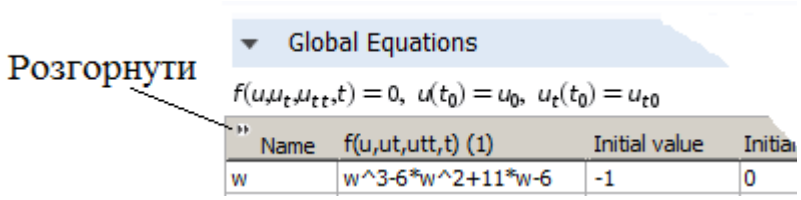
Global ODEs and DAEs(ge) - Global Equations 2:

Global Equations – Name = w

– f(u, ut, utt, t) = w^3-6*w^2+11*w-6

– Initial value = -1

Майте на увазі, що в межах спільного компонента змінні повинні мати різні імена. Тому тут нову шукану змінну ми позначили ідентифікатором w. Щоб побачити все рівняння, якщо воно не вміщується в полі f(u,ut,utt,t), натисніть кнопку «Розгорнути», розташовану в лівому верхньому куті таблиці рівнянь.



Виконайте обчислення.

`Home` **->** Compute

Помітних змін в ДМ не відбувається, але значення невідомої `w` буде збережено в робочому просторі системи. Надрукуйте його.

Results - Derived Values - State variable u: RMB > Duplicate

Label = `State variable w`

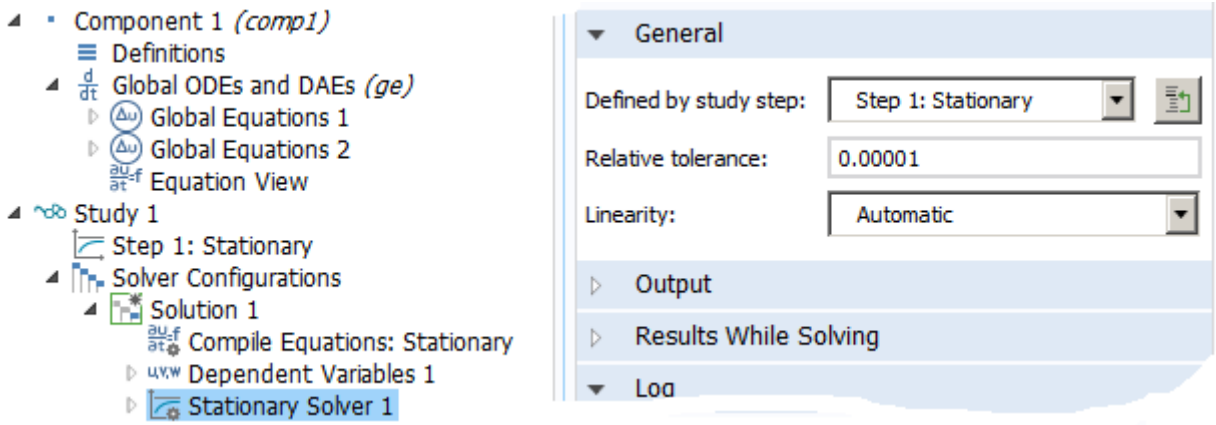
`Expression` – **Expression** = `comp1.w`

LMB > Evaluate – New Table

З'являється нова таблиця результатів зі значенням кореня `0.99987`. Корінь обчислено з похибкою. Щоб підвищити точність обчислень, зайдіть в ПВ вузла **Study 1 - Solver Configurations - Solution 1 - Stationary Solver 1** і в полі **Relative tolerance** (відносна похибка) введіть значення `0.00001`.

Study 1 - Solver Configurations - Solution 1- Stationary Solver 1:

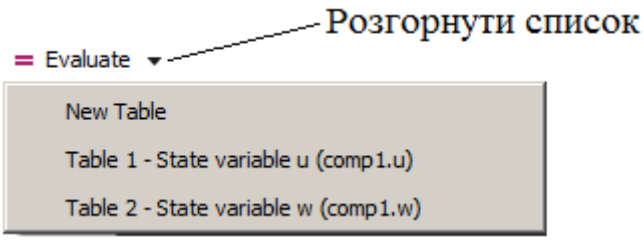
`General` – **Relative tolerance** = `0.00001`



Знову обчисліть модель.

`Home` **->** Compute

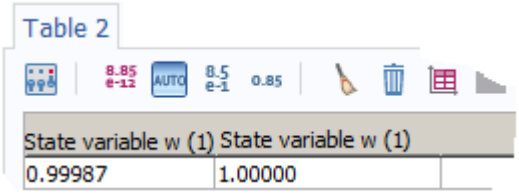
Надрукуйте нове значення шуканої змінної `w` в існуючу таблицю `Table 2`. В ПВ вузла **State variable w** вгорі розкрийте список біля кнопки = Evaluate .



В списку виберіть команду Table 2 - State variable w. Це означає, що значення невідомої буде виводитися в існуючу таблицю, в якій вже знаходиться попереднє значення `w`.

Results - Derived Values - State variable w : LMB > Evaluate –Table 2

В другій колонці таблиці з'явиться нове значення кореня `1.00000`, обчислене з новими налаштуваннями Солвера (новою відносною похибкою обчислень).



Щоб дізнатися, чи є у рівняння інші корені, побудуємо графік функції $f(x)=x^3-6x^2+11x-6$. Для цього створіть цю функцію в ДМ.

Global Definitions : RMB > Functions – Analytic

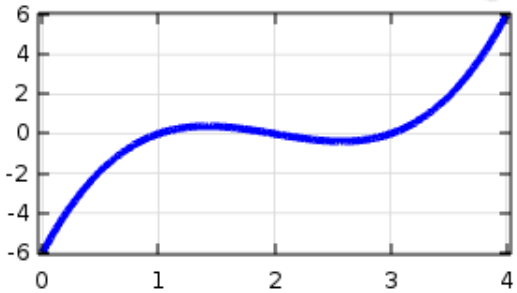
Function name = `f`

`Definition` – **Expression** = `x^3-6*x^2+11*x-6`

– **Arguments** = `x`

`Plot Parameters` – **x, Lower limit, Upper limit** = `x, 0, 4` (*три поля*)

LMB > Create Plot (*побудувати графік в окремому вікні*)

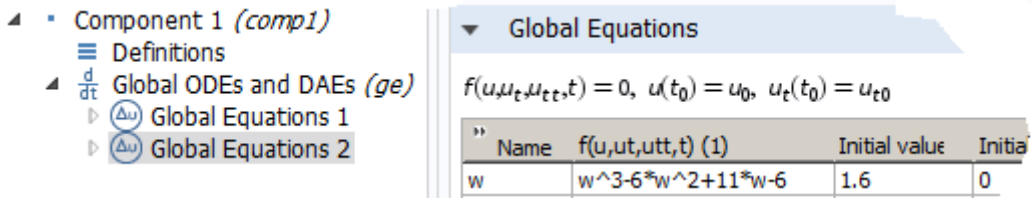


Команда Create Plot створює одновимірне джерело даних **Grid 1D** і одновимірну графічну групу **1D Plot Group 1** з вузлом **Line Graph 1** лінійного графіка $f(x)$. У цьому вузлі ви можете налаштувати стиль кривої (заради цього натискалася кнопка Create Plot). З графіка зрозуміло, що рівняння має ще два кореня. Щоб знайти їх, потрібно в ТР змінити початкове наближення.

Повертаємося в ПВ вузла **Global Equations 2** і в ТР міняємо значення початкового наближення на `1.6`.

Global ODEs and DAEs(ge) - Global Equations 2:

`Global Equations` – **Initial value** = `1.6`



Знову обчислюємо модель і друкуємо значення шуканої змінної `w` в таблицю `Table 2`.

`Home` -> Compute

Derived Values - State variable w: LMB > Evaluate –Table 2

В третій колонці таблиці `Table 2` з'являється значення кореня `2.0000`, прилеглого до початкового наближення `1.6`. Ще раз повертаємося в ПВ вузла **Global Equations 2** і в ТР міняємо значення початкового наближення на `5`.

Global ODEs and DAEs(ge) - Global Equations 2:

`Global Equations` – **Initial value** = `5`

Обчислюємо модель і друкуємо нове значення змінної `w` в таблицю `Table 2`.

`Home` -> Compute

Derived Values - State variable w: LMB > Evaluate –Table 2

В таблиці результатів з'являється значення ще одного кореня `3.0000`.

Table 2

State variable w (1)	State variable w (1)	State variable w (1)	State variable w (1)
0.99987	1.00000	2.0000	3.0000

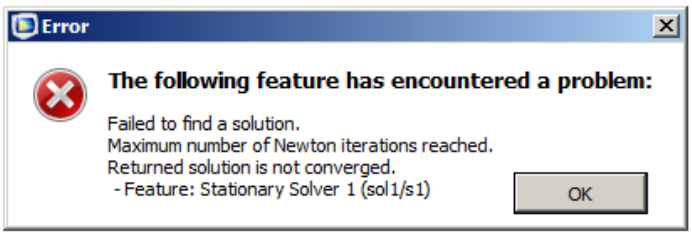
Зауваження 1. Можна не створювати два різні вузли `Global Equations`, а в ТР вузла **Global Equations 1** ввести друге рівняння.

Component 1 *(comp1)*
Definitions
Global ODEs and DAEs *(ge)*
Global Equations 1

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t) (1)	Initial value	Initial value
u	exp(-u)-u	1	0
w	w^3-6*w^2+11*w-6	-1	0

Зауваження 2. Якщо розв'язку рівняння немає (наприклад $u^3-e^{u+2}=0$), то ви отримаєте повідомлення про помилку приблизно такого змісту



Вправа. Розв'яжіть рівняння $\sin x - x^2\cos x = 0$. Знайдіть всі корені, розташовані на відрізку `[-5,5]`.

Відповідь: `0.0000, -1.8539,-4.7566,4.6665`.

■

В Comsol Multiphysics можна чисельно розв'язувати задачі з параметрами. Для цього в дереві моделі створюється вузол **Parameters**. Безпосередньо для параметричного аналізу моделі в гілці **Study** створюється підвузол **Parametric Sweep**, або у вузли **Study 1 – Step 1: Stationary** в розділі **`Study Extensions`** встановлюється перемикач `Auxiliary Sweep` (допоміжний аналіз).

Приклад 4. Розв'язати алгебраїчне рівняння з параметром

$$x^2-2x-a^4=0$$

Створіть модель задачі з ODEs and DAEs інтерфейсом.

File –> New –> Model Wizard –> 0D –> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) –> Stationary

В гілці **Global Definitions** створіть вузол **Parameters**. В його ПВ в таблиці параметрів в колонці **`Name`** введіть ім'я параметра `a`, а в колонці **`Expression`** введіть яке-небудь значення, наприклад, одиницю.

Global Definitions: RMB > Parameters

`Parameters` – Name, Expression = `a, 1`

Global Definitions
Parameters
Materials

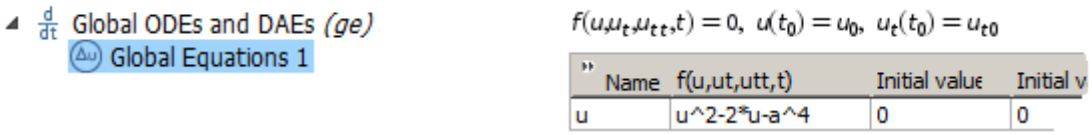
Parameters

Name	Expression	Value
a	1	1

В ПВ вузла **Global Equations 1** заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name** = u
– **f(u, ut, utt, t)** = u^2-2*u-a^4
– **Initial value** = 0



Для визначення залежності розв'язку рівняння від параметра використаємо Auxiliary Sweep (допоміжний) аналіз, який вмикається встановленням відповідного прапорця в ПВ вузла **Study – Step 1: Stationary** (нагадаємо що, коли параметр не змінює геометрію моделі, то доцільніше використовувати Auxiliary Sweep параметричний аналіз).

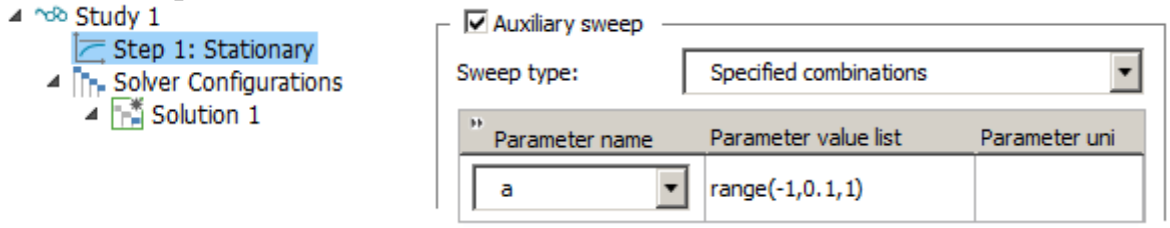
Study 1 – Step 1: Stationary:

Study Extensions – **Auxiliary sweep** = True

В ПВ вузла **Step 1: Stationary** в розділі **Study Extensions** активується таблиця вибору параметрів. Клацніть по таблиці правою кнопкою миші і з випадного меню виберіть команду Add. Зі списку, який з'явиться в стовпці **Parameter name**, виберіть параметр a. У другому стовпці таблиці задайте діапазон зміни цього параметра.

Study Extensions – **Parameter name** = a
– **Parameter value list** = range(-1,0.1,1)

Функція range(-1,0.1,1) представляє значення параметра a, які змінюються від -1 до 1 з кроком 0.1.

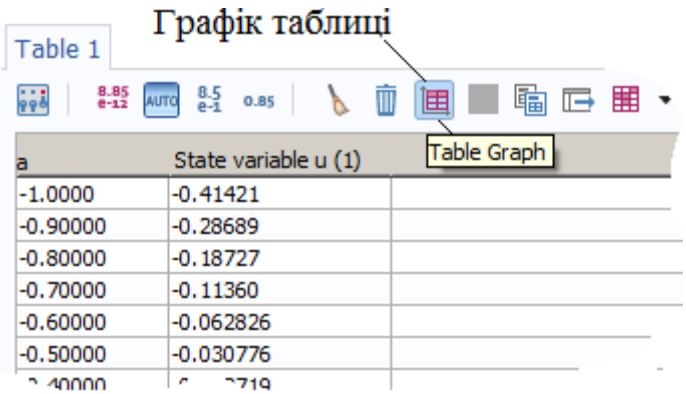


Після цього виконайте обчислення моделі.

Study-> Compute

В гілці **Results – Derived Values** з'являється вузол **State variable u**. Натискання в його ПВ на кнопку Evaluate призводить до побудови таблиці розв'язків, які відповідають різним значенням параметра a з заданого діапазону range(-1,0.1,1).

Results – Derived Values – State variable u: LMB > Evaluate



Побудуйте графік залежності кореня рівняння від параметра a. Для цього достатньо клацнути мишею по кнопці Table Graph, яка розташована над таблицею.

Table 1 : LMB > Table Graph

Буде створена графічна група **1D PlotGroup 1** з вузлом **Table Graph 1**. Скорегуйте в ньому стиль кривої і легенду.

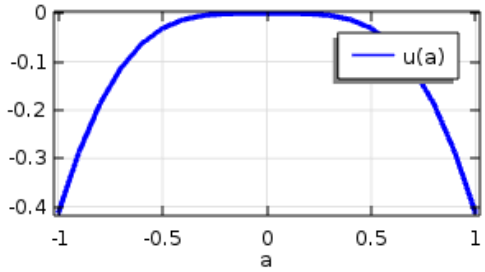
1D Plot Group 1 - Table Graph 1:

Coloring and Style – **Width** = 3

Legends – **Show legends, Legends, Legends** = True, Manual, 'u(a)'

LMB > Plot

В результаті отримаєте графік функції $u(a)$ – залежності розв'язку u алгебраїчного рівняння $u^2 - 2u - a^4 = 0$ від параметра a.



Приклад 5. Розв'язати систему лінійних рівнянь

$$\begin{cases} 2u + 3v - 13 = 0 \\ 3u - 2v = 0 \end{cases}$$

Створіть модель задачі з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

В ПВ вузла **Global Equations 1** в таблиці рівнянь в першому рядку в стовпці **Name** введіть ім'я першої невідомої u, а в другому рядку цього стовпця введіть ім'я другої невідомої v. Потім в другому стовпці ТР в першому рядку введіть вираз 2*u+3*v-13, а в другому рядку введіть вираз 3*u-2*v.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name, f(u, ut, utt, t)** = u, 2*u+3*v-13

– **Name, f(u, u, utt, t)** = v, 3*u-2*v

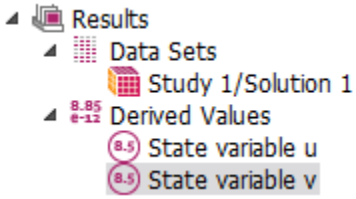
Name	f(u,ut,utt,t)	Initial value	Initial value
u	2*u+3*v-13	0	0
v	3*u-2*v	0	0

Оскільки розв'язок обраної лінійної системи один, то значення початкових наближень для невідомих несуттєві, і ми залишаємо їх рівними 0.

Виконайте обчислення.

Home -> Compute

В гілці **Results - Derived Values** з'являються два вузли **State variable u** і **State variable v**.



Надрукуйте значення величин `u`, `v` в спільну таблицю.

Derived Values – State variable u: LMB > Evaluate

Derived Values – State variable v: LMB > Evaluate - Table 1

State variable u (1)	State variable v (1)
2.0000	3.0000

Аналогічно розв'язуються лінійні системи з більшою кількістю невідомих.

Вправа. Розв'яжіть систему рівнянь

$$\begin{cases} x_1 + x_2 + x_3 = 6 \\ 2x_1 - x_2 - x_3 = -3 \\ 3x_1 - 2x_2 + x_3 = 2 \end{cases}$$

Відповідь: `1.0,2.0,3.0`.

Приклад 6. Розв'яжіть систему лінійних рівнянь

$$\begin{cases} x + y + z - 3 = 0 \\ 2x + 3y + 4z - 9 = 0 \end{cases}$$

Система має нескінченну кількість розв'язків. Знайдемо деякі з них.

Перший спосіб. Створіть модель задачі з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

У нашої конкретної системі одну змінну, наприклад третю, можна прирівняти довільній сталій, тобто додати третє рівняння у вигляді $z - a = 0$, де a – параметр моделі. Змінюючи його, можна отримати різні розв'язки.

Створіть вузол **Parameters** з одним параметром `a=1`.

Global Definitions: RMB > Parameters

`Parameters` – Name, Expression = `a, 1`

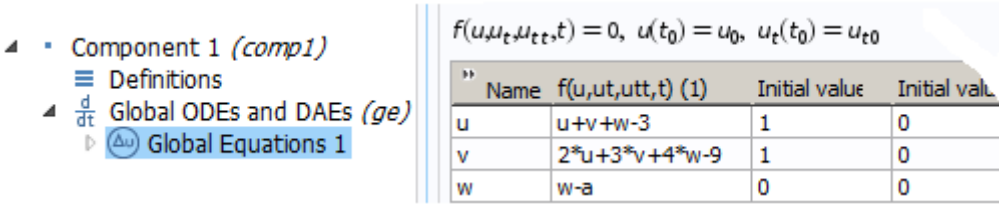
В ПВ вузла **Global Equations 1** заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – Name, f(u, ut, utt, t), Initial value = `u, u+v+w-3,1`

– Name, f(u, ut, utt, t), Initial value =`v, 2*u+3*v+4*w-9,1`

– Name, f(u, ut, utt, t) = `w, w-a`



Зауваження. Не називайте шукані невідомі іменами просторових координат *x*, *y*, *z*. В поточній задачі це не суттєво. Але не варто заводити хибні звички.

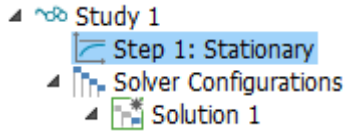
Для знаходження різних розв'язків системи використаємо допоміжний параметричний аналіз `Auxiliary Sweep`.

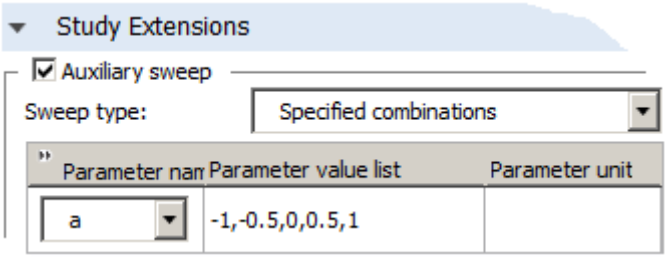
Study 1 – Step 1: Stationary:

`Study Extensions(таблиця)` – **Auxiliary sweep** = `True`

– **Parameter name** = `a`

– **Parameter value list** = `-1,-0.5,0,0.5,1`

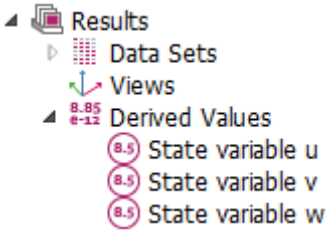




Виконайте обчислення.

`Home` -> Compute

В гілці **Results - Derived Values** з'являться три вузли **State variable u, State variable v** і **State variable w**.



Надрукуйте значення величин `u`, `v`,`w` в спільну таблицю.

Derived Values – **State variable u**: LMB > Evaluate

– **State variable v**: LMB > Evaluate - Table 1

– **State variable w**: LMB > Evaluate - Table 1

a	State variable u (1)	State variable v (1)	State variable w (1)
-1.0000	-1.00000	5.0000	-1.0000
-0.50000	-0.50000	4.0000	-0.50000
0.0000	6.6613E-16	3.0000	0.0000
0.50000	0.50000	2.0000	0.50000
1.0000	1.0000	1.00000	1.0000

Перший стовпець містить значення параметра `a`, а інші – трійки розв'язків, які відповідають цьому параметру. Отже розв'язками є трійки чисел: `-1,5,-1`; `-0.5,4,-0.5` і т.д.

Зауваження. Можна побудувати графіки залежності розв'язків $u(a), v(a), w(a)$ від параметра `a` таким же чином, як в прикладі 4.

Другий спосіб. Створіть новий файл (нову модель) з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -

ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

Оскільки в системі три невідомих, то в ТР повинно бути заповнено три рядки. В третьому рядку можна повторно записати одне з перших двох рівнянь, наприклад, перше. Окрім цього треба задати якісь початкові наближення.

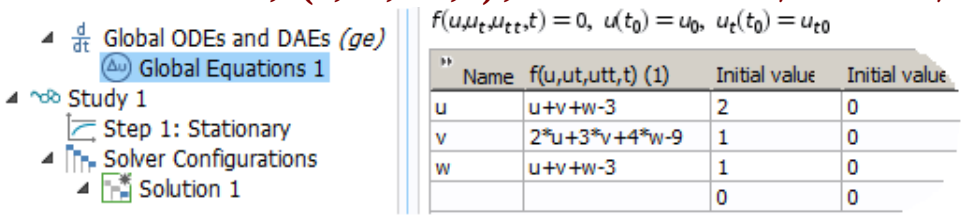
Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations –

– Name, f(u, ut, utt, t), Initial value = `u,u+v+w-3,2`

– Name, f(u, ut, utt, t), Initial value = `v, 2*u+3*v+4*w-9,1`

– Name, f(u, ut, utt, t) , Initial value = `w,u+v+w-3,1`



Якщо зараз виконати команду Compute, то ви отримаєте повідомлення про помилку (перевірте це). Щоб розв'язати систему, змініть солвер (метод розв'язання) в ПВ вузла **Study 1-Solution 1-Stationary Solver 1-Direct**. В розділі **General** цієї ПВ в випадному списку **Solver** виберіть PARDISO, і виконайте команду Compute.

Study 1-Solution 1-Stationary Solver 1-Direct:

General – **Solver** = PARDISO

Home -> Compute

Надрукуйте значення обчислених величин u, v,w в спільну таблицю.

Derived Values – State variable u: LMB > Evaluate

– State variable v: LMB > Evaluate - Table 1

– State variable w: LMB > Evaluate - Table 1

State variable u (1)	State variable v (1)	State variable w (1)
1.0000	1.0000	1.0000

Щоб знайти інший розв'язок змініть початкові умови в таблиці рівнянь. Наприклад, поміняйте початкові значення на (-1,-2,-1) і переобчисліть модель. Ви повинні отримати розв'язок 2.5000, -2.0000,2.5000.

Derived Values – State variable u: LMB > Evaluate – New Table

– State variable v: LMB > Evaluate – Table 2

– State variable w: LMB > Evaluate – Table 2

State variable u (1)	State variable v (1)	State variable w (1)
2.5000	-2.0000	2.5000

Приклад 7. Розв'яжіть систему нелінійних рівнянь

$$\begin{cases} 4u^2 - v^2 + 9 = 0 \\ -u^2 + v^2 - u \cdot v + u - v - 8 = 0 \end{cases}$$

Розв'язання. Створіть модель задачі з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

В ПВ вузла **Global Equations 1** заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations –

– Name, f(u, ut, utt, t), Initial value = `u, 4*u^2-v^2+9,1`

– Name, f(u, ut, utt, t), Initial value = `v,-u^2+v^2-u*v+u-v-8,3`

Name	f(u,ut,utt,t) (1)	Initial value (u_0) (1)	Initial value (u_t
u	4*u^2-v^2+9	1	0
v	-u^2+v^2-u*v+u-v-8	3	0

В якості початкових наближень виберіть, наприклад, 1 і 3, і виконайте обчислення.

`Home` **–>** Compute

В гілці **Results - Derived Values** з'являються два вузли **State variable u** і **State variable v**. Надрукуйте значення знайдених величин `u`, `v` в спільну таблицю.

Derived Values – State variable u: LMB > Evaluate

Derived Values – State variable v: LMB > Evaluate - Table 1

State variable u (1)	State variable v (1)
2.0000	5.0000

Спробуйте використати інші початкові наближення. Деякі з них, наприклад нульові, не призводять до розв'язку.

Вправа. Знайдіть відмінний від тривіального розв'язок системи рівнянь

$$\begin{cases} 4x^2 - y^2 = 0 \\ -x^2 + y^2 - x - y = 0 \end{cases}$$

Відповідь: `(1,2)`.

Вправа. Знайдіть розв'язок системи рівнянь

$$\begin{cases} x(2-y) - e^y \cdot \cos x = 0 \\ x - y - \cos x - e^y = -2 \end{cases}$$

Відповідь: `(0.73909,0.44285)`.

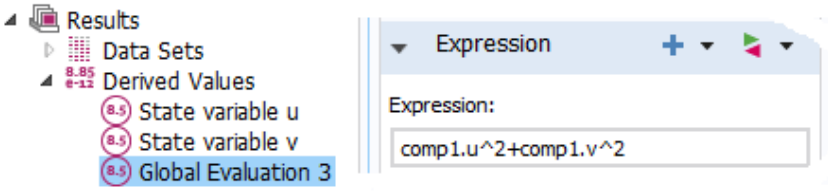
Зауваження 1. В гілці **Results - Derived Values** можна створити вузол типу `Global Evaluation`, за допомогою якого обчислити будь-який вираз, залежний від шуканих змінних. Наприклад, в моделі останнього файлу клацніть правою кнопкою миші по вузлу **Derived Values** і у випадному списку виберіть команду Global Evaluation. Буде створено вузол **Global Evaluation 3**. В його ПВ в полі **Expression** введіть вираз `comp1.u^2+comp1.v^2` і обчисліть його.

Results-Derived Values: RMB > Global Evaluation

Global Evaluation 3:

`Expression` – **Expression** = `comp1.u^2+comp1.v^2`

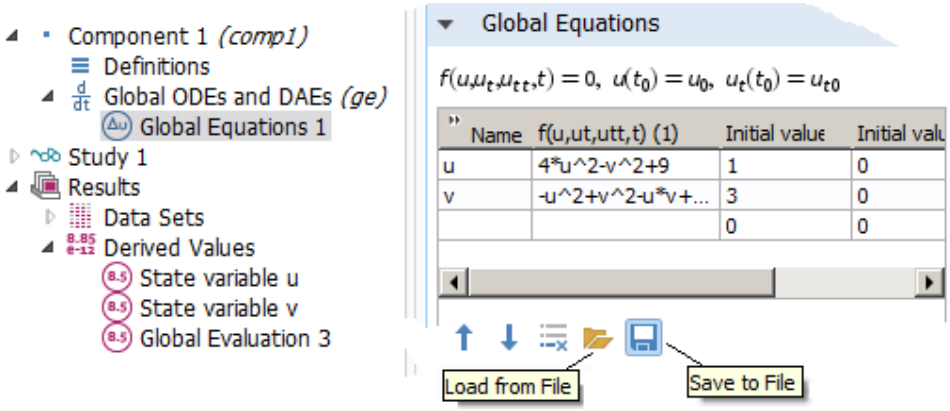
LMB > Evaluate – New Table



В таблиці результатів відобразиться число `29`.

Зверніть увагу, що в ПВ вузлів **State variable u** і **State variable v** використовуються повні імена змінних `comp1.u` і `comp1.v`. Вони містять префікс `comp1`, який є ім'ям вузла **Component 1**.

Зауваження 2. Вміст таблиці рівнянь можна зберігати в текстовий файл. Для цього в ПВ вузла **Global Equations 1** натисніть на кнопку Save to File, дайте ім'я текстового файлу і збережіть його.



Збережені рівняння можна використати в інший моделі, якщо завантажити їх за допомогою кнопки Load from File. ■

В моделях з `Global ODEs and DAEs` інтерфейсом в таблиці рівнянь в стовпці з заголовком **`f(u,ut,utt,t)`** можна вводити не тільки імена шуканих змінних, таких як `u`, але також їх перші (`ut`) і другі (`utt`) похідні. Ми це будемо робити в наступних параграфах. Але з заголовка видно, що можна використовувати параметр часу `t` в коефіцієнтах і вільних членах рівнянь. Це означає, що можна розв'язувати алгебраїчні рівняння з параметром `t`.

Приклад 8. Розв'яжіть систему рівнянь з параметром.

$$\begin{cases} u^2 - u(vt + \sin t) + vt\sin t = 0 \\ v^2 - v(t-u) - ut = 0 \end{cases}$$

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Від статичного інтерфейсу відмінність полягає в тому, що на останньому етапі роботи майстра в панелі `Select Study` вибирається динамічна модель `Time Dependent`, а не стаціонарна (`Stationary`).

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

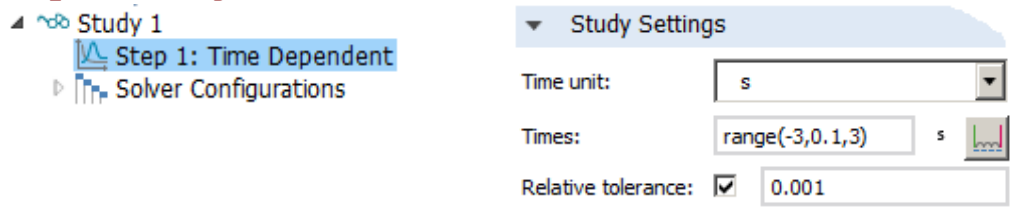
Задайте діапазон зміни параметра `t`.

Step 1: Time Dependent:

`Study Settings` – Times = `range(-3,0.1,3)`

Вираз `range(-3,0.1,3)` означає, що параметр `t` буде приймати значення від `-3` до `3` з кроком `0.1`. В цій задачі потрібно також відразу зменшити відносну похибку обчислень в полі **Relative tolerance.**

`Study Settings` – Relative tolerance = `True, 0.001`



В ПВ вузла **Global Equations 1** заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – Name, f(u, ut, utt, t), Initial value =

`u, u^2-u*(v*t+sin(t))+v*t*sin(t),0`

– Name, f(u, ut, utt, t), Initial value = `v, v^2-v*(t-u)-u*t,-2`

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

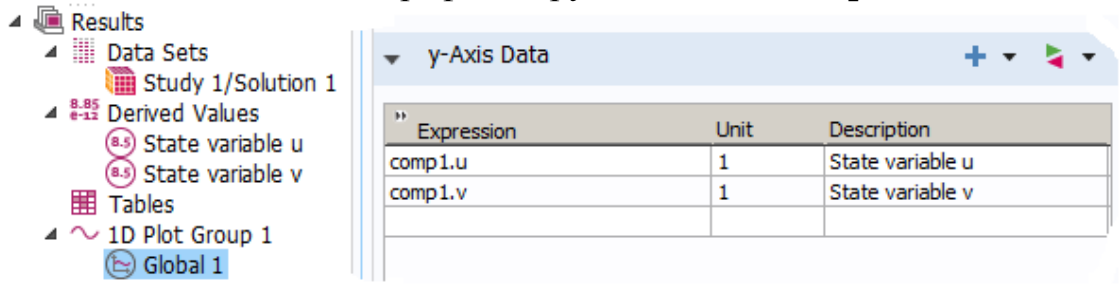
Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
u	u^2-u*(v*t+sin(t))+v*t*sin(t)	0	0
v	v^2-v*(t-u)-u*t	-2	0
		0	0

В таблиці використано початкові наближення $u(-3)=0, v(-3)=-2$. Те, що дані взяті при $t=-3$, визначається початком діапазону `range(-3,0.1,3)`. Самі значення задані в стовпці з заголовком **`Initial value (u_0)`**. Тут вони розглядаються як початкові наближення, а не як початкові значення.

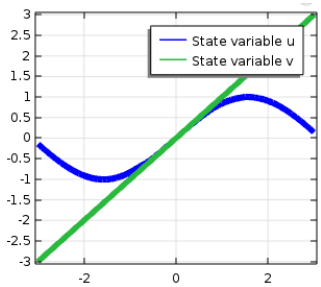
На закладці `Home` натисніть кнопку = Compute .

`Home` -> Compute

В гілці **Results-Derived Values** з'являються підвузли **State variable u** і **State variable v**. Також з'являється графічна група **1D Plot Group 1**.



Панель властивостей графічного вузла **Global 1** містить таблицю виразів, графіки яких побудовані.



Зверніть увагу на імена змінних в стовпці **`Expression`** таблиці виразів вузла **Global 1**. Вони записані через точку після назви компонента. Ім'я компонента **comp1** задається в ПВ вузла **Component 1** в полі **Name**. Задача може складатися з кількох компонентів, кожен з яких представляє окрему фізичну модель. Повні імена змінних, які складаються з імені компонента і ідентифікатора змінної, використовуються в багатьох випадках. Особливо це важливо тоді, коли розв'язуються мультифізичні задачі.

Надрукуйте значення знайдених величин `u`, `v` в спільну таблицю.

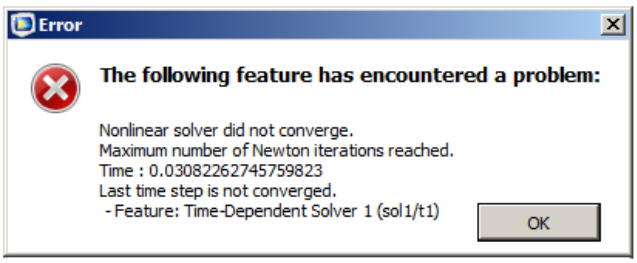
Derived Values – State variable u: LMB > Evaluate

Derived Values – State variable v: LMB > Evaluate - Table 1–State variable u

Time (s)	State variable u (1)	State variable v (1)
-0.30000	-0.29616	-0.30000
-0.20000	-0.19526	-0.20000
-0.100000	-0.099833	-0.100000
0.0000	-1.1875E-5	0.0000
0.10000	0.099737	0.10000
0.20000	0.19826	0.20000
0.30000	0.29466	0.30000

Таблиця результатів містить масиви значень розв'язків `u` і `v` в різні моменти часу з діапазону `range(-3,0.1,3)`.

Оскільки досліджувана система нелінійна, то вона може мати кілька розв'язків. Для їх пошуку змінимо початкові наближення. Наприклад, в таблиці рівнянь введіть значення `u0=0.4`, `v0=0.4`. Спроба розв'язання (натисніть кнопку = Compute) призводить до повідомлення про помилку: `Nonlinear solver did not converge.`



Щоб отримати розв'язок змініть відносну похибку, наприклад, зменшить її в 10 разів, поклавши R_t=`0.0001`.

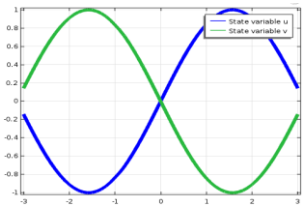
Step 1: Time Dependent:

`Study Settings` – **Relative tolerance** = `0.0001`

Тепер розв'яжіть задачу.

`Home` -> Compute

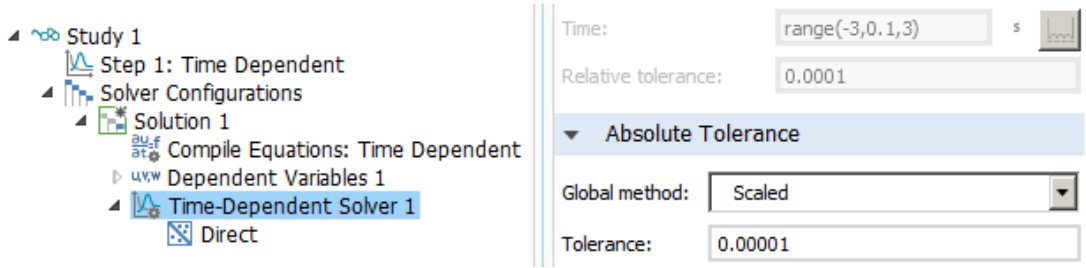
Графік у вузлі **Global 1** буде перебудовано. З нього зрозуміло, що отримана інша пара функцій.



Якщо ви введете нові початкові наближення, наприклад, `u0=0.2` і `v0=0.2`, то отримаєте ще одне повідомлення про помилку. Подальше зменшення відносної похибки не приводить до розв'язання задачі. Тоді слід зменшити абсолютну похибку обчислень.

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

`Absolute Tolerance` – **Tolerance** = `0.00001`



`Home` -> Compute

В нашому прикладі при початкових наближеннях `u0=0.2` і `v0=0.2` цього досить для отримання розв'язку.

В розділі **`Absolute Tolerance`** вузла **Time-Dependent Solver 1** є ще список **Global method**. В тих випадках, коли зменшення відносної і абсолютної похибки не призводить до результату, в цьому списку можна спробувати вибрати `Unscaled` метод. Крім того, у вас залишається можливість вибору іншого солвера (чисельного методу розв'язання). Це робиться в розділі **`General`** в випадному списку **Solver** ПВ вузла **Study 1-Solution 1-Stationary Solver 1-Direct**.

В таблиці рівнянь поверніть початкові наближення `u0=0.4`, `v0=0.4`, і ще раз переобчисліть модель.

`Home` -> Compute

Побудуйте графіки похідних $u'(t), v'(t)$ отриманих розв'язків. Для цього створіть ще одну одновимірну графічну групу і в ній графічний вузол **Global 1**. В таблиці цього вузла введіть вирази `d(comp1.u,t)` і `d(comp1.v,t)`.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 2` -> Add Plot – Global

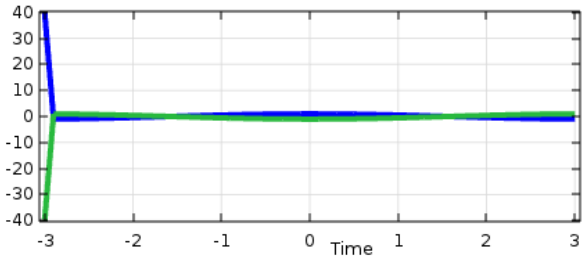
1D Plot Group 2 - Global 1:

`y-Axis Data` – **Expression** = `d(comp1.u,t)` (*перший рядок таблиці*)

`y-Axis Data` – **Expression** = `d(comp1.v,t)` (*другий рядок таблиці*)

Expression	Unit	Description
d(comp1.u,t)	1/s	
d(comp1.v,t)	1/s	

Натисніть кнопку Plot, і отримайте щось на зразок наступного графіка (у вас може вийти інше).



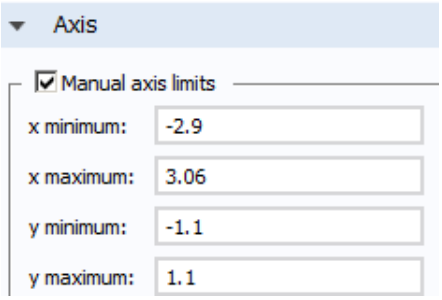
Уважно придивившись, можна зрозуміти, що причиною такого графіка є великі значення похідних в початковій точці діапазону. Через це вертикальний масштаб графіка став занадто великий. Справа в тому, що ми розв'язуємо алгебраїчні рівняння з параметром `t`, а не диференціальні, і тому похідна в початковій точці не задається. Як наслідок, вона була обчислена некоректно. Звідси випливає, що з графіка треба виключити невелику початкову ділянку. Це можна зробити в ПВ вузла **1D Plot Group 2** в розділі **`Axis`**, де задаються діапазони зміни горизонтальної та вертикальної координат.

1D Plot Group 2 :

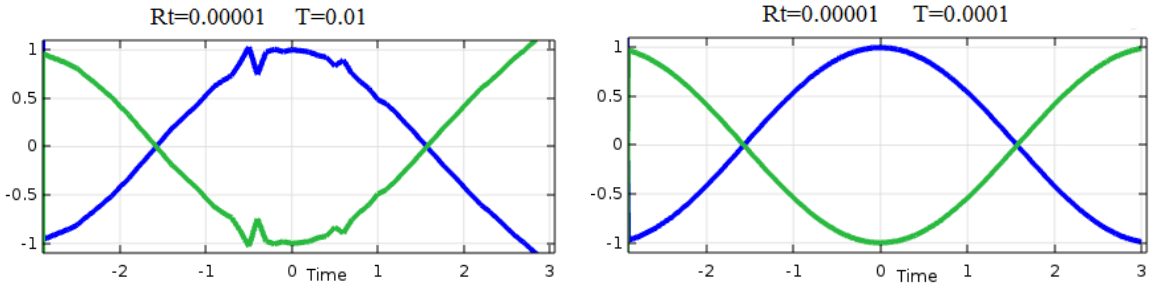
`Axis` – **Manual axis limits** = `True`

– **x minimum, x maximum** = `-2.9,3.0`

– **y minimum, y maximum** = `-1.1,1.1`



Ще раз натисніть кнопку Plot в ПВ вузла **1D Plot Group 2**. Графік похідних, отриманий при відносній похибці `Rt=0.00001` і абсолютній похибці `T=0.01`, показаний на наступному рисунку ліворуч. Праворуч показаний графік, отриманий при `Rt=0.00001`, `T = 0.00001`.

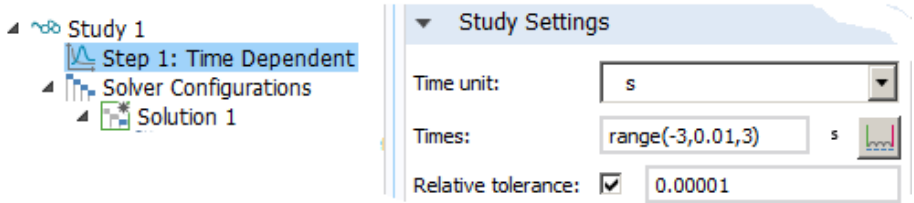


4.1.2. Задача Коші для звичайних диференціальних рівнянь.

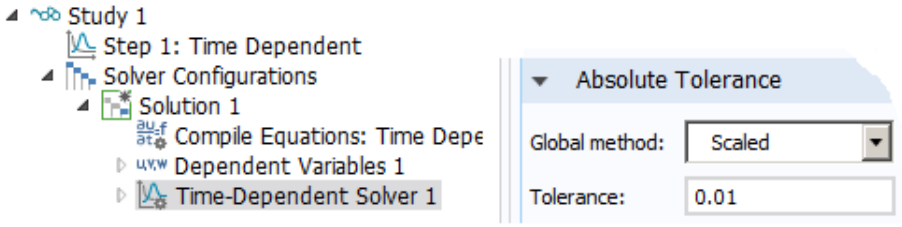
В останніх прикладах ви бачили, що в таблиці рівнянь в стовпці з заголовком **`f(u,ut,utt,t)`** можна вводити вирази, які містять не тільки імена шуканих змінних, таких як `u`, але і параметр часу `t`. В цьому випадку ми отримуємо рівняння (систему рівнянь) з параметром. У виразах ТР можна також використовувати перші (`ut`) і другі (`utt`) похідні шуканої функції по параметру часу. В такому випадку ми отримуємо звичайні диференціальні рівняння або їх системи.

Побудова моделі, призначеної для розв'язання диференціальних рівнянь, виконується з використанням динамічного Global ODEs and DAEs інтерфейсу в наступній послідовності:

1. За допомогою майстра створюється модель задачі.
   `File-> New ->Model Wizard-> 0D -> Mathematics - ODE and DAE Interfaces-Global ODEs and DAEs(ge)-> Time Dependent`
2. Зазвичай одиниці вимірювання шуканих величин ігноруються (фізичні величини призначаються безрозмірними). Для цього в ПВ кореневого вузла (або в гілці **Component 1**) в списку **Unit system** обирається `None`.
3. Задається діапазон зміни параметра `t`. Це виконується в ПВ вузла **Step 1: Time Dependent** в полі **Times**.
4. Налаштовується точність обчислень. Відносна похибка задається в ПВ вузла **Step 1: Time Dependent** в полі **Relative tolerance**.



Абсолютна похибка задається в полі **Tolerance** ПВ вузла **Study 1- Solver Configuration-Solution 1-Time-Dependent Solver 1** (звичайно, що номери вузлів можуть бути іншими) .



5. В ПВ вузла **Global Equations 1** заповнюється таблиця рівнянь і потім виконуються обчислення.

Завершується розв'язання задачі побудовою таблиць значень шуканих величин і графіків розв'язків.

Приклад 1. Розв'язати ЗДР $u''_{tt}+u=0$ з початковими умовами $u(0)=1, u_t'(0)=0$

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Задайте діапазон зміни параметра `t`.

Step 1: Time Dependent:

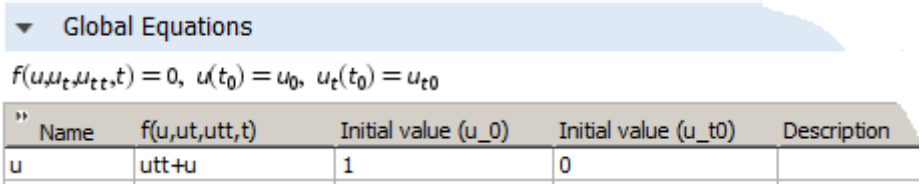
`Study Settings` – **Times** = `range(0,0.1,2*pi)`

Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `u, utt+u`

– **Initial value(u_0), Initial value(u_t0)** =`1,0`



Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)	Description
u	utt+u	1	0	

Стартове значення діапазону `range(0,0.1,2*pi)` (в нашому випадку 0) використовується в якості t_0 при задаванні початкових значень $u(t_0)=\mathrm{u_0}$, $u_t'(t_0)=\mathrm{u_t0}$.

Виконайте обчислення.

Home -> Compute

В гілці **Results** створюється вузол **1 D Plot Group 1 - Global 1**, в графічному вікні якого будується графік. Додайте на графік криві, що представляють першу

і другу похідні розв'язку. Для цього відкрийте ПВ вузла **Global 1** і в розділі **y-Axis Data** заповніть таблицю виразів.

1D Plot Group 1 - Global 1:

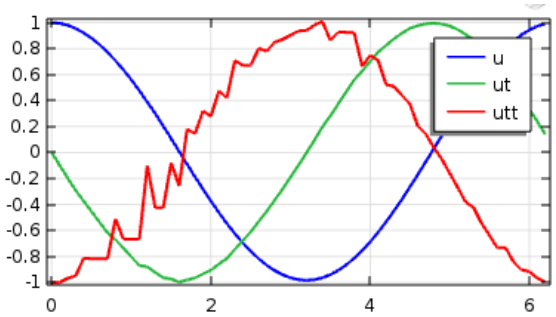
y-Axis Data – **Expression, Description** = comp1.u,u
– **Expression, Description** = comp1.ut,ut
– **Expression, Description** = comp1.utt,utt

В стовпці **Description** можна додати коментар (необов'язково).

y-Axis Data

Expression	Unit	Description
comp1.u		u
comp1.ut		ut
comp1.utt		utt

Натисніть кнопку Plot вгорі ПВ. Результівний графік (при значеннях відносної і абсолютної похибок за замовчуванням R_t=0.01, T=0.001) показаний на наступному рисунку.



Зверніть увагу, що тексти стовпця **Description** таблиці виразів автоматично перенесені на легенду графіка.

У цієї задачі існує точний розв'язок $\cos t$. Побудуємо графік різниці наближеного і точного розв'язків.

Results –> Plot Group - 1D Plot Group

1D Plot Group 2 –> Add Plot – Global

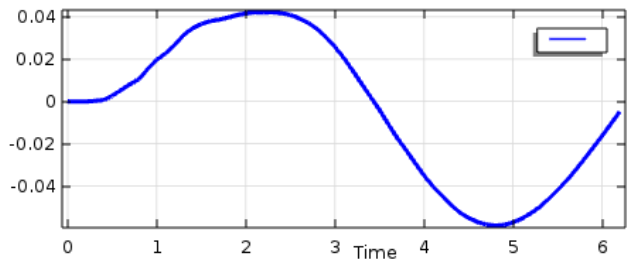
1D Plot Group 2 - Global 1:

y-Axis Data – **Expression** = comp1.u-cos(t)

1D Plot Group 2
Global 1

Expression	Unit	Description
comp1.u-cos(t)		

LMB > Plot



Останній графік побудований при відносній і абсолютній похибках обчислень R_t=0.01, T=0.001. З графіка видно, що максимальне значення модуля різниці точного і наближеного розв'язків приблизно дорівнює 0.05.

Підвищити точність обчислень можна, зменшивши в гілці **Study 1** відносну і абсолютну похибки.

Study 1 – Step 1: Time Dependent:

`Study Settings` – **Relative tolerance** = `0.0001`

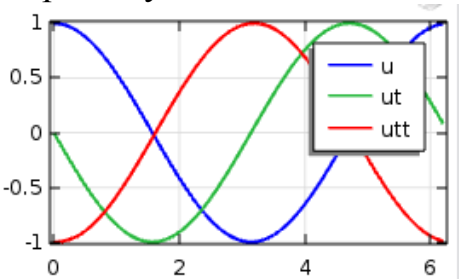
Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

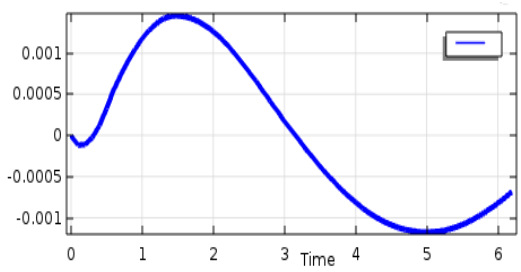
`Absolute Tolerance` – **Tolerance** = `True,0.0001`

Повторіть обчислення.

`Home -> Compute`

Графіки похідних стануть більш гладкими (показані на наступному рисунку ліворуч), а з графіка різниці `comp1.u-cos(t)` (показаний праворуч) видно, що вона не перевищує $2 \cdot 10^{-3}$.





Як бачимо, величина абсолютної похибки обчислень, яка задається в ПВ вузла **Time-Dependent Solver 1**, не збігається з максимумом різниці наближеного і точного розв'язків.

Маючи розв'язок, можна побудувати його первісну. Для інтегрування виразів по часу $\int_{t_1}^{t_2} \mathrm{expr}(t)\,dt$ слід використовувати оператор `timeint`. Він має наступний синтаксис:

`timeint(t1,t2,expr)`,

де інтегрування за часом виконується в межах від `t1` до `t2`, і змінну інтегрування вказувати непотрібно. До того ж межі інтегрування `t1,t2` не зобов'язані бути константами (можуть залежати від часу). У форматі

`timeint(t1,t2,expr,tol)`,

крім іншого, задається відносна похибка обчислення інтеграла, яка за замовчуванням дорівнює `1e-8`.

Аналогічний синтаксис має оператор `timeavg`.

`timeavg(t1,t2,expr)` або `timeavg(t1,t2,expr,tol)`.

Він обчислює середнє значення виразу `expr` на відрізку часу `t1`, `t2` за формулою $\dfrac{1}{t_2 - t_1}\int_{t_1}^{t_2} \mathrm{expr}(\tau)\,d\tau$.

В нашому прикладі для тренування побудуємо графіки функцій $f_1(t) = \int_0^t u^2(\tau)\,d\tau$ і $f_2(t) = \int_0^t u(\tau)\,d\tau$. Для цього в гілці **Results** створіть нову одновимірну графічну групу з `Global` вузлом і таблицею виразів, графіки яких будуть побудовані (заповнюється колонка **`Expression`**, в колонці **`Description`** нумеруємо вирази).

`Results -> Plot Group - 1D Plot Group`

`1D Plot Group 3` -> Add Plot – Global

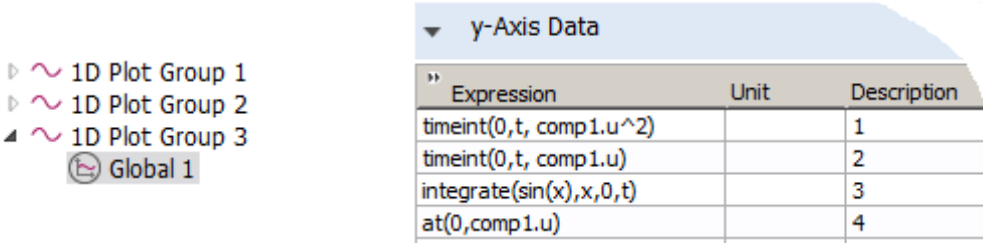
1D Plot Group 3 - Global 1:

y-Axis Data–**Expression, Description**=`timeint(0,t,comp1.u^2),1`

– **Expression, Description** = `timeint(0,t, comp1.u),2`

– **Expression,Description**=`integrate(sin(x),x,0,t),3`

– **Expression, Description** = `at(0,comp1.u),4`



В третьому рядку (для порівняння) ми використали оператор `integrate`, який має наступний синтаксис:

`integrate(expr(x),x,x1,x2)`,

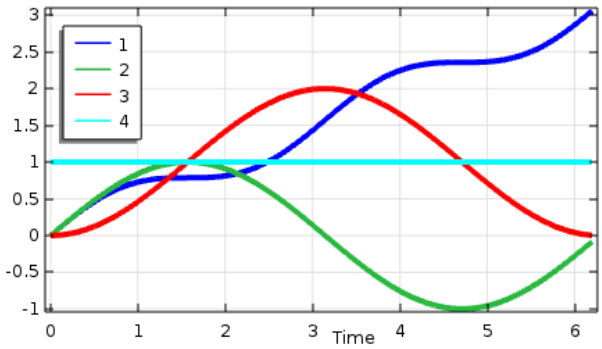
де `expr(x)` являє підінтегральний вираз, `x` – ім'я змінної інтегрування, `x1` і `x2` - нижня і верхня межі інтегрування. Таким чином, в третьому рядку записано вираз $f_3(t)=\int_0^t \sin x\,dx$.

В четвертому рядку ми використали оператор `at`. Він повертає значення розв'язку (або якогось виразу) в заданий момент часу. Наприклад, запис `at(0,comp1.u)` повертає значення розв'язку в нульовий момент часу, і графік цього виразу є горизонтальною прямою.

Перевірте, що у вузлі **Global 1** в розділі **x-Axis Data** в списку **Parameter** обрано `Time` (час), скорегуйте розташування легенди, і побудуйте графіки.

1D Plot Group 3 : **Legend** – **Position** = `Upper left`

LMB > Plot



Тут текст стовпця **Description** таблиці виразів вузла **1D Plot Group 3 - Global 1** автоматично перенесений на легенду графіків.

Наведемо ще два оператори `timemax` і `timemin`. Вони мають синтаксис

```
timemax(t1,t2,expr)
timemin(t1,t2,expr)
```

і обчислюють максимальне і мінімальне значення виразу `expr` на відрізку часу `t1≤t≤t2`. У форматі

```
timemax(t1,t2,expr,'nointerp')
timemin(t1,t2,expr,'nointerp')
```

вони обчислюють максимальне і мінімальне значення без виконання інтерполяції між точками розбиття часового інтервалу.

Наприклад, для обчислення максимальної різниці між наближеним розв'язком `u` і точним `cos(t)` на відрізку часу від `0` до `6.2` виконаємо наступні інструкції.

`Results` -> **`Derived Values`** – Global Evaluation

Derived Values - Global Evaluation 2:

`Expression` – **Expression** = `timemax(0,6.2,abs(comp1.u-cos(t)))`

`Data Series Operation` – **Operation** = `Maximum`

LMB > Evaluate

Maximum: timemax(0,6.2,comp1.u-cos(t))
0.0014551

Отже максимальна різниця між наближеним (`comp1.u`) і точним (`cos(t)`) розв'язками становить `0.0014551`.

Функції `timeint`, `timeavg` , `at`, `timemax`, `timemin` викликаються при обробці результатів. Їх можна використовувати тільки в виразах, що стоять в полях **Expression** вікон властивостей вузлів гілки **Results** і не можна використовувати при побудові моделі.

Приклад 2. Розв'язати ЗДР $4u''_{tt}+0.8u'_t+2u=20$ з початковими умовами $u(0)=-1,\ u'_t(0)=0$.

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): **`Unit System`** – <> = `None`

Задаєйте діапазон зміни параметра `t` і відносну похибку.

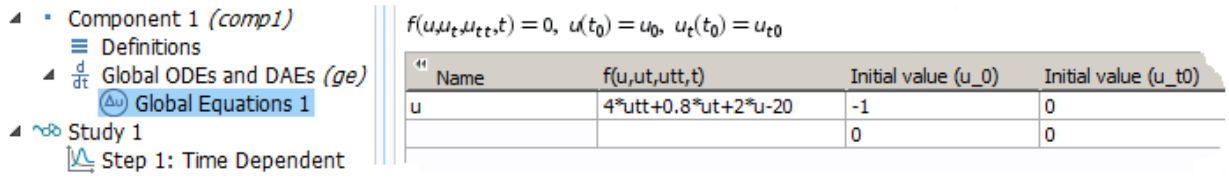
Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.1,70)`
– **Relative tolerance** = `True, 0.001`

Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `u, 4*utt+0.8*ut+2*u-20`
– **Initial value(u_0), Initial value(u_t0)** =`-1,0`

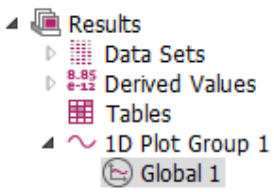


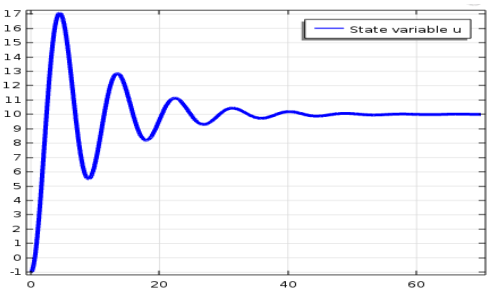
Перший момент діапазону `range(0,0.1,70)` (в нашому прикладі 0) використовується в якості `t0` при задаванні початкових значень `u_0`, `u_t0`.

Виконайте обчислення.

`Home` -> Compute

В гілці **Results** створюється вузол **1 D Plot Group 1 - Global 1**, в графічному вікні якого будується графік.





Вгорі ПВ вузла **Results - Derived Values - State variable u** натисніть кнопку Evaluate. В правому нижньому куті вікна програми з'явиться панель результатів з таблицею значень розв'язку.

Time	State variable u
0.0000	-1.00000
0.10000	-0.96659
0.20000	-0.88541
0.30000	-0.75267
0.40000	-0.56953
0.50000	-0.33788
0.60000	-0.059817
0.70000	0.26275

Приклад 3. Розв'язати два непов'язаних диференціальних рівняння

$$\begin{cases} y_t' + 5t^3 y = 0, \qquad y(0) = 0.01 \\ z'' + 0.2\, z_t' + z = 0,\ z(0) = 0,\ z_t'(0) = 1 \end{cases}.$$

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Задайте діапазон зміни параметра `t` і відносну похибку.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(-1.5,0.01,1.5)`

– **Relative tolerance** = `True, 0.00001`

Заповніть таблицю рівнянь. Нагадаємо, що ми не радимо називати шукані змінні іменами просторових координат. Тому ми їх позначимо літерами `u` і `v`.

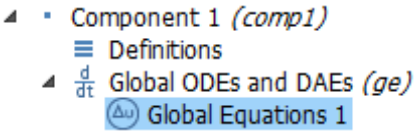
Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `u, ut+5*u*t^3`

– **Initial value(u_0), Initial value(u_t0)** = `0.01, 0`

`Global Equations` – **Name, f(u, ut, utt, t)** = `v, vtt+0.2*vt+v`

– **Initial value(u_0), Initial value(u_t0)** = `0, 1`



Global Equations

$f(u,u_t,u_{tt},t) = 0,\ u(t_0) = u_0,\ u_t(t_0) = u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
u	ut+5*u*t^3	0.01	0
v	vtt+0.2*vt+v	0	1

Виконайте обчислення.

`Home` **->** Compute

На отриманому графіку налаштуйте легенду.

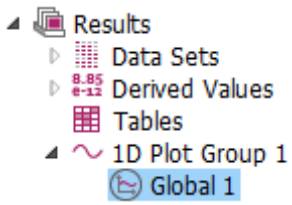
1D Plot Group 1 – Global 1:

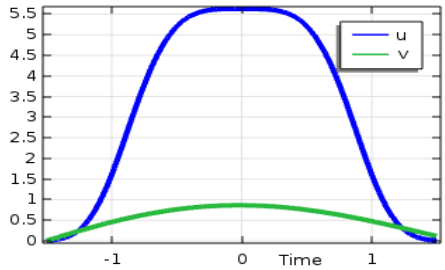
`Legends` – **Show legends, Legends,** = `True, Manual`

– **Legends** = `u` (*рядок* 1)

– **Legends** = `v` (*рядок* 2)

Графіки розв'язків показано нижче.





Приклад 4. Розв'язати систему двох ЗДР першого порядку

$$\begin{cases} \dfrac{d\,y_1}{d\,t} - y_2 = 0.01 \cdot t^2, \; y_1(0) = 0 \\ \dfrac{d\,y_2}{d\,t} + y_1 = 0 \qquad , y_2(0) = 1 \end{cases}$$

і побудувати фазову траєкторію.

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): **`Unit System`** – <> = `None`

Заповніть таблицю рівнянь.

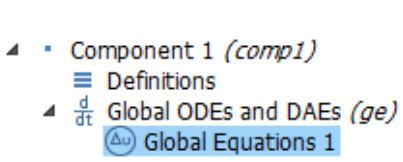
Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `y1, y1t-y2-0.01*t^2`

– **Initial value(u_0), Initial value(u_t0)** = `0, 0`

`Global Equations` – **Name, f(u, ut, utt, t)** = `y2, y2t+y1`

– **Initial value(u_0), Initial value(u_t0)** = `1, 0`



$f(u,u_t,u_{tt},t) = 0, \; u(t_0) = u_0, \; u_t(t_0) = u_{t0}$

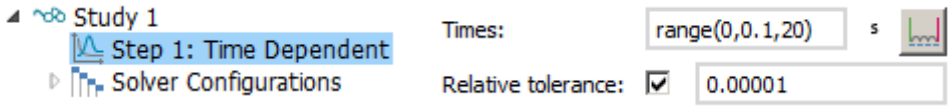
Name	f(u,ut,utt,t)	Initial value	Initial value
y1	y1t-y2-0.01*t^2	0	0
y2	y2t+y1	1	0
		0	0

Задайте діапазон зміни параметра `t` і відносну похибку.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.1,20)`

– **Relative tolerance** = `True, 0.00001`



Виконайте обчислення.

`Home` **->** Compute

На отриманому графіку налаштуйте легенду.

1D Plot Group 1 – Global 1:

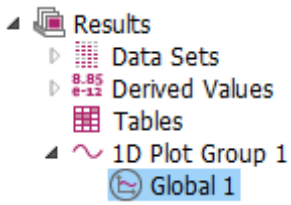
`Legends` – **Show legends, Legends,** = `True, Manual`

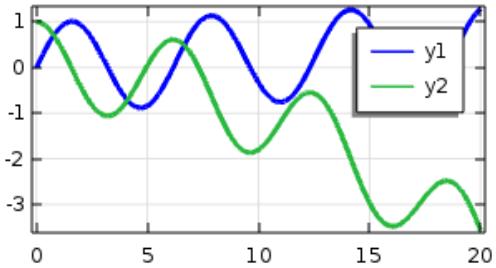
– **Legends** = `y1` (*рядок* 1)

– **Legends** = `y2` (*рядок* 2)

LMB > Plot

Графіки розв'язків показані на наступному рисунку.





Для побудови фазової траєкторії створіть і налаштуйте графічний вузол **1D Plot Group 2 - Global 1**.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 2` -> Add Plot – Global

1D Plot Group 2 :

`Legend` – **Show legends** = `False`

1D Plot Group 2 - Global 1:

`y-Axis Data` – **Expression** = `comp1.y2`

`x-Axis Data` – **Parameter, Expression** = `Expression, comp1.y1`

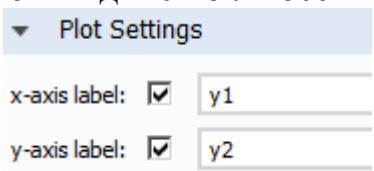
Останні вирази кажуть, що вертикальні координати кривої відповідають значенням `y2`, а горизонтальні - значенням `y1`.

`Title` – **Title type** = `None`

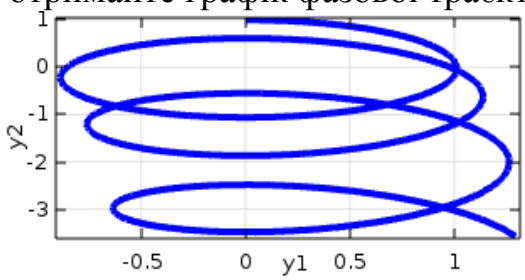
`Plot Settings` – **x-axis label** = `True,y1`

– **y-axis label** = `True,y2`

Останні дві опції створюють підписи біля осей на графіку.



Натисніть кнопку Plot і отримайте графік фазової траєкторії.



Приклад 5. Розв'язати задачу Коші для ЗДР 2-го порядку $z''_{tt} + 0.2 \cdot z'_t + z = 0$, $z(0) = 0$, $z'_t(0) = 1$, і побудувати фазову траєкторію.

Розв'язання. Послідовність дій така ж, як в попередніх прикладах.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

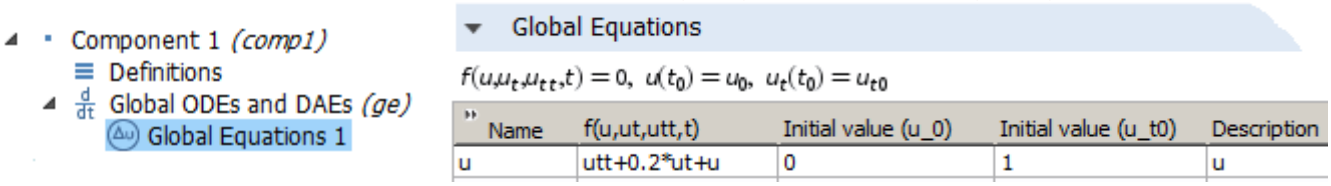
Untitled.mph (root): `Unit System` – <> = None

В якості ідентифікатора шуканої величини візьмемо літеру `u`.

Global ODEs and DAEs(ge) - Global Equations 1:

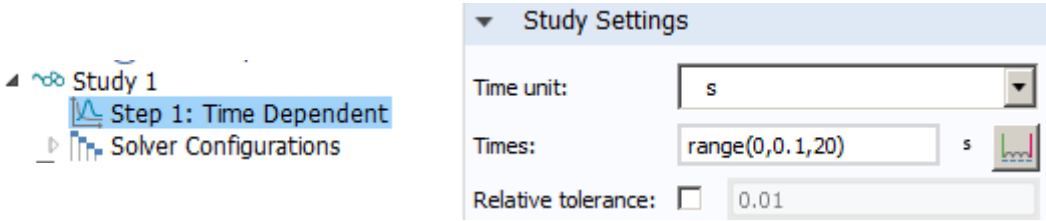
`Global Equations` – Name, f(u, ut, utt, t) = `u, utt+0.2*ut+u`

– Initial value(u_0), Initial value(u_t0) = `0,1`



Step 1: Time Dependent:

`Study Settings` – Times = `range(0,0.1,20)`



`Home` -> Compute

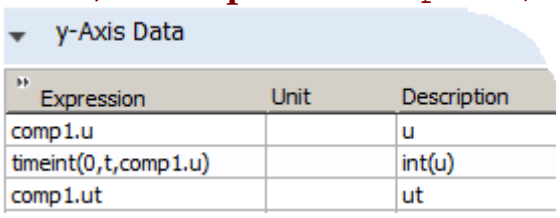
В гілці **Results** з'явиться вузол з графіком розв'язку. Додайте на нього графіки первісної і похідної.

1D Plot Group 1 - Global 1:

`y-Axis Data` – Expression, Description = `comp1.u,u`

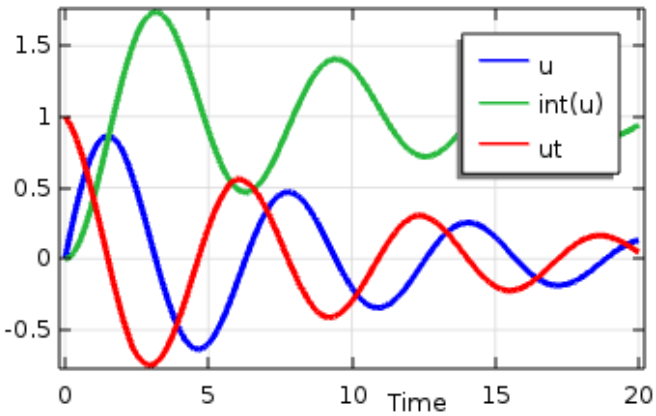
– Expression, Description =`timeint(0,t,comp1.u),int(u)`

– Expression, Description = `comp1.ut,ut`



Тут ми ще заповнили стовпець **`Description`**, вміст якого (за замовчуванням) з'явиться на легенді графіків.

LMB > Plot



Побудуйте фазову траєкторію. Для цього створіть другу графічну групу **1D Plot Group 2** з вузлом `Global`.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 2` -> Add Plot – Global

1D Plot Group 2:

`Legends` – **Show legends** = `False`

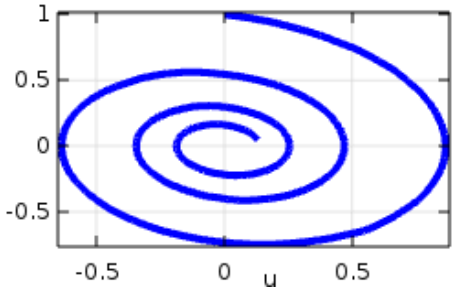
1D Plot Group 2 - Global 1:

`y-Axis Data` – **Expression** = `comp1.ut`

`x-Axis Data` – **Parameter, Expression** = `Expression, comp1.u`

При побудові фазової траєкторії в якості вертикальної координати використовується величина `comp1.ut`, а в якості горизонтальної - `comp1u`.

LMB > Plot



■

Вільні члени рівнянь можуть містити інтегральні вирази. Існує два способи їх використання. Один полягає у застосуванні оператора `integrate(...)`, а другий – у введенні додаткового диференціального рівняння, розв'язання якого призводить до того ж інтегрування.

Приклад 6. Розв'язати задачу Коші

$$v''_{tt}(t)+2v'_t(t)+v(t)=\int_0^t \cos(10\cos\tau)\,d\tau,\ v(0)=0, v'(0)=0.$$

Якщо позначити $u'_t(t)=\cos(10\cos t)$ і додати умову $u(0)=0$, то це можна розглядати як задачу Коші з розв'язком $u(t)=\int_0^t \cos(10\cos\tau)\,d\tau$. Тоді входову задачу можна замінити системою наступного вигляду

$$v''_{tt}(t)+2v'_t(t)+v(t)=u(t), v(0)=0, v'(0)=0$$

$$u'_t(t)=\cos(10\cos t),\ u(0)=0$$

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

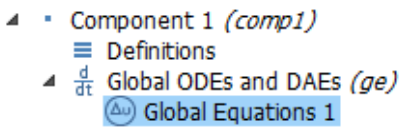
Untitled.mph (root): `Unit System` – <> = None

Таблицю рівнянь заповніть даними перебудованої системи.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – Name, f(u, ut, utt, t) = `u, ut-cos(10*cos(t))`
– Initial value(u_0) = `0`

`Global Equations` – Name, f(u, ut, utt, t) = `v, vtt+2*vt+v-u`
– Initial value(u_0), Initial value(u_t0) = `0, 0`



$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
u	ut-cos(10*cos(t))	0	0
v	vtt+2*vt+v-u	0	0

Задайте діапазон зміни параметра `t`.

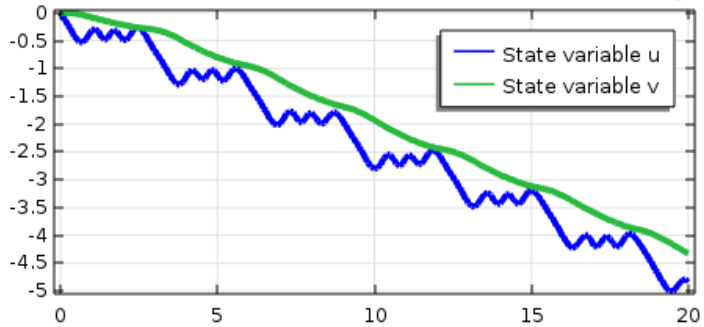
Step 1: Time Dependent:

`Study Settings` – Times = `range(0,0.1,20)`

Виконайте обчислення.

`Home` -> Compute

Графіки отриманих розв'язків показані на наступному рисунку.



Збережіть модель в файл, але не закривайте його. В цьому файлі розв'яжемо ще одне ЗДР

$$V''_{tt}+V=\int_0^{\pi}\cos(t\cos\tau)d\tau\,,V(0)=0,V'(0)=0.$$

Якщо позначити $U(t)=\int_0^{\pi}\cos(t\cos\tau)d\tau$, то цю задачу можна замінити системою наступного вигляду

$$\begin{aligned}&U(t)=\int_0^{\pi}\cos(t\cos\tau)d\tau\,,U(0)=\pi\\&V''_{tt}+V=U\,,V(0)=0,V'(0)=0\end{aligned}.$$

Додайте в файл попередньої моделі ще один `Global Equations` вузол.

Global ODEs and DAEs(ge): RMB > Global Equations

Таблицю рівнянь створеного вузла **Global Equations 2** заповніть даними перебудованої системи.

Global ODEs and DAEs(ge) - Global Equations 2:

Global Equations – **Name, f(u, ut, utt, t), Initial value(u_0)** = `U, U-integrate(cos(t*cos(tau)),tau,0,pi), pi`

Global Equations – **Name, f(u, ut, utt, t)** = `V, Vtt+V-U`

– **Initial value(u_0), Initial value(u_t0)** = `0, 0`

Знову виконайте обчислення.

Home -> Compute

Побудуйте графіки отриманих розв'язків U і V.

1D Plot Group 1 : RMB > Duplicate

1D Plot Group 2 - Global 1:

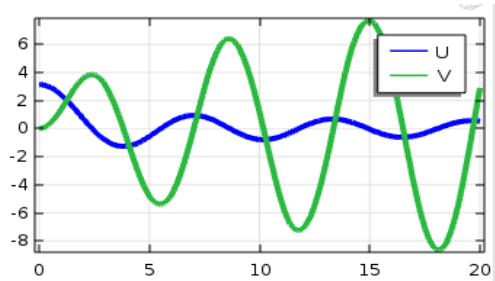
y-Axis Data – **Expression, Description** = `comp1.U,U`

– **Expression, Description** = `comp1.V,V`

y-Axis Data

Expression	Unit	Description
comp1.U		U
comp1.V		V

LMB > Plot



Вправа 1. Розв'язати задачу $u''_{tt} - u'_t + u = \cos\left(\int_0^t x \cdot \sqrt{t}\, d\,x\right), u(0) = 1, u'(0) = 0$.

Вправа 2. Побудувати графік невизначеного інтеграла $\int e^{-x^2} \sin^2 4x\, d\,x$.

Вказівка. Розв'язати задачу Коші $u'_t = e^{-t^2} \sin 4t$ з початковою умовою $u(0) = 0$.

Приклад 7. Розв'язати задачу Коші для нелінійного ЗДР 2-го порядку $u''_{tt} + u^3 - \sin t = 0$, $u(0) = 0$, $u'_t(0) = 0$, і побудувати фазову траєкторію.

Розв'язання. Послідовність інструкцій така ж, як і в попередніх прикладах.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Untitled.mph (root): Unit System – <> = None

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name, f(u, ut, utt, t)** = `u, utt+u^3-sin(t)`

– **Initial value(u_0), Initial value(u_t0)** = `0,0`

Global ODEs and DAEs *(ge)*
Global Equations 1

Global Equations

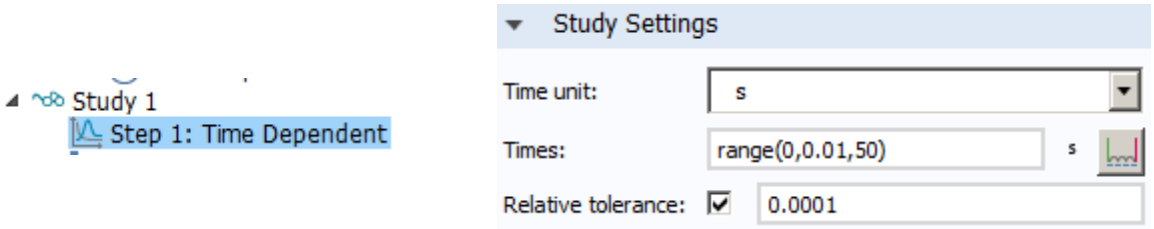
$f(u,u_t,u_{tt},t) = 0,\ u(t_0) = u_0,\ u_t(t_0) = u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
u	utt+u^3-sin(t)	0	0

Step 1: Time Dependent:

Study Settings – **Times** = `range(0,0.01,50)`

– **Relative tolerance** = `True, 0.0001`

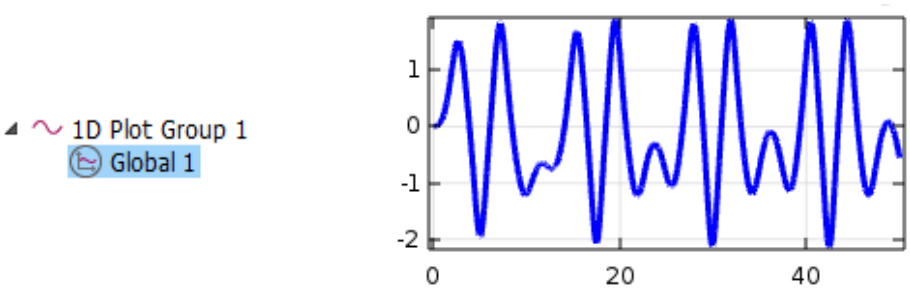


`Home` -> Compute

1D Plot Group 1:

`Legends` – **Show legends** = `False`

LMB > Plot



Фазова траєкторія будується наступними інструкціями.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 2` -> Add Plot – Global

1D Plot Group 2:

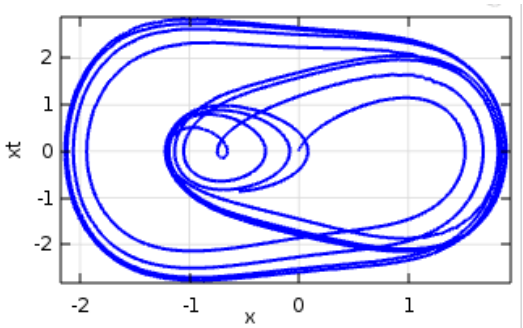
`Legends` – **Show legends** = `False`

1D Plot Group 2 - Global 1:

`y-Axis Data` – **Expression** = `comp1.ut`

`x-Axis Data` – **Parameter, Expression** = `Expression, comp1.u`

LMB > Plot



Приклад 8. Розв'язати задачу Коші для нелінійної системи двох ЗДР першого порядку

$$\begin{cases} u_t' - v = 0 & , u(0) = 0 \\ v_t' + 0.01 \cdot v + \sin(u) = 0 & , v(0) = 2.1 \end{cases},$$

і побудувати фазову траєкторію.

Розв'язання. Послідовність інструкцій схожа на попередні приклади.

File -> New -> Model Wizard -> 0D -> Mathematics -

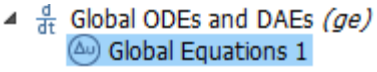
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Untitled.mph (root): **`Unit System`** – <> = None

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name, f(u, ut, utt, t)** = `u, ut-v`
– **Initial value(u_0)** = `0`

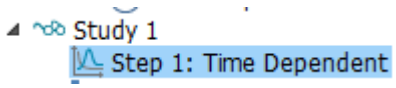
Global Equations – **Name, f(u, ut, utt, t)** = `v, vt+0.01*v+sin(u)`
– **Initial value(u_0)** = `2.1`

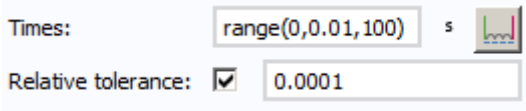


Name	f(u,ut,utt,t)	Initial value (u_0)
u	ut-v	0
v	vt+0.01*v+sin(u)	2.1

Step 1: Time Dependent:

Study Settings – **Times** = `range(0,0.1,100)`
– **Relative tolerance** = `True, 0.0001`





`Home` -> Compute

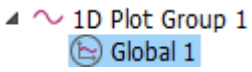
1D Plot Group 1 :

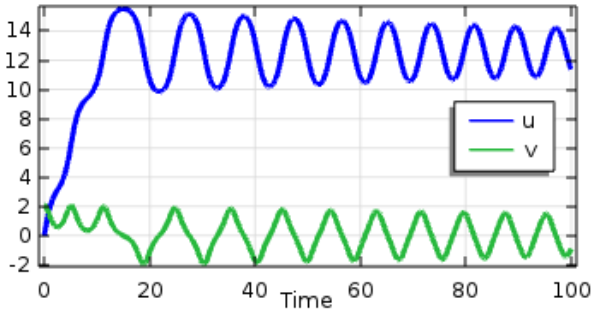
Legend – **Position** = `Midle right`

1D Plot Group 1 - Global 1:

Legends – **Show legends, Legends,** = `True, Manual`
–**Legends (row 1)** = `u`
–**Legends (row 2)** = `v`

LMB > Plot





Для побудови фазової траєкторії виконайте наступні інструкції.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 2` -> Add Plot – Global

1D Plot Group 2:

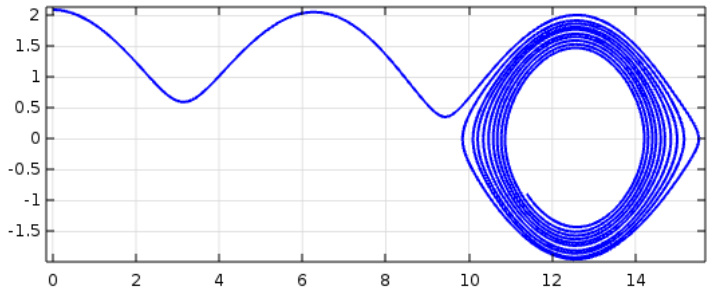
Legends – **Show legends** = `False`

1D Plot Group 2 - Global 1:

y-Axis Data – **Expression** = `comp1.v`

x-Axis Data – **Parameter, Expression** = `Expression, comp1.u`

LMB > Plot



Приклад 9. Розв'язати задачу Коші для нелінійної системи двох ЗДР першого порядку

$$\begin{cases} x_t' - y = 0 & , x(0) = 0 \\ y_t' + x^3 - x = 0 & , y(0) = 0.1 \end{cases}$$

і побудувати фазову траєкторію.

Розв'язання. Фазову траєкторію можна будувати, використовуючи графічний вузол типу `Point Trajectories`. Але для цього потрібно мати двовимірну модель. Перевага такого підходу полягає в тому, що він легко узагальнюється на тривимірний випадок (тривимірні фазові траєкторії).

Послідовність розв'язання задачі така ж, як і в попередніх прикладах, за виключенням того, що створюється двовимірна модель.

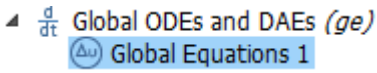
File -> New -> Model Wizard -> 2D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

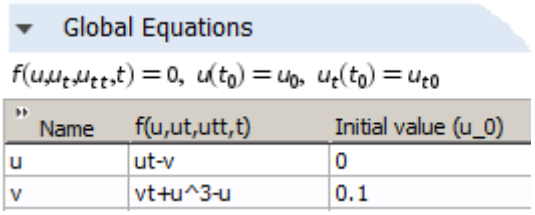
Untitled.mph (root): `Unit System` – <> = None

Шукані змінні позначимо літерами `u` і `v`.

Global ODEs and DAEs(ge) - Global Equations 1:

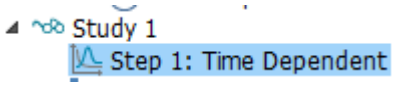
`Global Equations` – Name, f(u, ut, utt, t) = `u, ut-v`
– Initial value(u_0) = `0`
`Global Equations` – Name, f(u, ut, utt, t) = `v, vt+u^3-u`
– Initial value(u_0) = `0.1`

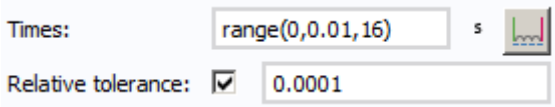




Step 1: Time Dependent:

`Study Settings` – Times = `range(0,0.1,16)`
– Relative tolerance = `True, 0.0001`





`Home` -> Compute

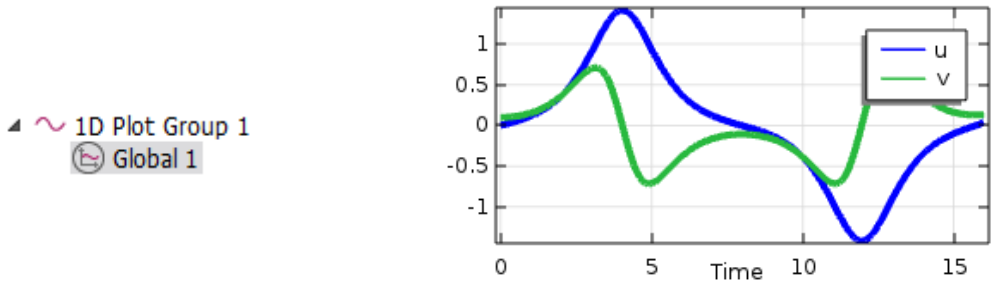
1D Plot Group 1 - Global 1:

`y-Axis Data` – Expression, Description = `comp1.u, u`
`y-Axis Data` – Expression, Description = `comp1.v, v`

Текст в стовпці **`Description`** за замовчуванням використовується в якості легенди графіка.

Expression	Unit	Description
comp1.u		u
comp1.v		v

LMB > Plot



Почнемо будувати фазову траєкторію. Для цього спочатку створіть двовимірну графічну групу.

`Results` -> Plot Group - 2D Plot Group

В створену графічну групу додайте вузол `Point Trajectories`.

`2D Plot Group 2` -> **`More Plots`** -> Point Trajectories

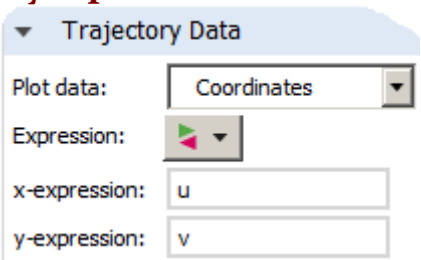
2D Plot Group 2 - Point Trajectories 1 :

`Trajectory Data` – **Plot data** = `Coordinates`

Опція **Plot data**=`Coordinates` означає, що траєкторія точки буде задаватися виразами в наступних полях `expression`.

`Trajectory Data` – **x-expression** = `u`

– **y-expression** = `v`



Налаштуйте стиль кривої.

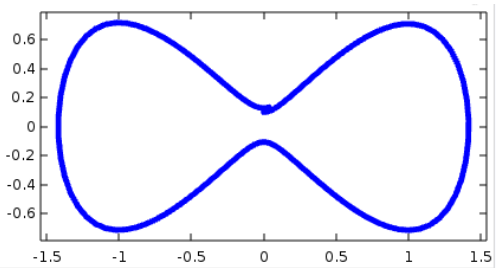
`Coloring and Style` – **`Line style`** – **Type** =`Tube`

– **`Line style`** – **Tube radius expression** =`0.01`

– **`Line style`** – **Radius scale factor** =`True,1`

– **Color** =`Blue`

LMB > Plot



Приклад 10. Розв’язати систему трьох диференціальних рівнянь

$$\begin{cases} u_t' - v + w = 0 & , u(0) = 1 \\ v_t' - w + u = 0 & , v(0) = 0 \\ w_t' - u + 2 \cdot v = 0, & w(0) = 2 \end{cases},$$

і побудувати її фазову траєкторію.

Розв’язання. Оскільки фазова траєкторія системи є кривою в тривимірному просторі, то для її побудови нам знадобиться тривимірна модель.

File -> New -> Model Wizard -> 3D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Untitled.mph (root): `Unit System` – <> = None

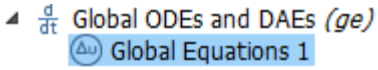
Для побудови графіків розв'язку виконуємо інструкції такі самі, як і в попередніх прикладах.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t) =** `u, ut-v+w`
– **Initial value(u_0)** = `1`

`Global Equations` – **Name, f(u, ut, utt, t) =** `v, vt-w+u`
– **Initial value(u_0)** = `0`

`Global Equations` – **Name, f(u, ut, utt, t) =** `w, wt-u+2*v`
– **Initial value(u_0)** = `2`

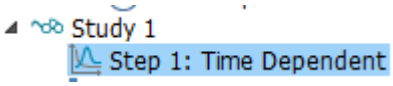


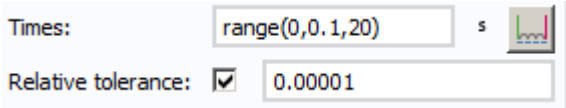
$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)
u	ut-v+w	1
v	vt-w+u	0
w	wt-u+2*v	2

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.1,20)`
– **Relative tolerance** = `True, 0.0001`





`Home` -> Compute

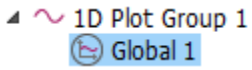
1D Plot Group 1 : `Legend` – **Position** = `Upper left`

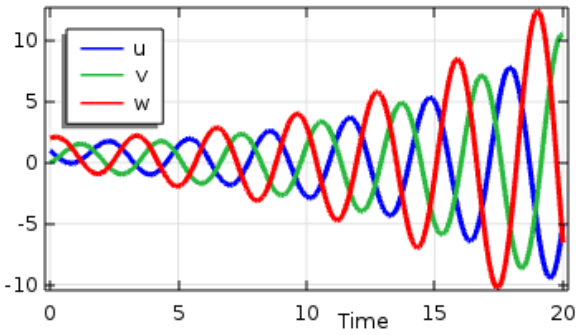
1D Plot Group 1 - Global 1:

`y-Axis Data` – **Expression, Description** = `comp1.u, u`
– **Expression, Description** = `comp1.v, v`
– **Expression, Description** = `comp1.w, w`

Нагадаємо, що текст в стовпці **`Description`** використовується в якості легенди графіка.

LMB > Plot





Для побудови фазової траєкторії створіть тривимірну графічну групу, в яку додайте вузол `Point Trajectories`.

`Results` -> Plot Group - 3D Plot Group

`3D Plot Group 2` -> **`More Plots`** -> Point Trajectories

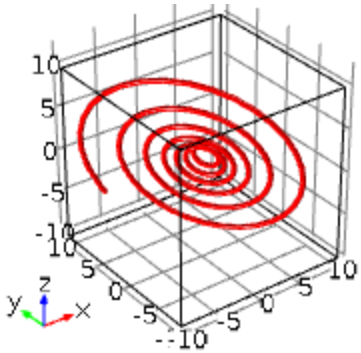
3D Plot Group 2 - Point Trajectories 1 :

Trajectory Data – **x-expression** = u
– **y-expression** = v
– **z-expression** = w

Налаштуйте стиль кривої.

Coloring and Style – **Line style** – **Type** =Tube
– **Line style** – **Tube radius expression** =0.3
– **Line style** – **Radius scale factor** =True,1
– **Color** =Red

LMB > Plot



■

В Comsol Multiphysics можна розв'язувати ЗДР з параметрами. Для цього в гілці **Study** створюється підвузол **Parametric Sweep**, або у вузлі **Study 1 – Step 1: Stationary** в розділі **Study Extensions** встановлюється перемикач Auxiliary Sweep (допоміжний аналіз). В другому випадку вважається, що параметр використовується в коефіцієнтах рівнянь або в початкових умовах і не застосовується для визначення геометричних розмірів.

Приклад 11. Розв'язати задачу Коші

$$x''_{tt} + 0.15 \cdot x'_t - x + x^3 - 0.3 \cdot \cos t = 0,\ x(0) = -1, x'_t(0) = x_{t0},$$

де x_{t0} - параметр, який приймає значення $1, 1.02, 1.04, 1.06, 1.08, 1.1$.

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

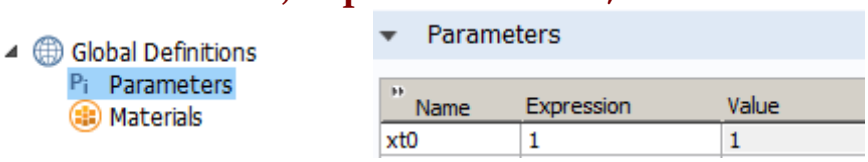
Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

У вузлі **Global Definitions** створіть вузол **Parameters** і в його таблиці задайте параметр xt0.

Global Definitions : RMB > Parameters

Parameters – **Name, Expression** = xt0, 1



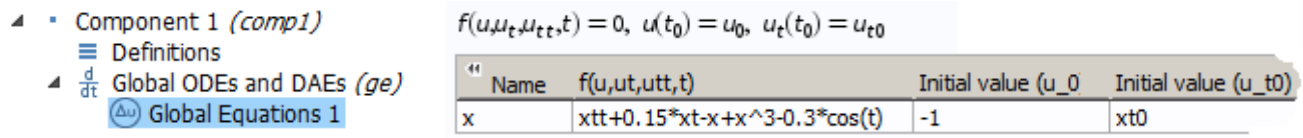
Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name, f(u, ut, utt, t)** =

u, utt+0.15*ut-u+u^3-0.3*cos(t)

– **Initial value(u_0), Initial value(u_t0)** = -1, xt0

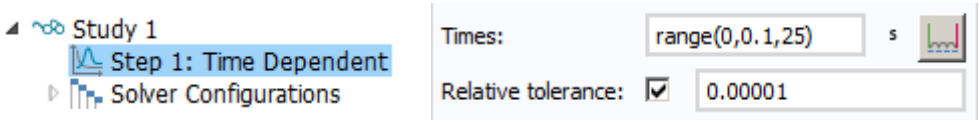


Задайте діапазон зміни параметра t і відносну похибку.

Step 1: Time Dependent:

Study Settings – **Times** = range(0,0.1,25)

– **Relative tolerance** = True, 0.00001



Для визначення залежності розв'язку рівняння від параметра використаємо Auxiliary Sweep (допоміжний) аналіз, який вмикається встановленням відповідного прапорця в ПВ вузла **Study – Step 1: Stationary** (нагадаємо що, коли параметр не змінює геометрію моделі, то доцільніше використовувати Auxiliary Sweep параметричний аналіз).

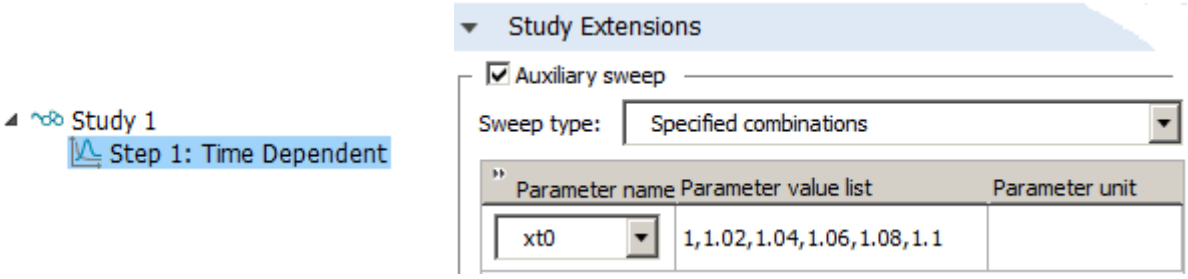
Study 1 – Step 1: Stationary:

Study Extensions – **Auxiliary sweep** = True

В ПВ вузла **Step 1: Stationary** в розділі **Study Extensions** активується таблиця вибору параметрів. Клацніть по таблиці правою кнопкою миші і з випадного меню виберіть команду Add. Зі списку, який з'явиться в стовпці **Parameter name**, виберіть параметр xt0. У другому стовпці таблиці задайте множину значень цього параметра.

Study Extensions – **Parameter name** = xt0

– **Parameter value list** = 1,1.02,1.04,1.06,1.08,1.1



Виконайте обчислення моделі.

Home -> Compute

В гілці **Results** створиться вузол **1D Plot Group 1-Global 1**. Скорегуйте його легенду і перебудуйте графіки.

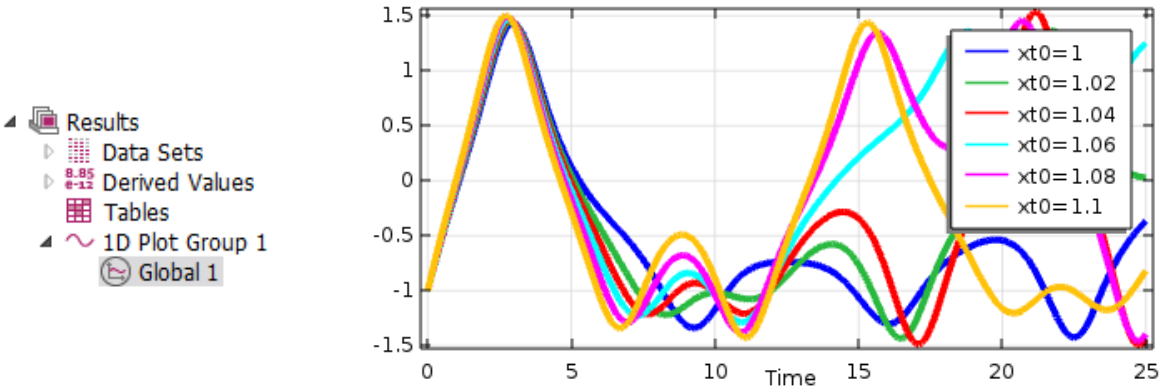
1D Plot Group 1 - Global 1:

Legends – **Show legends, Legends,** = True, Automatic

– **Description** = False

Вимкнення перемикача **Description** призводить до відображення на легенді лише значень параметра xt0.

LMB > Plot



Фазові траєкторії будуються у `Global` вузлі аналогічно попереднім прикладам.

`Results` –> Plot Group - 1D Plot Group

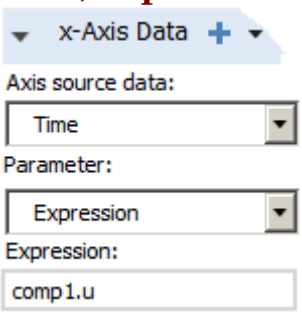
`1D Plot Group 2` –> Add Plot – Global

1D Plot Group 2 - Global 1:

`y-Axis Data` – **Expression** = `comp1.ut`

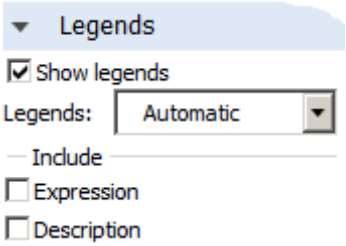
`x-Axis Data` – **Axis source data** = `Time`

– **Parameter, Expression** = `Expression, comp1.u`

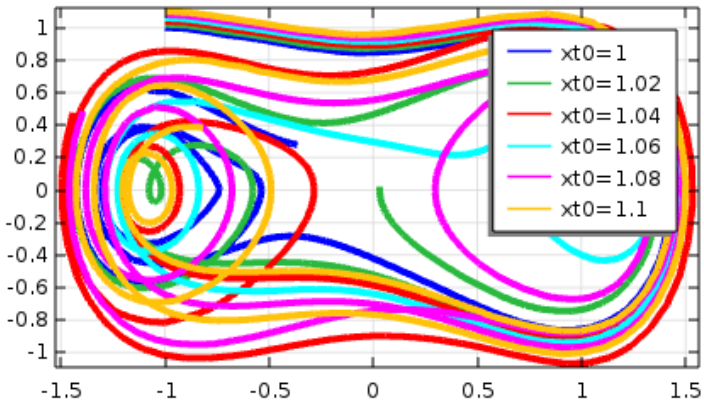


`Legends` – **Show legends, Legends,** = `True, Automatic`

– **Description** = `False`

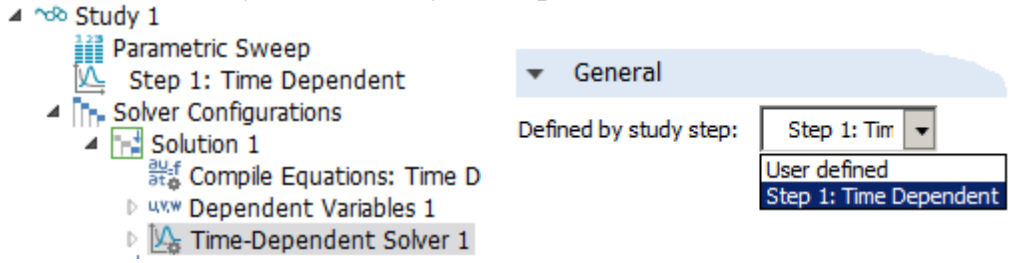


LMB > Plot



Іноді потрібно перевіряти, що у вузлі **Study 1 – Solver Configurations – Solution 1 - Time Dependent Solver 1** встановлені відносна похибка і діапазон зміни часу такі ж самі, як у вузлі **Step1: TimeDependent**. Для цього в списку

Defined by study step повинно бути обрано `Step 1:Time Dependent`. В принципі, ці налаштування можуть відрізнятися.



■

Параметричний аналіз можна виконувати за кількома величинами.

Приклад 12. Розв'язати рівняння

$$\frac{du}{dt} = \frac{\sin u}{\sin t},$$

з початковою умовою, яка буде управлятися двома параметрами $u(t_0) = p1 + p2$.

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

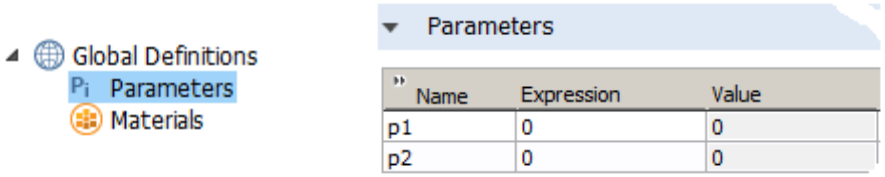
Untitled.mph (root): `Unit System` – <> = None

У вузлі **Global Definition**s створіть вузол **Parameters**, і в його таблиці задайте параметри `p1` та `p2` з нульовими значеннями.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `p1, 0` (row1)

– **Name, Expression** = `p2, 0` (row2)

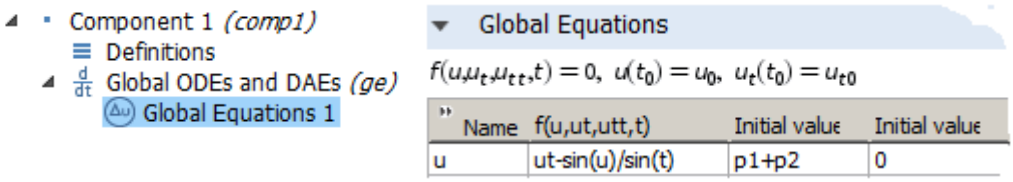


Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `u,ut-sin(u)/sin(t)`

– **Initial value(u_0)** = `p1+p2`

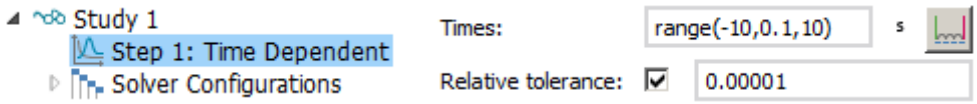


Задайте діапазон зміни параметра `t` і відносну похибку.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(-10,0.1,10)`

– **Relative tolerance** = `True, 0.00001`



Отже $t_0 = -10$.

В цій задачі бажано відразу зменшити абсолютну похибку. Але необхідний для її введення вузол в дереві моделі спочатку не відображається. Тому виконайте команду

`Study` -> Solver - Show Default Solver.

У вузлі **Study 1** з'явиться кілька дочірніх вузлів і можна буде виконати інструкцію

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

`Absolute Tolerance` – **Tolerance** = `0.00001`

Для виконання параметричного аналізу встановіть відповідний прапорець в ПВ вузла **Study – Step 1: Stationary**.

Study 1 – Step 1: Stationary:

`Study Extensions` – **Auxiliary sweep** = `True`

В розділі **`Study Extensions`** активується таблиця вибору параметрів. Клацніть по таблиці правою кнопкою миші і з випадного меню виберіть команду Add. Зі списку, який з'явиться в стовпці **Parameter name**, виберіть параметр `p1`. У другому стовпці таблиці задайте множину значень цього параметра.

`Study Extensions` – **Parameter name** = `p1`

– **Parameter value list** =`-0.75*pi,-0.5*pi, -0.25*pi,0.25*pi,0.5*pi, 0.75*pi`

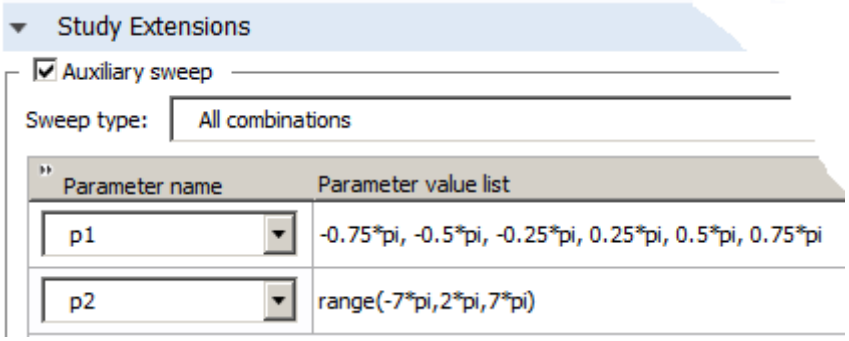
Зверніть увагу, що ми пропустили нуль. У цю ж таблицю додайте другий параметр `p2`.

`Study Extensions` – **Parameter name** = `p2`

– **Parameter value list** = `range(-7*pi,2*pi,7*pi)`

Щоб аналізувалися всілякі поєднання значень параметрів `p1` і `p2`, у списку **Sweep Types** виберіть `All combinations`.

`Study Extensions` – **Sweep Types** = `All combinations`



Виконайте обчислення моделі.

`Home` -> Compute

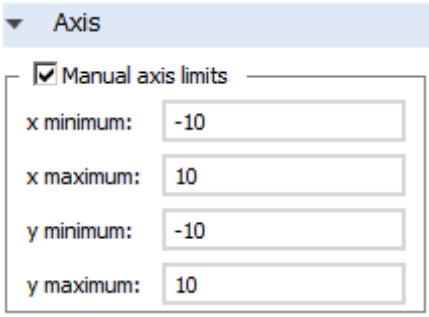
У створеному вузлі **1D Plot Group 1** налаштуйте параметри оформлення, зокрема, задайте мінімальні (`-10`) і максимальні (`+10`) значення горизонтальних і вертикальних координат графіка.

1D Plot Group 1 :

Axis – **Manual axis limits** = `True`

– **x minimum, x maximum** = `-10,10`

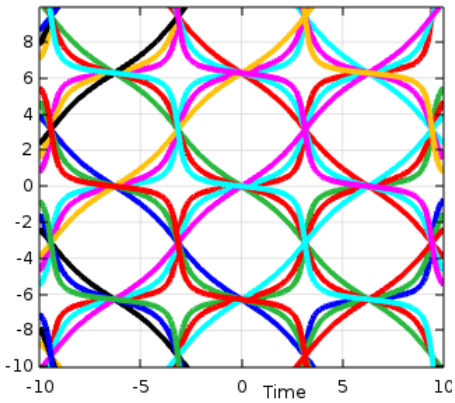
– **y minimum, y maximum** = `-10,10`



Legend – **Show legends** = `False`

В ПВ вузла **1D Plot Group 1 - Global 1** перевірте, що в розділі **y-Axis Data** в таблиці записано вираз `comp1.u` і, при бажанні, налаштуйте стиль кривих в розділі **Coloring and Style.**

Натисніть на кнопку Plot. Множина кривих, які представляють розв'язки задачі при різних початкових умовах $u(-10) = p1 + p2$, формує наступний «візерунок».



Насправді це множина кривих, побудованих при різних початкових значеннях. Пояснимо це, відобразивши лише частину кривих, які відповідають значенням параметрів: `p1=-0.75*pi` (першому значенню в множині першого параметра), `p2=-7*pi,-5*pi,-3*pi,-pi, pi` (перші п'ять значень в списку другого параметра).

1D Plot Group 1 : RMB > Duplicate

1D Plot Group 2 :

Data – **Parameter selection (p1)** = `Manual`

– **Parameter values (p1)(1-6)** = `1`

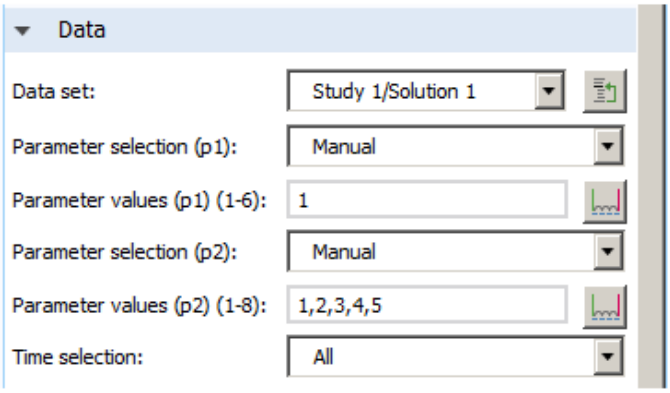
Data – **Parameter selection (p2)** = `Manual`

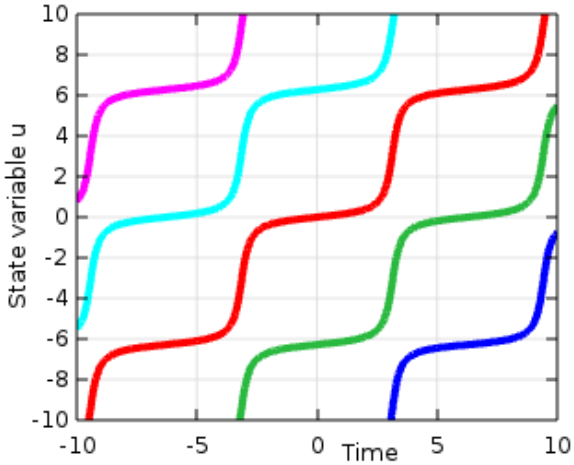
– **Parameter values (p2)(1-8)** = `1,2,3,4,5`

Тут вводяться номери параметрів, а не їх значення.

LMB > Plot

На наступному рисунку праворуч зображено результівний графік. Отже різні сполучення значень параметрів `p1` і `p2` дають різні криві, які утворюють «візерунок» попереднього рисунка.



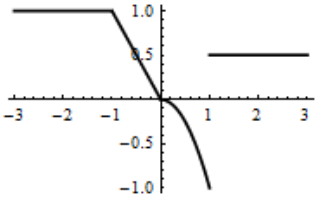


Система успішно розв'язує задачу Коші для «кускових» ЗДР.

Приклад 13. Розв'язати задачу

$$u''_{tt} + f^2(u)\cdot u = 0,\ u(0) = -1,\ u'_t(0) = 0,$$

де $f(x)$ кусково задана функція, яка має наступний вигляд.



$$f(x) = \begin{cases} 1\ , x \le -1 \\ -x\ , -1 < x \le 0 \\ -x^2\ , 0 < x \le 1 \\ 0.5\ , x > 1 \end{cases} \qquad (1)$$

Побудувати також фазову траєкторію.

Р о з в ' я з а н н я . Створіть двовимірну модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 2D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Створіть кускову функцію (1).

Global Definitions: RMB > Functions – Piecewise

Function name = `psw`

`Definition` – **Extrapolation** = `Constant`

В таблиці **`Intervals`** задайте початок і кінець інтервалів, а також вираз для кожної ділянки.

`Definition` – **Intervals (таблиця)** -

– **Start, End, Function** = `-4,-1,1`
– **Start, End, Function** = `-1,0,-x`
– **Start, End, Function** = `0,1, -x^2`
– **Start, End, Function** = `1,4,0.5`

Intervals

Start	End	Function
-4	-1	1
-1	0	-x
0	1	-x^2
1	4	0.5

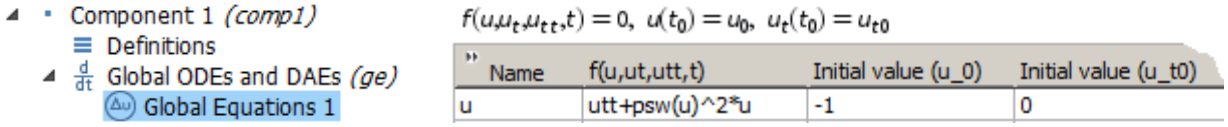
Вибір значення `Constant` в списку **`Extrapolation`** означає, що поза заданих ділянок функція продовжується константами - своїми значеннями, які вона набуває на лівому і правому краях області визначення. Клацання по кнопці Plot вгорі вікна будує графік функції (для самоперевірки).

Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `u,utt+psw(u)^2*u`

– **Initial value(u_0), Initial value(u_t0)** =`-1,0`

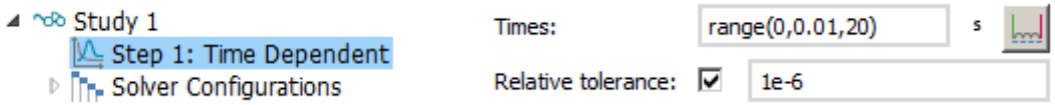


Задайте діапазон зміни параметра `t` і відносну похибку.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.01,20)`

– **Relative tolerance** = `True, 1e-6`



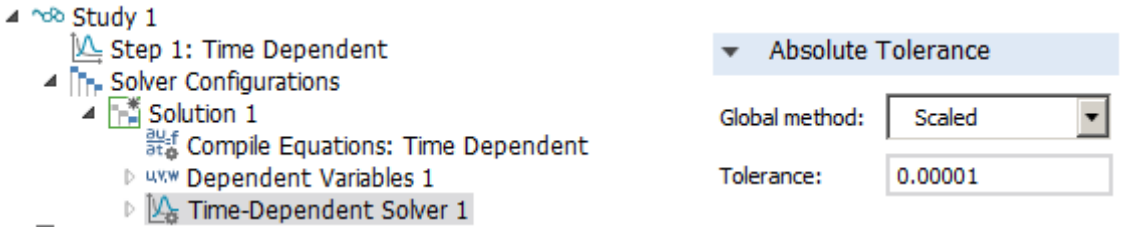
Виконайте обчислення моделі.

`Home` -> Compute

В гілці **Results** буде створено вузол **1D Plot Group 1 - Global 1** з графіком розв'язку. Однак отриманий результат не зовсім коректний. У цій моделі слід зменшити абсолютну похибку обчислень, яка задається в ПВ вузла **Study 1 - Solver Configurations - Solution 1 - Time-Dependent Solver 1** (цей вузол з'являється після першого обчислення або ви можете створити його відразу командою Show Default Solver).

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

`Absolute Tolerance` – **Tolerance** = `0.00001`

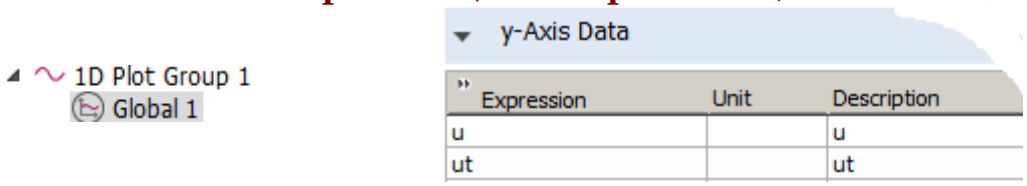


Повторно натисніть кнопку Compute. Потім добудуйте у вузлі **1D Plot Group 1 - Global 1** графік похідної розв'язку. Для цього в ПВ вузла в таблиці виразів заповніть другий рядок, а також скорегуйте стовпець **`Description`**.

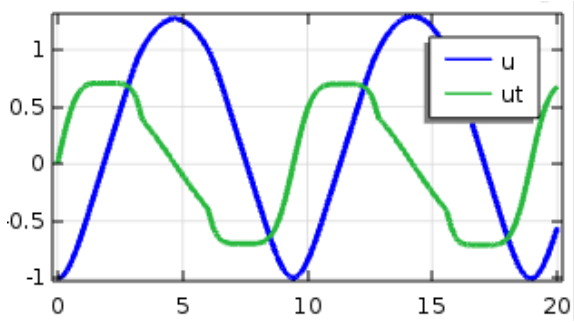
1D Plot Group 1 - Global 1:

`y-Axis Data` – **Expression, Description** =`u, u`

– **Expression, Description** =`ut, ut`



LMB > Plot



Для побудови фазової траєкторії створіть двовимірну графічну групу, в яку додайте вузол `Point Trajectories`.

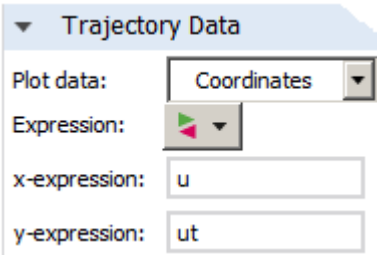
`Results` -> Plot Group - 2D Plot Group

`2D Plot Group 2` -> **`More Plots`** -> Point Trajectories

2D Plot Group 2 - Point Trajectories 1 :

`Trajectory Data` – **x-expression** = `u`

– **y-expression** = `ut`

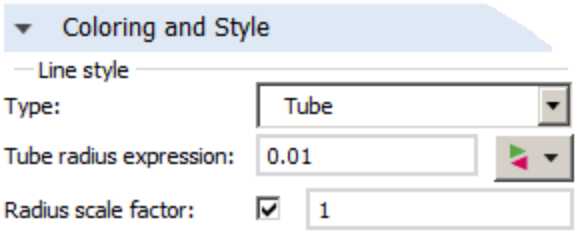


Налаштуйте стиль кривої.

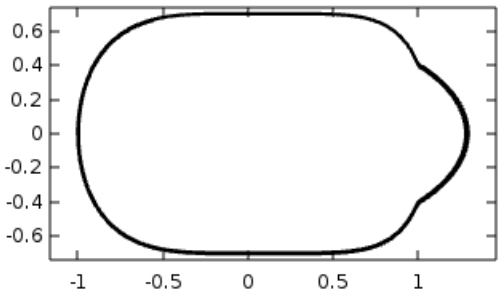
`Coloring and Style` – **`Line style`** – **Type** =`Tube`

– **`Line style`** – **Tube radius expression** =`0.01`

– **`Line style`** – **Radius scale factor** =`True,1`



LMB > Plot



Зауваження. Кускові вирази можна задавати аналітичним способом в стилі Matlab. Коли логічний вираз з'являється в арифметичній формулі, то він інтерпретується як +1, якщо умова істинна, або 0, якщо умова хибна. Зокрема, кускову функцію (1), яка використовується в нашому прикладі, можна задати наступною формулою:

$$f(x)=\underbrace{(x<=-1)}_{\mathrm{I}}\underbrace{-(x>-1\,\&\&\,x<0)*x}_{\mathrm{II}}\underbrace{-(0<x\,\&\&\,x<=1)*x\text{^}2}_{\mathrm{III}}+\underbrace{(x>1)*0.5}_{\mathrm{IV}}\quad(2)$$

Тут символи подвійного амперсанда && позначають операцію логічного І.

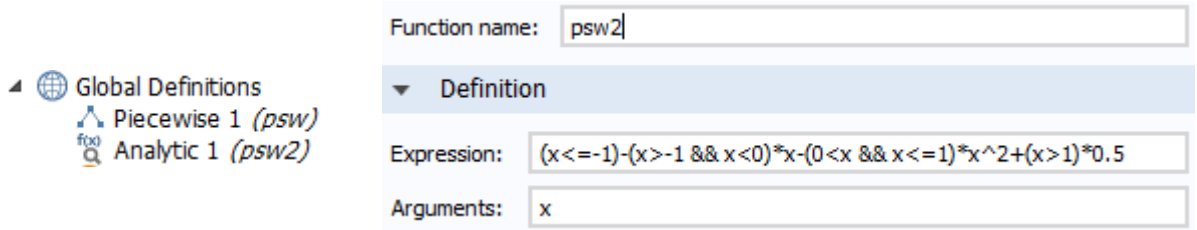
Проаналізуємо вираз (2). Якщо `x<=-1`, то в (2) доданок І дорівнює одиниці, а решта - дорівнюють нулю. Якщо `x>-1 && x<0`, то вираз в дужках у доданка II дорівнює одиниці, а сам доданок дорівнюватиме `-x`, в той час як інші члени в (2) дорівнюватимуть нулю. Аналогічно інші доданки в (2) будуть відмінні від нуля в разі, коли відповідний логічний вираз, записаний в дужках, набуває значення ІСТИНА. Проаналізувавши всі можливі випадки, бачимо, що формула (2) представляє функцію (1).

Перевіримо цей спосіб запису кускових виразів. Для цього в нашому прикладі створіть аналітичну функцію, задану виразом (2).

Global Definitions: RMB > Functions – Analitic.

Function name = `psw2`

Definition – Expression = `(x<=-1)-(x>-1 && x<0)*x-(0<x && x<=1)*x^2+(x>1)*0.5`



Потім замініть в ПВ вузла **Global Equations 1** в таблиці рівнянь ім'я функції `psw` на `psw2` і виконайте обчислення. Ви отримаєте той же результат, що і раніше (графік розв'язку і фазова траєкторія не зміняться).

Приклад 14. Розв'язати систему трьох диференціальних рівнянь першого порядку (Атрактор Лоренца)

$$\begin{cases} x_t' = s\cdot(y-x) \\ y_t' = x\cdot(r-z)-y, \\ z_t' = x\cdot y - b\cdot z \end{cases}$$

де покласти $s=10$, $r=25$, $b=3$ і узяти наступні початкові умови:

$$x(0)=1,\quad y(0)=-1,\ z(0)=10.$$

Розв'язання. Створіть тривимірну модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 3D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Задайте систему рівнянь і початкові умови.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – Name, f(u, ut, utt, t) = `u, ut-10*(v-u)`

– Initial value(u_0) = `1`

`Global Equations` – Name, f(u, ut, utt, t) = `v, vt-u*(25-w)+v`

– **Initial value(u_0)** = -1

Global Equations – **Name, f(u, ut, utt, t)** = w, wt-u*v+3*w

– **Initial value(u_0)** = 10

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Global ODEs and DAEs *(ge)*
Global Equations 1

Name	f(u,ut,utt,t)	Initial value	Initial value
u	ut-10*(v-u)	1	0
v	vt-u*(25-w)+v	-1	0
w	wt-u*v+3*w	10	0

Зауваження. В тривимірних задачах не називайте шукані величини іменами просторових змінних x,y,z. Це може призводити до конфлікту імен, і деякі побудови працювати не будуть.

Задайте діапазон зміни параметра t і відносну похибку.

Step 1: Time Dependent:

Study Settings – **Times** = range(0,0.01,40)

– **Relative tolerance** = True, 1e-6

Study 1
Step 1: Time Dependent
Solver Configurations

Times: range(0,0.01,40) s

Relative tolerance: ☑ 1e-6

Для чисельного дослідження подібних динамічних систем підходить метод Дорманда - Прінса. Виберть його.

Study –> Solver - Show Default Solver.

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

Absolute Tolerance – **Tolerance** = 0.000001

Time Stepping – **Method** = Dormand-Prince 5

Study 1
Step 1: Time Dependent
Solver Configurations
Solution 1
Compile Equations: Time Dependent
Dependent Variables 1
Time-Dependent Solver 1

Absolute Tolerance
Global method: Scaled
Tolerance: 0.00001

Time Stepping
Method: Dormand-Prince 5

Виконайте обчислення моделі.

Home -> Compute

У вузлі **Results** створюється графічний вузол **1D Plot Group 1 - Global 1**, який містить графіки розв'язків. Вони зображені на наступному рисунку. Ми лише скорегували легенду.

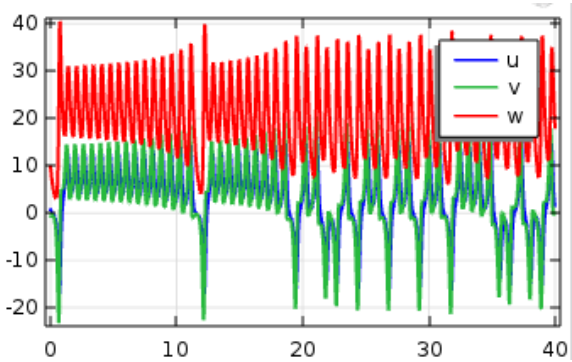
1D Plot Group 1 - Global 1:

Legends – **Show legends, Legends,** = True, Manual

–**Legends (row 1)** = u

–**Legends (row 2)** = v

–**Legends (row 3)** = w



Для побудови фазової траєкторії створіть тривимірну графічну групу, в яку додайте вузол `Point Trajectories`.

`Results` -> Plot Group - 3D Plot Group

`3D Plot Group 2` -> **`More Plots`** -> Point Trajectories

3D Plot Group 2 - Point Trajectories 1 :

`Trajectory Data` – **x-expression** = `u`

– **y-expression** = `v`

– **z-expression** = `w`

Налаштуйте стиль кривої.

`Coloring and Style` – **`Line style`** – **Type** =`Tube`

– **`Line style`** – **Tube radius expression** =`0.1`

– **`Line style`** – **Radius scale factor** =`True,1`

До вузла **Point Trajectories 1** додайте підвузол **Color Expression 1**.

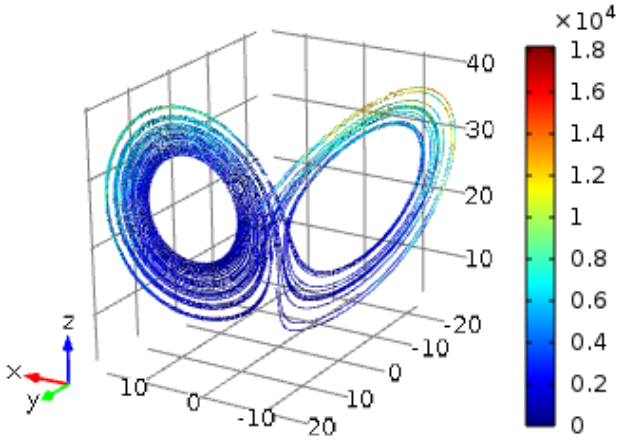
3D Plot Group 2 - Point Trajectories 1: RMB > Color Expression

3D Plot Group 2 - Point Trajectories 1 - Color Expression 1:

`Expression` – **Expression** =`sqrt(ut^2+vt^2+wt^2)`

Введений вираз являє величину швидкості руху точки по фазовій траєкторії і буде визначати колір кривої.

LMB > Plot



Зауваження. Система описує хаотичний атрактор, тому будь-які невеликі зміни в початкових умовах або в точності обчислень будуть призводити до істотної зміни розв'язку. Ви це зможете визначити за кількістю петель фазової траєкторії в правій частині графіка.

Приклад 15. Рівняння Ван дер Поля

$$y''_{tt} - \mu\left(1 - y^2\right)y'_t + y = 0 \qquad (3)$$

описує електричні коливання в замкнутому контурі, який складається з послідовно з'єднаних конденсатора, індуктивності, нелінійного елементу опору та елементу джерела енергії, де $y(t)$, позначає величину електричного струму в контурі.

Розв'язати рівняння (3) при $\mu = 1$, $y_t'(0) = 0$ і кількох різних початкових значеннях $y(0) = y_0$ електричного струму.

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

У вузлі **Global Definitions** створіть вузол **Parameters** і в його таблиці задайте параметр `u0`.

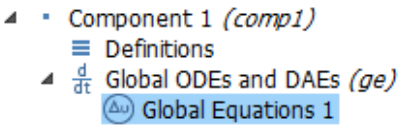
Global Definitions : RMB > Parameters

`Parameters` – Name, Expression = `u0, 0.1`

Задайте диференціальне рівняння.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – Name, f(u, ut, utt, t) =
`u, utt-(1-u^2)*ut+u`
– Initial value(u_0), Initial value(u_t0) = `u0, 0`



$f(u,u_t,u_{tt},t) = 0,\ u(t_0) = u_0,\ u_t(t_0) = u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
u	utt-(1-u^2)*ut+u	u0	0

Задайте діапазон зміни параметра `t` і відносну похибку.

Step 1: Time Dependent:

`Study Settings` – Times = `range(0,0.1,20)`
– Relative tolerance = `True, 0.00001`

Для визначення залежності розв'язку рівняння від параметра використайте `Auxiliary Sweep` (допоміжний) аналіз.

Study 1 – Step 1: Stationary:

`Study Extensions` – Auxiliary sweep = `True`

В ПВ вузла **Step 1: Stationary** в розділі **`Study Extensions`** активується таблиця вибору параметрів. Додайте в неї аналізовуваний параметр `u0`.

`Study Extensions` – Parameter name = `u0`
– Parameter value list = `range(0.1,0.3,1) range(2.1,0.3,3)`

Для виконання останніх дій клацніть по таблиці правою кнопкою миші і з випадного меню виберіть команду Add. Зі списку, який з'явиться в стовпці **Parameter name**, виберіть параметр `u0`. У другому стовпці таблиці **`Parameter value list`** задається множина зміни цього параметра.

Виконайте обчислення.

`Home` -> Compute

В гілці **Results** створюється вузол **1D Plot Group 1-Global 1**. Скорегуйте його графік, відобразивши лише перші чотири криві і підправивши легенду.

1D Plot Group 1:

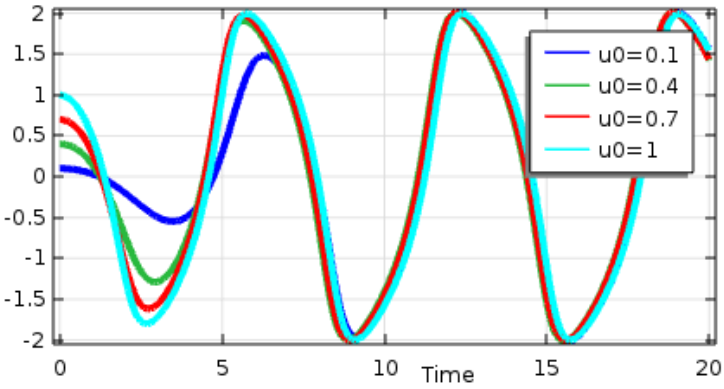
Data – **Parameter selection (u0)** = Manual

– **Parameter values (u0)(1-8)** = 1,2,3,4

1D Plot Group 1 - Global 1: Legends – **Description** = False

LMB > Plot

Розв'язком рівняння Ван дер Поля є коливання, вид яких зображено на наступному рисунку (при різних початкових значеннях u0).

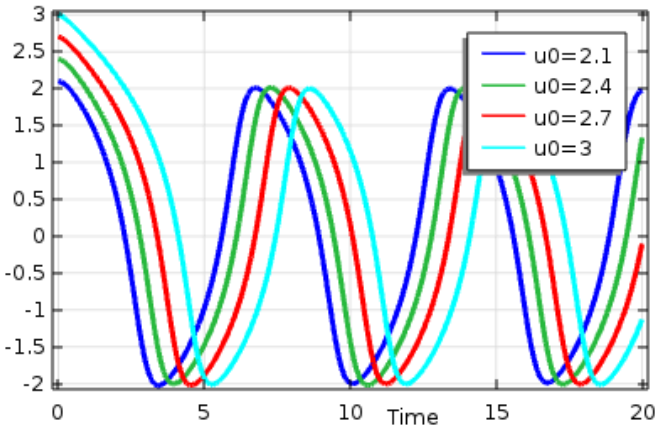


Тепер побудуйте графіки інших розв'язків.

1D Plot Group 1: RMB > Duplicate

1D Plot Group 2:

Data – **Parameter values (u0)(1-8)** = 5,6,7,8



Фазові траєкторії будуються у Global вузлі.

Results –> Plot Group - 1D Plot Group

1D Plot Group 3:

Legend – **Position** = Upper left

1D Plot Group 3 –> Add Plot – Global

1D Plot Group 3 - Global 1:

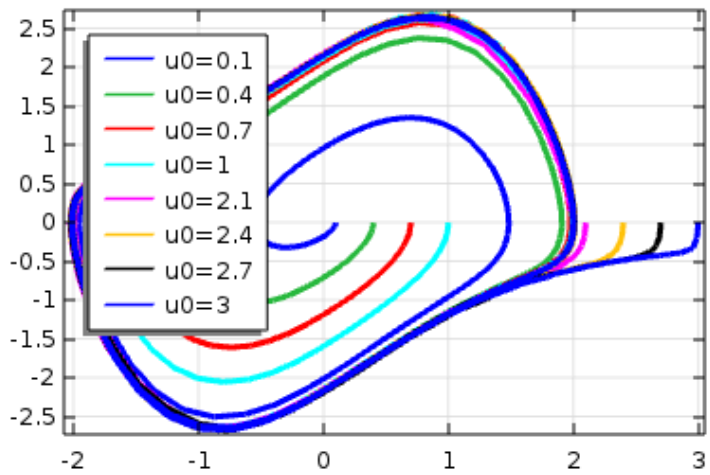
y-Axis Data – **Expression** = comp1.ut

x-Axis Data – **Axis source data** = Time

– **Parameter, Expression** = Expression, comp1.u

Legends – **Description** = False

LMB > Plot



Коливання, які спостерігаються на останніх графіках називаються автоколиваннями, бо їх характеристики (амплітуда, період) не залежать від початкових умов, а визначаються виключно властивостями самої динамічної системи. На останньому рисунку видно, що на фазовій площині існує замкнена крива, до якої асимптотично притягуються всі навколишні траєкторії, що виходять з різних початкових точок, як зсередини, так і ззовні граничного циклу. В результаті через деякий час розв'язок виходить на один і той же граничний цикл коливань (атрактор).

4.1.3. Диференціально-алгебраїчні системи рівнянь.

Задача, яка зв'язує невідомі функції, може містити як диференціальні, так і алгебраїчні рівняння. Тоді її називають системою **д**иференціально - **а**лгебраїчних **р**івнянь (ДАР). В таких задачах важливо слушно ставити початкові умови, інакше солвер не зможе знайти розв'язок.

Приклад 1. Розв'язати систему ДАР.

$$\begin{cases} \dfrac{du}{dt} - v = 0 \ , u(0) = 0.5 \\ u^2 + v^2 = 1 \end{cases} \qquad (1)$$

Зауваження. При початкових значеннях $u(0)=1, v(0)=0$ існують дві пари розв'язків $u \equiv 1, v \equiv 0$ і $u = \cos t, v = \sin t$. При $u(0)=0, v(0)=1$ розв'язками є $u \equiv 0, v \equiv 1$ і $u = \sin t, v = \cos t$ (якщо в початкових умовах замість $+1$ стоїть -1, то мінус з'являється в розв'язку у відповідних величин). При інших значеннях розв'язок має вигляд $u = a\cos t + b\sin t$, $v = b\cos t - a\sin t$, де коефіцієнти a, b підлягають пошуку з урахуванням початкового значення для функції u і початкового наближення для функції v. Будь які узгоджені неперервні комбінації ділянок попередніх функцій також можливі.

Розв'язання. Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File –> New –> Model Wizard –> 0D –> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) –> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Заповніть таблицю рівнянь.

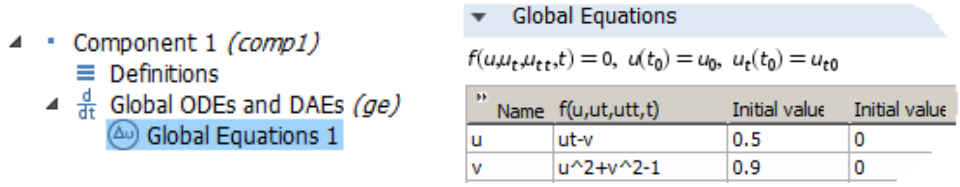
Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `u, ut-v`

– **Initial value(u_0)** = `0.5`

`Global Equations` – **Name, f(u, ut, utt, t)** = `v, u^2+v^2-1`

– **Initial value(u_0)** = `0.9`

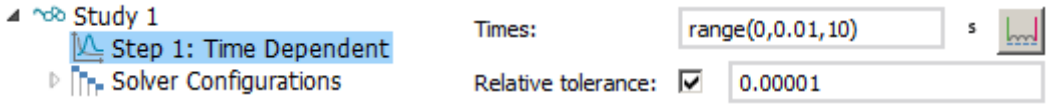


Задайте діапазон зміни параметра `t` і відносну похибку.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.01,10)`

– **Relative tolerance** = `True, 0.00001`



В цій задачі бажано відразу зменшити абсолютну похибку. Необхідний для її введення вузол в дереві моделі спочатку не відображається. Тому виконайте команду

`Study` -> Solver - Show Default Solver.

В гілці **Study 1** з'явиться кілька дочірніх вузлів, в одному з яких можна буде зменшити абсолютну похибку.

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

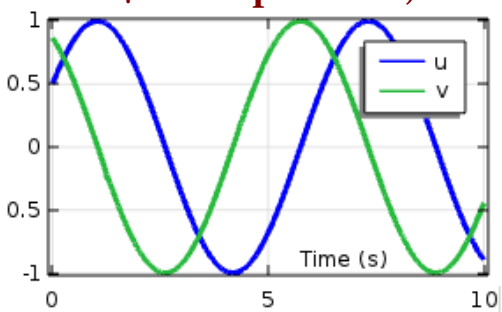
`Absolute Tolerance` – **Tolerance** = `0.00001`

Виконайте команду Compute і в графічній зоні вузла **1D Plot Group 1–Global 1** отримайте криві розв'язку. Вони наведені на наступному рисунку. Щоб скорегувати легенду ми лише змінили текст в полях **Description** таблиці виразів.

1D Plot Group 1 - Global 1:

`y-Axis Data` – `таблиця` - **Expression, Description** = `comp1.u, u`

– `таблиця` - **Expression, Description** = `comp1.v, v`

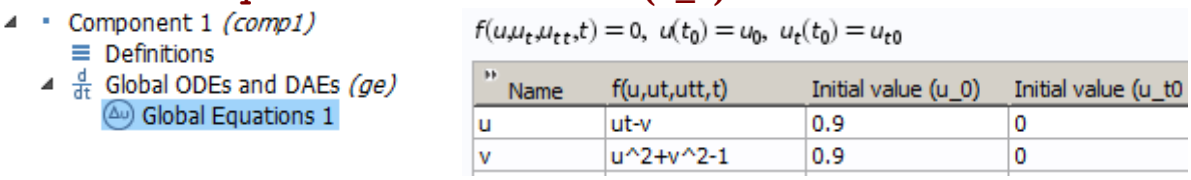


Однак для багатьох пар початкових значень чисельний розв'язок отримати не вдається, наприклад, для $u(0)=0.9, v(0)=0.9$. Щоб це перевірити змініть початкові значення в таблиці рівнянь

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – Initial value(u_0) = `0.9`

`Global Equations` – Initial value(u_0) = `0.9`



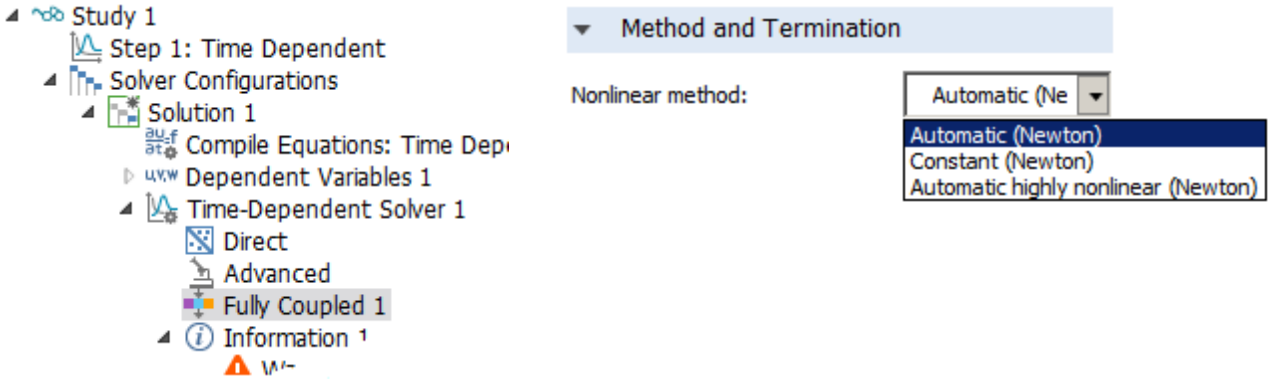
Виконайте команду Compute і ви отримаєте повідомлення

Воно означає, що системі під час пошуку розв'язку не вдалося підібрати узгоджені початкові значення (початкові значення називаються узгодженими, якщо вони задовольняють алгебраїчним рівнянням). Часто в цьому «винен» метод, який використовується для їх підбору. В такому випадку, радимо змінити метод розв'язку `Constant(Newton)` на інший.

Study 1 – Solver Configurations – Solution 1 –

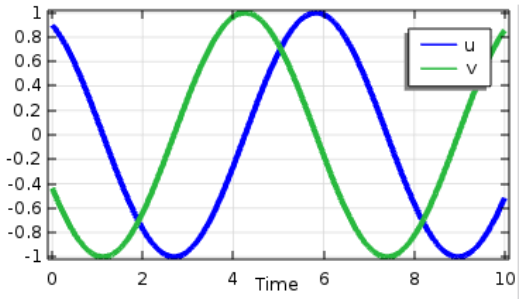
– Time Dependent Solver 1 – Fully Coupled 1:

`Method and Termination` – Nonlinear Method = `Automatic(Newton)`



Цей вибір при визначенні коренів алгебраїчних рівнянь змушує систему користуватися методом Ньютона з загасанням (damped Newton's method). Він добре працює в тих випадках, коли звичайний метод Ньютона «зависає», і полягає в тому, що на деяких кроках ітерації похідна множиться на числовий коефіцієнт $0<\alpha<1$.

Виконайте команду Compute і в графічній зоні вузла **Results - 1D Plot Group 1 - Global 1** отримайте наступні криві.



Зверніть увагу на значення $v(0)$, надрукувавши таблицю величин `u` і `v`.

Results - Derived Values - State variable u: LMB > Evaluate

Results - Derived Values - State variable v: LMB > Evaluate –Table 1

Time	State variable u	State variable v
0.0000	0.90000	-0.43589
0.010000	0.89558	-0.44489
0.020000	0.89106	-0.45387
0.030000	0.88646	-0.46[illegible]
0.040000	0.88178	

Маємо $v(0) \approx -0.4359$. Воно не співпадає зі значенням 0.9, яке ми ввели в таблиці рівнянь. Таким чином, в нашому прикладі початкове значення $u(0) = 0.9$ для функції `u` не змінилося (бо перше рівняння системи є диференціальним), а підбиралося початкове значення для функції `v`.

Спробуйте використати інші початкові наближення. Наприклад, для значень $u(0) = 1,\ v(0) = 0$, сумісних з другим рівнянням системи, розв'язком будуть функції $u(t) \equiv 1, v(t) \equiv 0$. Для початкових наближень $u(0) = 0,\ v(0) = 1$ розв'язком будуть функції $u(t) \equiv \sin t, v(t) \equiv \cos t$ (точніше, ви отримаєте графіки цих функцій).

Приклади показують, що система не намагається підбирати початкове значення для функції `u` і, якщо його задати зовсім «погано», наприклад $u(0) = 2$, то розв'язок не буде знайдено. Справді, не існує дійсного значення $v(0)$, яке задовольняло б алгебраїчному рівнянню $2^2 + v^2(0) - 1 = 0$.

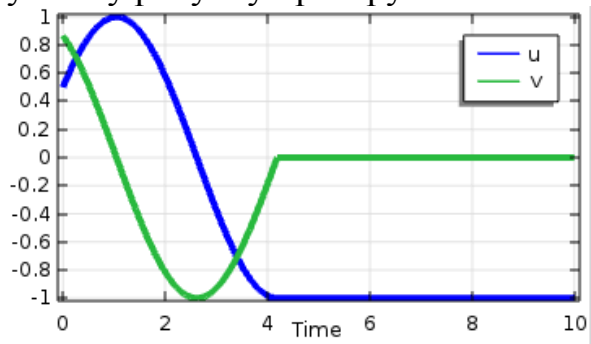
Змініть ще раз початкові значення $u(0) = 0.5,\ v(0) = 0.99$ і розв'яжіть систему. В результаті будуть побудовані графіки розв'язку, які показано на наступному рисунку ліворуч. В момент часу приблизно `t=4.19` відбувається стрибок з однієї пари розв'язків на іншу. Це викликано недостатньою точністю обчислень

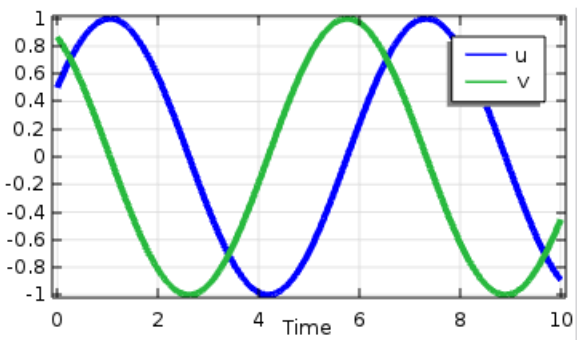
Step 1: Time Dependent:

`Study Settings` – **Relative tolerance** = `True, 0.000001`

`Home` **->** Compute (наступний рисунок праворуч)

Зменшивши відносну похибку, ви отримаєте графіки, які показані на наступному рисунку праворуч.





Іноді мала похибка може справити негативний вплив. Спробуйте розв'язати нашу диференціально-алгебраїчну систему методом `Constant(Newton)` з початковими значеннями $u(0) = 1,\ v(0) = 1$ і з відносною і абсолютною похибками `0.00001`.

Step 1: Time Dependent:

Study Settings – **Relative tolerance** = True, 0.00001

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

Absolute Tolerance – **Tolerance** = 0.00001

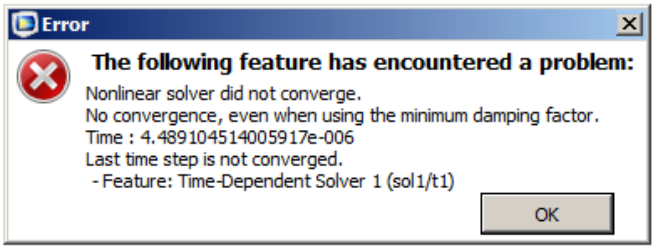
Study 1 – Solver Configurations – Solution 1 –

– Time Dependent Solver 1 – Fully Coupled 1:

Method and Termination – **Nonlinear Method** =Constant (Newton))

Home -> Compute

Ви отримаєте наступне повідомлення.

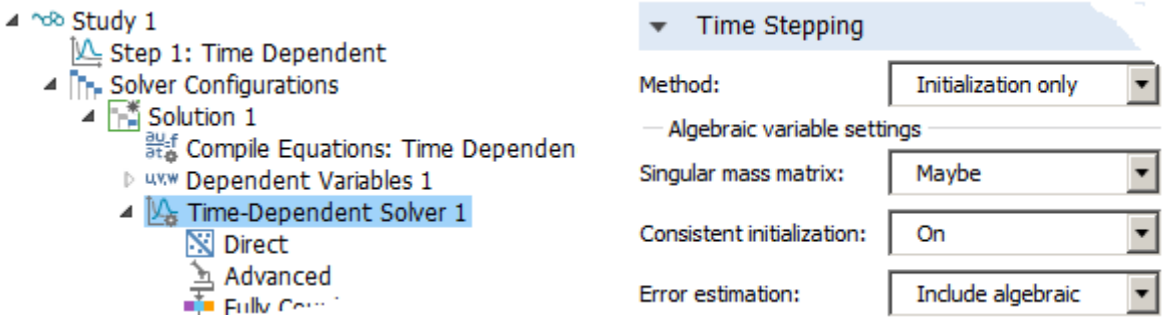


Але якщо ви знизите абсолютну похибку до значення 0.01, то отримаєте правильну відповідь: $u \equiv 1, v \equiv 0$ (насправді ви отримаєте графіки розв'язків, які відповідають цим функціям). При абсолютній похибці 0.001 ви отримаєте графіки косинуса і мінус синуса (теж правильно!). Отже для «особливого початкового значення» u=1 можна отримати майже «випадково» одну чи іншу пару розв'язків.

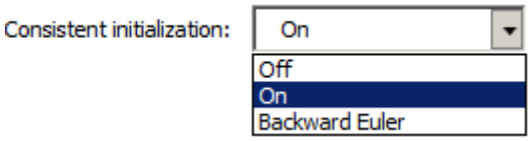
Повернемо в таблиці рівнянь початкові значення $u(0) = 0.9,\ v(0) = 0.9$ і наведемо ще один шлях пошуку розв'язку системи ДАР. В програмі є розв'язувач, призначений лише для підбору початкових значень систем ДАР. Виберіть його.

Study 1 – Solver Configurations – Solution 1 – Time Dependent Solver 1:

Time Stepping – **Method** = Initialization only



При цьому в списку **Consistent initialization** виберіть варіант On, який означає, що початкові значення неузгоджені (вибір Off означає, що початкові значення вже сумісні і солвер не намагатиметься їх підбирати).



Вибір зі списку **Singular mass matrix** значення Maybe налаштовує солвер на автоматичне визначення того, чи включає система алгебраїчні рівняння.

Значення `Yes` стверджуватиме, що система містить алгебраїчні рівняння. Але і в цьому випадку (для цієї задачі) потрібно вибирати метод Ньютона з загасанням.

Study 1 – Solver Configurations – Solution 1 –

– Time Dependent Solver 1 – Fully Coupled 1:

`Method and Termination` – Nonlinear Method = `Automatic(Newton)`

`Home` **->** Compute

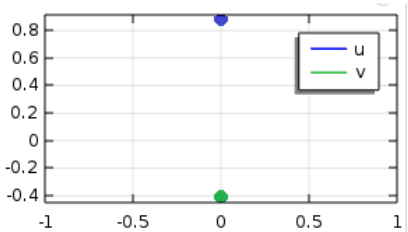
Результатом є набір узгоджених початкових значень для шуканих функцій. При цьому визначення розв'язків системи для інших моментів часу не виконується. Якщо обчислення закінчилися без повідомлення про помилку, то в гілці **Results – Derived Values** вузли **State variable u** і **State variable v** будуть містити ці узгоджені початкові значення. Роздрукуйте їх.

Derived Values - State variable u: LMB > Evaluate – New Table

Derived Values - State variable v: LMB > Evaluate –Table 2

Time (s)	State variable u (1)	State variable v (1)
0.0000	0.90000	-0.43589

Ви також можете спостерігати розв'язок у вигляді двох точок в графічній зоні вузла **1D Plot Group 1**.



Тепер, ви можете змінити початкові значення, повернути вихідний чисельний метод розв'язання (зазвичай це BDF метод) і заново обчислити задачу.

BDF (Backward Differentiation Formulas) солвер використовує різні модифікації зворотних методів диференціювання (від першого до п'ятого порядків).

В таблиці рівнянь задайте початкові значення $u(0) = 0.9$, $v(0) = -0.4$, близькі до тих, які отримані в процесі «узгодження».

Global Equations

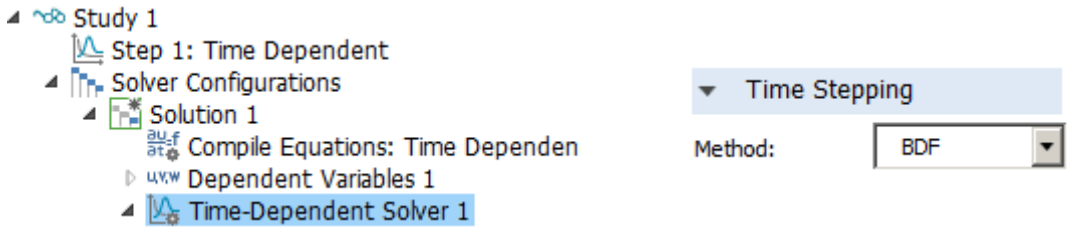
$f(u,u_t,u_{tt},t) = 0,\ u(t_0) = u_0,\ u_t(t_0) = u_{t0}$

Name	f(u,ut,utt,t)	Initial value	Initial value
u	ut-v	0.9	0
v	u^2+v^2-1	-0.4	0

Оберіть BDF метод і виконайте обчислення.

Study 1 – Solver Configurations – Solution 1 – Time Dependent Solver 1:

`Time Stepping` – Method = `BDF`



`Home` **->** Compute

Іноді «спрацьовує» ще один спосіб. Створіть нову модель так само, як і раніше в цьому прикладі. Задайте відносну і абсолютну похибки `0.00001`, і в таблиці рівнянь використайте початкові значення $u(0) = 0.1$, $v(0) = 0.9$. Обчислення моделі покаже, що початкові значення неузгоджені. Спробуйте змінити метод масштабування `Initial value based`.

Study 1 - Solver Configurations - Solution 1 - Dependent Variables 1:

`Scaling` – **Method** = `Initial value based`

Знову виконайте обчислення, але на цей раз ви отримаєте розв'язок.

■

Розв'яжемо ще кілька диференціально - алгебраїчних систем рівнянь.

Коли алгебраїчні рівняння лінійні, то узгоджені початкові умови підбираються відносно просто.

Приклад 2. Розв'язати систему ДАР

$$\begin{cases} x' + x + 2y = 0 \ , \ x(0) = -1 \\ y' - x - z = 0 \ , \ y(0) = 3 \ . \\ x + y + z = 0 \ , \ z(0) = -1 \end{cases}$$

Тут, очевидно, що $x(0) + y(0) + z(0) \neq 0$, тобто початкові дані неузгоджені.

Р о з в ' я з а н н я . За допомогою майстра створіть 0D модель задачі.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі спостереження за розмірностями фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

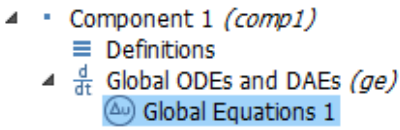
Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `u, ut+u+2*v`
– **Initial value(u_0)** = `-1`

`Global Equations` – **Name, f(u, ut, utt, t)** = `v, vt-u-w`
– **Initial value(u_0)** = `3`

`Global Equations` – **Name, f(u, ut, utt, t)** = `w, u+v+w`
– **Initial value(u_0)** = `-1`



Global Equations

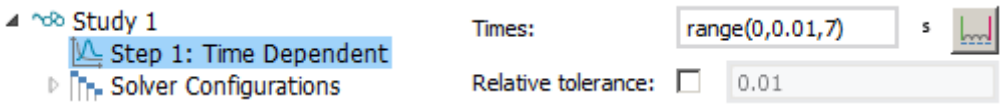
$f(u,u_t,u_{tt},t) = 0,\ u(t_0) = u_0,\ u_t(t_0) = u_{t0}$

Name	f(u,ut,utt,t)	Initial value	Initial value
u	ut+u+2*v	-1	0
v	vt-u-w	3	0
w	u+v+w	-1	0

Задайте діапазон зміни параметра `t`.

Step 1: Time Dependent:

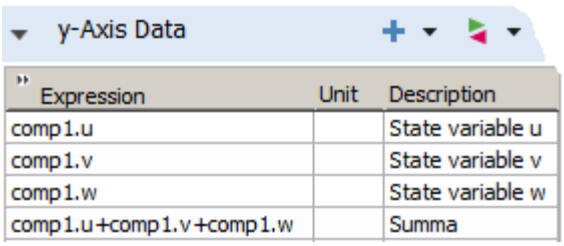
Study Settings – **Times** = `range(0,0.01,7)`



Виконайте команду Compute і в графічній зоні вузла **Results - 1D Plot Group 1 - Global 1** отримайте графіки розв'язків. Для контролю в ПВ цього вузла в розділі **y-Axis Data** додайте вираз, який представляє суму шуканих функцій. Відповідно до третього рівнянням системи, його графік повинен бути тотожним нулем.

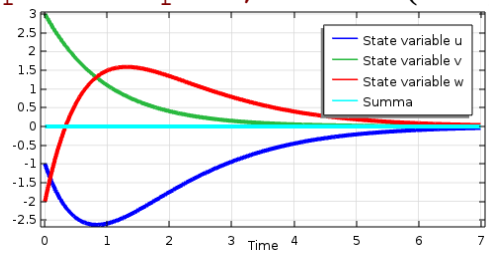
1D Plot Group 1 - Global 1:

y-Axis Data – таблиця – Expression, Description = `comp1.u+comp1.v+comp1.w, Summa` (row4)



y-Axis Data

Expression	Unit	Description
comp1.u		State variable u
comp1.v		State variable v
comp1.w		State variable w
comp1.u+comp1.v+comp1.w		Summa



Зверніть увагу, що початкові значення функцій `u` і `v` солвер не міняв. Для узгодження з алгебраїчним рівнянням було підібрано тільки стартове значення $w(0)$.

Приклад 3. Аналогічно до попереднього прикладу досліджується система

$$\begin{cases} x' + x + y = 0 \ , x(0) = 1 \\ y - x + 2z = 2 \ , \ y(0) = -1. \\ x + 2z = 1 \qquad , \ z(0) = 1 \end{cases}$$

Розв'язання. За допомогою майстра створіть 0D модель задачі.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі спостереження за розмірностями фізичних величин.

Untitled.mph (root): Unit System – <> = None

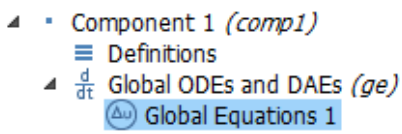
Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name, f(u, ut, utt, t)** = `u, ut+u+v`
– **Initial value(u_0)** = `1`

Global Equations – **Name, f(u, ut, utt, t)** = `v, v-u+2*w-2`
– **Initial value(u_0)** = `-1`

Global Equations – **Name, f(u, ut, utt, t)** = `w, u+2*w-1`
– **Initial value(u_0)** = `1`



Global Equations

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

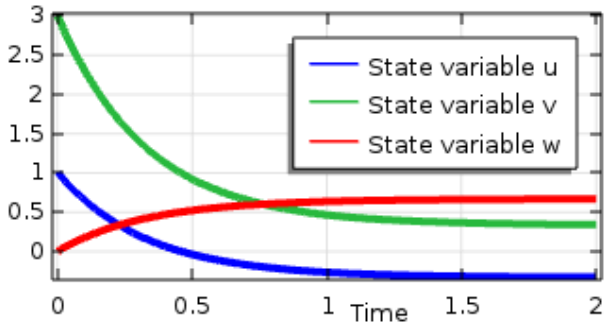
Name	f(u,ut,utt,t)	Initial value
u	ut+u+v	1
v	v-u+2*w-2	-1
w	u+2*w-1	1

Задайте діапазон зміни параметра `t`.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.01,2)`

Виконайте команду Compute і в графічній зоні вузла **Results - 1D Plot Group 1 - Global 1** отримайте графіки розв'язків.



Оскільки функції `v` і `w` входять в систему без диференціювання, то солвер дозволив собі підбирати їх початкові значення і, як бачите, вони змінилися.

Приклад 4. Розв'язати систему рівнянь

$$\begin{cases} x'' - \dfrac{1}{4}y = 0 & , x(0) = -1 \\ x + y - \cos t = 0, & y(0) = 0 \end{cases}.$$

Розв'язання. За допомогою майстра створіть 0D модель задачі.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі спостереження за розмірностями фізичних величин.

Untitled.mph (root): **`Unit System`** – <> = None

Заповніть таблицю рівнянь.

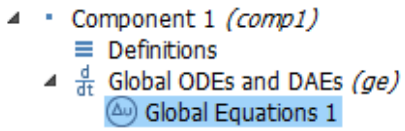
Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `u, utt-v/4`

– **Initial value(u_0), Initial value(u_t0)** =`-1,0`

`Global Equations` – **Name, f(u, ut, utt, t)** = `v, u+v-cos(t)`

– **Initial value(u_0)** = `0`



Global Equations

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
u	utt-v/4	-1	0
v	u+v-cos(t)	0	0

Задайте діапазон зміни параметра `t`.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.01,8*pi)`

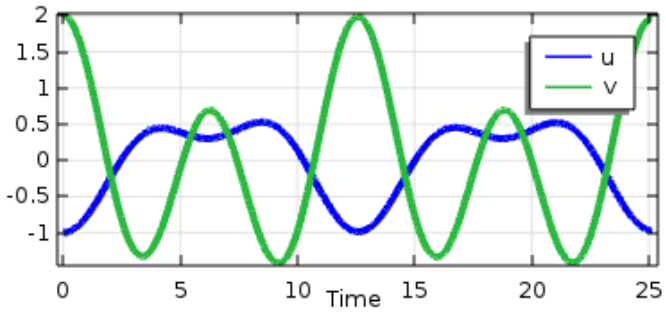
Виконайте команду Compute і в графічній зоні вузла **1D Plot Group 1–Global 1** отримайте графіки розв'язку. Для корегування легенди змініть текст в полях **Description** таблиці виразів.

1D Plot Group 1 - Global 1:

y-Axis Data – таблиця - Expression, Description = comp1.u, u

– таблиця - Expression, Description = comp1.v, v

LMB > Plot



Оскільки в першому рівнянні системи міститься друга похідна шуканої функції u, то для неї використовуються обидві початкові умови $u(0)=-1, u'(0)=0$, а початкове значення для функції v підбирається розв'язувачем.

Приклад 5. Розв'язати систему рівнянь

$$\begin{cases} x' - 2y - xy = 0, \; x(0) = -1.5 \\ \dfrac{x^2}{4} + y^2 = 1 \quad , \; y(0) = 0 \end{cases}.$$

Розв'язання. За допомогою майстра створіть нульвимірну динамічну модель задачі.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі спостереження за розмірностями фізичних величин.

Untitled.mph (root): Unit System – <> = None

Заповніть таблицю рівнянь.

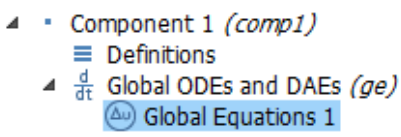
Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – Name, f(u, ut, utt, t) = u, ut-2*v-u*v

– Initial value(u_0) =-1.5

Global Equations – Name, f(u, ut, utt, t) = v, u^2/4+v^2-1

– Initial value(u_0) = 1



$f(u,u_t,u_{tt},t)=0,\; u(t_0)=u_0,\; u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)
u	ut-2*v-u*v	-1.5
v	u^2/4+v^2-1	1

Задайте діапазон зміни параметра t, а також відносну і абсолютну похибку обчислень 0.0001.

Step 1: Time Dependent:

Study Settings – Times = range(0,0.01,10)

– Relative tolerance = True, 0.0001

`Study` –> Solver - Show Default Solver.

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

`Absolute Tolerance` – **Tolerance** = `0.0001`

Оберіть метод Ньютона з загасанням.

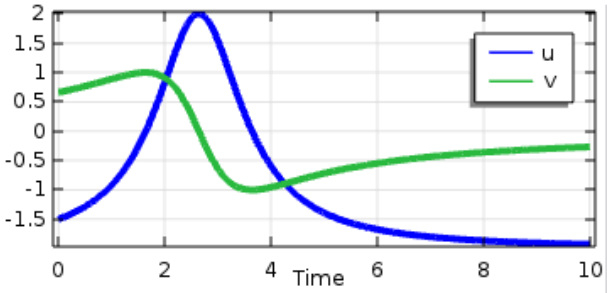
Study 1 – Solver Configurations – Solution 1 –

– Time Dependent Solver 1 – Fully Coupled 1:

`Method and Termination` – **Nonlinear Method** = `Automatic(Newton)`

LMB > Compute

В графічній зоні вузла **1D Plot Group 1-Global 1** отримайте графіки розв'язків.



Тепер побудуйте параметричну криву $u = u(t), v = v(t)$. Для цього створіть і налаштуйте графічний вузол **1D Plot Group 2 - Global 1**.

`Results` –> Plot Group - 1D Plot Group

`1D Plot Group 2` –> Add Plot – Global

1D Plot Group 2 :

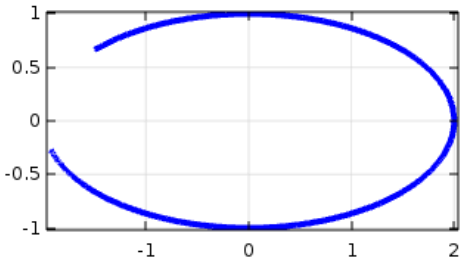
`Legend` – **Show legends** = `False`

1D Plot Group 2 - Global 1:

`y-Axis Data` – **Expression** = `comp1.v`

`x-Axis Data` – **Parameter, Expression** = `Expression, comp1.u`

LMB > Plot



Як і слід було очікувати, точки кривої $u = u(t), v = v(t)$ розташовані на еліпсі $\frac{u^2}{4} + v^2 = 1$.

4.1.4. Використання в рівняннях операторів.

В таблиці рівнянь ПВ вузла `Global ODEs and DAEs(ge)` вирази в стовпці **`f(u,ut,utt,t)`** можуть містити не тільки арифметичні операції, функції, параметри і змінні моделі, а і оператори системи або користувача. Частіше за все використовується оператор `integrate(...)`. Але зустрічаються випадки застосування інших операторів. Розв'яжемо кілька навчальних задач, рівняння яких містять оператори.

Приклад 1. Відносно невідомої X (або X і Y) розв'язати наступні рівняння:

1. $f'(X) = 0$, де $f(x)$ відома функція. Розшукується точка екстремуму.
2. $f''(X) = 0$, де $f(x)$ відома функція. Розшукується точка перегину.
3. $\int_0^X f(x)dx = a$, де $f(x)$ відома функція, і a задане число. Розшукується відрізок [0,X], на якому площа під кривою є заданим числом.
4. $\int_X^Y f(x)dx = a$, $Y - X = b$, де $f(x)$ відома функція, a, b задані числа. Розшукується відрізок [X,Y] довжиною b, на якому площа під кривою є заданим числом a.
5. $\int_0^X g(x, X)d\,x = a$, де $g(x, y) = e^{xy}$.

Розв'язання. Створіть модель задачі зі стаціонарним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Створіть функцію $f(x) = 2 + 12x + 3x^2 - 2x^3$.

Global Definitions: RMB > Functions – Analytic

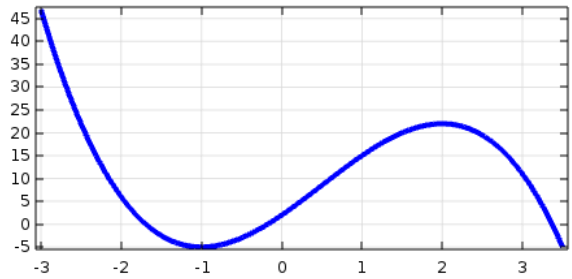
Function name = `f`

`Definition` – **Expression** =`2+12*x+3*x^2-2*x^3`

Для перевірки виконайте побудову графіка цієї функції на відрізку [-3, 3.5].

`Plot Parameters` – **Low limit, Upper limit** = -3, 3.5

LMB> Plot



Дивлячись на графік, можна приблизно оцінити положення точки максимуму і точки перегину.

Задайте перше рівняння і початкове наближення до точки максимуму.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u,ut,utt,t)** = `u, d(f(u),u)`
– **Initial value(u_0)** = `2.5`

Запишіть друге рівняння і початкове наближення до точки перегину.

`Global Equations` – **Name** = `v`
– **f(u,ut,utt,t)** = `d(d(f(v),v),v)`
– **Initial value(u_0)** = `0`

Виконайте обчислення.

Home -> Compute

В гілці **Results - Derived Values** з'являються два вузли **State variable u** і **State variable v**. Надрукуйте значення знайдених величин `u`, `v` в спільну таблицю.

Derived Values – State variable u: LMB > Evaluate
Derived Values – State variable v: LMB > Evaluate - Table 1

State variable u	State variable v
2.0000	0.50000

Це є абсцисами точки правого екстремуму (максимуму) і точки перегину.

Перейдемо до розв'язання третьої і четвертої задачі. Спочатку створіть параметри моделі.

Global Definitions : RMB > Parameters

Parameters – **Name, Expression** = a, 10
– **Name, Expression** = b, 1

Додайте третій вираз в таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name** = w
Global Equations – **f(u,ut,utt,t)** = a-integrate(f(x),x,0,w)
– **Initial value(u_0)** = 1

Зауважимо, що для коректної роботи метода Ньютона-Рафсона початкове наближення w потрібно обирати так, щоб нахил функції (дотичної в цій точці) не дорівнював нулю. В цьому прикладі це не повинні бути корені підінтегральної функції.

Home -> Compute

Після розв'язання задачі надрукуйте w.

Derived Values : RMB > Global Evaluation
Label = Variable w

Derived Values – Variable w: LMB > Evaluate - Table 1

State variable u	State variable v	State variable w
2.0000	0.50000	1.0963

Тобто $\int_0^{1.0963} f(x)dx = 10$.

Розв'яжіть четверте завдання.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name** = X
– **f(u,ut,utt,t)** = a-integrate(f(x),x,X,Y)
– **Initial value (u_0)** = 1

Global Equations – **Name** = Y
– **f(u,ut,utt,t)**= Y-X-b
– **Initial value (u_0)** = 2

Home -> Compute

Надрукуйте розв'язок.

Derived Values – Variable w: RMB > Duplicate
Label = Variable X

Derived Values – Variable X:

Expression –**Expression** = comp1.X

LMB > Evaluate – New Table

Derived Values – Variable X: RMB > Duplicate

Label = `Variable Y`

Derived Values – Variable Y:

Expression –**Expression** = `comp1.Y`

LMB > Evaluate – Table 2

State variable X	State variable Y
0.11562	1.1156

Таким чином `X=0.11562`, `Y=1.1156`.

Зауваження. Знайти верхню межу Y відрізка інтегрування можна було б за допомогою одного рівняння `a-integrate(f(x),x,Y-b,Y)=0`.

Для розв'язання п'ятого рівняння $\int_0^X g(x,X)\,d\,x = a$ спочатку створіть функцію $g(x,y)$.

Global Definitions: RMB > Functions – Analytic

Function name = `g`

Definition – **Expression** = `exp(x*y)`

– **Arguments** = `x,y`

Тепер запишіть рівняння в шостий рядок таблиці.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name** = `Z`

– **f(u,ut,utt,t)(1)** = `integrate(g(x,Z),x,0,Z)-a`

– **Initial value (u_0)** = `1`

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
u	d(f(u),u)	2.5	0
v	d(d(f(v),v),v)	0	0
w	a-integrate(f(x),x,0,w)	0	0
X	a-integrate(f(x),x,X,Y)	1	0
Y	Y-X-b	2	0
Z	integrate(g(x,Z),x,0,Z)-a	1	0

Знову обчисліть модель і надрукуйте розв'язок `Z`.

`Home` **->** Compute

Derived Values – Variable Y: RMB > Duplicate

Label = `Variable Z`

Derived Values – Variable Z:

Expression –**Expression** = `comp1.Z`

LMB > Evaluate – New Table

State variable Z
1.7015

■

Підбір апроксимуючої кривої. Коли задано багато експериментальних точок, в значеннях яких є похибка, не має сенсу шукати функцію, графік якої точно проходить крізь ці точки. Часто бажано знайти параметри функції (наприклад, коефіцієнти полінома), яка несуттєво відхиляється від множини точок.

Розглянемо задачу апроксимування одновимірного набору точок кубічним поліномом $P = c_0x^3 + c_1x^2 + c_2x + c_3$, який найкращим чином апроксимує

дані. Термін «найкращим чином» потребує уточнення. Вважатимемо, що через експериментальні точки $(x_i, y_i)_{i=1}^n$, абсциси яких розташовані на відрізку $[a, b]$, проходить ламана $L(x)$, і найкращим поліномом є той, який мінімізує інтеграл

$$R(\mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_2, \mathrm{c}_3) = \int_a^b (L(x) - P(\mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_2, \mathrm{c}_3, x))^2 dx \qquad (1)$$

В точці мінімуму виразу $R(\mathrm{c}_0, \mathrm{c}_1, \mathrm{c}_2, \mathrm{c}_3)$ похідні R по коефіцієнтам c_i повинні дорівнювати нулю. Тому будуть виконуватися співвідношення:

$$\frac{\partial R}{\partial c_0} = \frac{\partial R}{\partial c_1} = \frac{\partial R}{\partial c_2} = \frac{\partial R}{\partial c_3} = 0 \qquad (2)$$

Розв'язавши систему рівнянь (2) відносно коефіцієнтів c_i, ми отримаємо шуканий поліном $P = \mathrm{c}_0 x^3 + \mathrm{c}_1 x^2 + \mathrm{c}_2 x + \mathrm{c}_3$.

Приклад 2. Побудувати кубічний поліном, «найкращим чином» апроксимуючий наступні дані.

0	0.03
0.1	0.06
0.2	0.09
0.3	0.15
0.4	0.1
0.5	0.3
0.6	0.4
0.7	0.4
0.8	0.5
0.9	0.9
1.0	1.1

Будемо вважати, що ці дані знаходяться в текстовому файл `InterpData_01.txt`, в якому лівий стовпець містить абсциси входових точок, а правий – їх ординати (дані в рядках розділені проміжком). Ви можете створити такий файл в блокноті і зберегти з відповідним ім'ям.

План розв'язання цієї задачі в COMSOL буде наступним.

1. Створіть одновимірну модель, на відрізку [0,1] якої розташовані абсциси x_i входових даних;
2. Створіть інтерполяційну функцію $L(x)$ (ламану);
3. Створіть оператор інтегрування по відрізку [0,1];
4. Створіть змінну моделі P, яка буде залежати від координати x і чотирьох поки що невідомих параметрів c_i. По формулі (1) створіть змінну R.
5. В таблиці рівнянь задайте систему (2), після розв'язання якої можна буде накреслити графік полінома P.

Розв'язання. Створіть одновимірну стаціонарну модель з Global ODEs and DAEs інтерфейсом.

File –> New –> Model Wizard –> 1D –> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) –> Stationary

Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

В геометрії моделі створіть відрізок [0,1] і побудуйте на ньому точки x=0,0.1,0.2,...,0.9,1.0.

Geometry **-> Primitives->** Interval

Geometry **-> Primitives** – Point

Component 1 - Geometry 1 - Point 1:

Point – **x** = 0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0

LMB > Build All Objects

По експериментальним даним створіть інтерполяційну ламану. Для цього по файлу даних InterpData_01.txt створіть Interpolation функцію, якій дайте ім'я *L*. Активуйте прапорець Use spatial coordinates as arguments (використовувати просторові координати в якості аргументів) і переконайтеся, що обрано Linear спосіб інтерполяції.

Component 1 - Definitions: RMB > Functions – Interpolation

Definition – **Data source** = File

Натиснувши кнопку Browse, виберіть файл даних.

– **Filename:>** Browse (*натискання кнопки*)

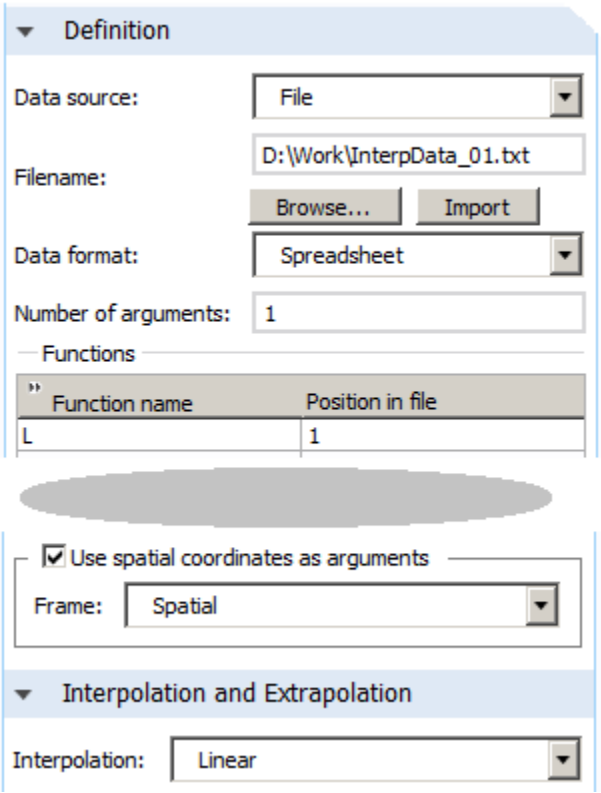
– **Filename:** = D:\Work\InterpData_01.txt

– **Functions(таблиця) - Function name** = L

– **Use spatial coordinates as arguments** = True

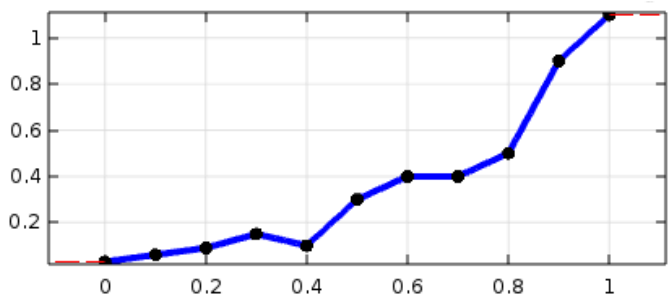
– **Frame** = Spatial

Interpolation and Extrapolation – **Interpolation** = Linear

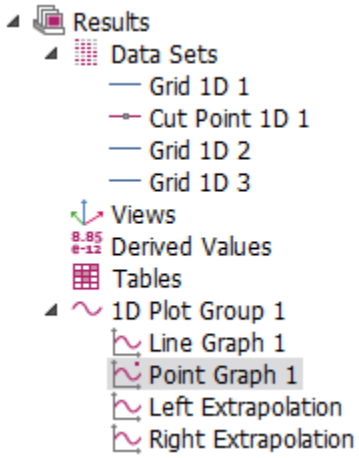


Для перевірки виконайте побудову графіка цієї функції

LMB> Create Plot

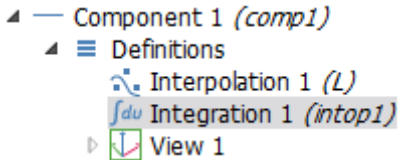


Для навчальних цілей проаналізуйте набір джерел даних, а також вузлів в графічній групі **1D Plot Group 1**, які були автоматично створені при виконанні останньої команди.



Створіть оператор інтегрування по відрізку [0,1].

Component 1 :> `Definitions` > Component Couplings – Integration



Component 1 - Definitions - Integration 1(intop1):

Оскільки відрізок поділено на 10 інтервалів, то в оператор інтегрування потрібно включити їх всі.

`Source Selection` – **Selection** = `All domains`

Створіть змінні моделі *P* і *R*.

Component 1(comp1) – Definitions: RMB > Variables

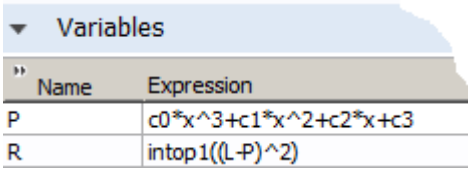
Definitions – Variables 1:

`Geometric Entity Selection` – **Geometric entity level** = `Entire model`

`Variables` – **Name, Expression** = `P, c0*x^3+c1*x^2+c2*x+c3`

Значення коефіцієнтів c_i визначимо пізніше.

`Variables` – **Name, Expression** = `R, intop1((L-P)^2)`



Variables

Name	Expression
P	c0*x^3+c1*x^2+c2*x+c3
R	intop1((L-P)^2)

Зверніть увагу на те, що змінні визначені на всій моделі

У вузлі `Global Equations` задайте чотири рівняння (2), яким повинні задовольняти коефіцієнти c_i.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `c0, d(R,c0)`
– **Name, f(u, ut, utt, t)** = `c1, d(R,c1)`
– **Name, f(u, ut, utt, t)** = `c2, d(R,c2)`
– **Name, f(u, ut, utt, t)** = `c3, d(R,c3)`

Нарешті, ми повинні накласти СкЕ сітку на нашу одновимірну розрахункову область. Для того, щоб сітка мала вузли в експериментальних точках, ми на відрізку [0,1] побудували геометричні точки. Для розв'язання задачі нам достатньо між кожною парою сусідніх точок мати лише один скінченний елемент (бо інтерполяційна крива – ламана).

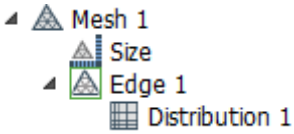
Component 1 (comp1) - Mesh 1: RMB > Edge (*сітка на ребрах*)

Component 1 (comp1) - Mesh 1 – Edge 1: RMB > Distribution

Component 1 (comp1) - Mesh 1 – Edge 1 - Distribution 1:

`Domain Selection` – **Selection** = `All domains`

`Distribution` – **Number of elements** = `1` (*на кожному відрізку один скінченний елемент*)



Тепер можна розв'язати задачу знаходження чисельних значень коефіцієнтів c_i, і побудувати результат.

`Home` -> Compute

В гілці **Derived Values** будуть створені вузли типу `Global Evaluation` для обчислення величин `c0`, `c1`, `c2`, `c3`. Надрукуйте їх значення в спільну таблицю.

Derived Values – State variable c0: LMB > Evaluate

Derived Values – State variable c1: LMB > Evaluate - Table 1

Derived Values – State variable c2: LMB > Evaluate - Table 1

Derived Values – State variable c3: LMB > Evaluate - Table 1

State variable c0	State variable c1	State variable c2	State variable c3
1.7665	-1.3332	0.68976	0.0043989

Побудуйте графіки інтерполяційної ламаної L і полінома P.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 2` -> Add Plot – Line Graph

1D Plot Group 2 - Line Graph 1:

`Selection` – **Selection** = `All domains`

`y-Axis Data` – **Expression** = `P`

`x-Axis Data` – **Parameter, Expression** = `Expression,x`

`Coloring and Style` –**Width** = `3`

1D Plot Group 2 - Line Graph 1: RMB>Duplicate

1D Plot Group 2 - Line Graph 2:

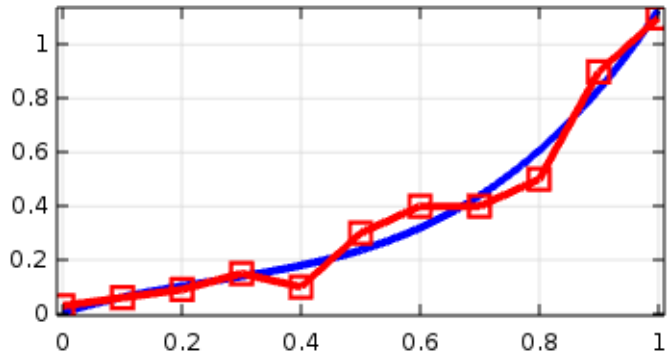
`y-Axis Data` – **Expression** = `L`

`Coloring and Style` – **Color** = `Red`

Coloring and Style – **Line markers** – **Marker** = Square
– **Line markers** – **Positioning** = In data points
Quality – **Resolution** = Custom

Останні три опції додатково налаштовують графічну систему відображати змінну L у вузлах СкЕ сітки у вигляді квадратів (тобто відображаються вузли ламаної).

LMB > Plot



На графіку побудовано набір початкових точок, лінійну інтерполяцію між ними і підібрану функцію. Ми мінімізували інтеграл від квадрата різниці між цими кривими і отримали гладку і просту функцію, яка досить добре апроксимує наші дані.

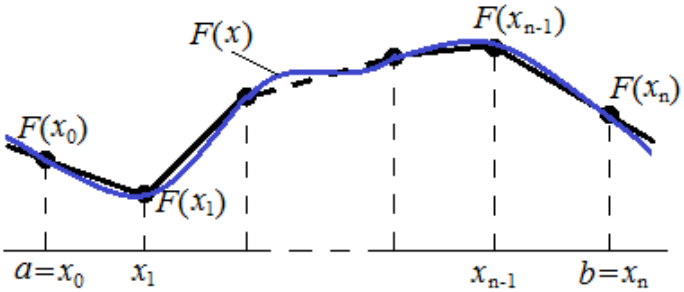
Зауваження. Вузли ламаної можна відобразити за допомогою точкового графіка (Point Graph) так, як це зроблено в гілці **1D Plot Group 1**. ■

Апроксимація функції ламаною. Знайти рівняння ламаної $L(x)$, яка на відрізку $[a,b]$ співпадає з функцією $F(x)$ в n+1 точках $a = x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n = b$, включно з кінцями $a = x_0$, $b = x_n$.

Відомо, що рівняння будь-якої ламаної можна записати у вигляді [3]

$$L(x) = a_0 + a_1 x + \sum_{i=1}^{n-1} c_i \left| x - x_i \right|, \tag{3}$$

де x_i - абсциси вершин ламаної. В нашій задачі треба враховувати, що точки зламу x_i $(i = 1, 2, \ldots, n-1)$ є внутрішніми точками відрізка $[a,b]$, а в кінцевих точках $a = x_0$, $b = x_n$ зламів немає.



Коефіцієнти $a_0, a_1, c_1, \ldots, c_{n-1}$ визначатимуться з n+1 рівняння

$$L(x_i) = F(x_i),\ i = 0, 1, \ldots, n-1, n. \tag{4}$$

Зауваження. Ми шукатимемо саме рівняння (3) ламаної (спочатку з невідомими коефіцієнтами), яке в інших задачах можна буде використовувати у вузлах

Variables, а не Interpolation функцію-ламану, яка повинна з самого початку мати координати всіх вузлів визначеними.

Приклад 3. Знайти рівняння ламаної, яка на відрізку [0,1] наближає функцію $f(x)=e^{3x}\cdot\sin\pi x$. Використати 4 внутрішніх вузли.

Розв'язання. Створіть одновимірну стаціонарну модель з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 1D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Створіть апроксимовувану функцію $f(x)$.

Global Definitions: RMB > Functions – Analytic

Function name = F

Definition – **Expression** = sin(pi*x)*exp(3*x)

– **Arguments** =x

В геометрії моделі створіть відрізок [0,1].

Geometry -> **Primitives->** Interval

LMB > Build All Objects

Внутрішні вузли ламаної встановимо в точках $x_1=0.2, x_2=0.4, x_3=0.6, x_4=0.8$.

Створіть змінну моделі L, використовуючи формулу (3).

Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 1:

Variables – **Name, Expression** = L, a0+a1*x+c1*abs(x-0.2)+c2*abs(x-0.4)+c3*abs(x-0.6)+c4*abs(x-0.8)

Значення коефіцієнтів визначатиметься далі.

У вузлі Global Equations задайте шість рівнянь (4), яким повинні задовольняти коефіцієнти ламаної.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name, f(u, ut, utt, t)** = a0,at1(0,L)-F(0)

– **Name, f(u, ut, utt, t)** = c1,at1(0.2,L)-F(0.2)

– **Name, f(u, ut, utt, t)** = c2,at1(0.4,L)-F(0.4)

– **Name, f(u, ut, utt, t)** = c3,at1(0.6,L)-F(0.6)

– **Name, f(u, ut, utt, t)** = c4,at1(0.8,L)-F(0.8)

– **Name, f(u, ut, utt, t)** = a1,at1(1,L)-F(1)

Global Equations

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)
a0	at1(0,L)-F(0)
c1	at1(0.2,L)-F(0.2)
c2	at1(0.4,L)-F(0.4)
c3	at1(0.6,L)-F(0.6)
c4	at1(0.8,L)-F(0.8)
a1	at1(1,L)-F(1)

Нагадаємо, що оператор `at1(x0,expr)` обчислює вираз `expr` в будь якій одновимірній геометрії в точці з координатою `x0`. Тому, наприклад, вираз `at1(0,L)-F(0)`, записаний в першому рядку таблиці, представляє рівняння (4) для $x_0 = 0$, тобто є рівнянням $L(x_0) - F(x_0) = 0$.

Тепер можна розв'язати модель і надрукувати чисельні значення коефіцієнтів $a_0, a_1, c_1, c_2, c_3, c_4$.

`Home` **->** Compute

В гілці **Derived Values** будуть створені вузли типу `Global Evaluation` для обчислення величин `a0`, `a1`, `c1`, `c2`, `c3`,`c4`. Надрукуйте їх значення в спільну таблицю.

Derived Values – State variable a0: LMB > Evaluate
Derived Values – State variable a1: LMB > Evaluate - Table 1
Derived Values – State variable c1: LMB > Evaluate - Table 1
Derived Values – State variable c2: LMB > Evaluate - Table 1
Derived Values – State variable c3: LMB > Evaluate - Table 1
Derived Values – State variable c4: LMB > Evaluate - Table 1

State variable a0	State variable c1	State variable c2	State variable c3	State variable c4	State variable a1
16.198	2.5390	1.2733	-4.6756	-18.012	-13.521

Побудуйте графіки ламаної L і входової функції $F(x)$.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 1` -> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

`Selection` – **Selection** = `All domains`

`y-Axis Data` – **Expression** = `F(x)`

`x-Axis Data` – **Parameter, Expression** = `Expression,x`

`Coloring and Style` –**Width** = `3`

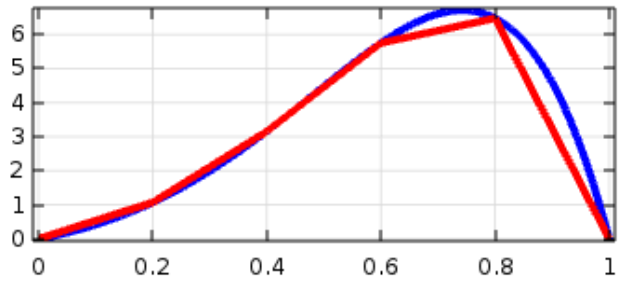
1D Plot Group 1 - Line Graph 1: RMB>Duplicate

1D Plot Group 2 - Line Graph 2:

`y-Axis Data` – **Expression** = `L`

`Coloring and Style` – **Color, Width** = `Red,3`

LMB > Plot



4.1.5. Наближення інтегральних і інтегро-диференціальних рівнянь.

В цьому параграфі будуть наведені приклади застосування Global ODEs and DAEs (ge) інтерфейсу для наближеного розв'язання рівнянь, які містять

інтегрування шуканих функцій. Це будуть приклади інтегральних і інтегро-диференціальних рівнянь.

Інтегральне рівняння Вольтерри 2-го роду [2] має форму

$$y(x)-\int_a^x K(x,t)y(t)dt = f(x), \qquad (1)$$

де $y(x)$ - невідома функція $(a \le x \le b)$, $K(x,t)$ - ядро інтегрального рівняння, $f(x)$ - деяка відома функція. Для неперервних $K(x,t), f(x)$ таких, що $K(x,x) \ne 0 \quad \forall x \in [a,b]$, задача має єдиний розв'язок.

Наближено функцію $y(x)$ можна шукати у вигляді полінома $P(x)=\sum_{k=0}^{n-1} c_k x^k$, підбір коефіцієнтів якого виконувати з умови того, що вираз

$$R \equiv P(x)-\int_0^x P(\xi)K(x,\xi)d\xi - f(x) \qquad (2)$$

обертається на нуль в n точках $a \le x_0 < x_1 < \ldots < x_{n-1} \le b$, тобто

$$R(x_i)=P(x_i)-\int_0^{x_i} P(\xi)K(x_i,\xi)d\xi - f(x_i)=0.\ i=0,1,\ldots,n-1. \qquad (3)$$

Це буде системою n алгебраїчних рівнянь відносно коефіцієнтів $c_0, \ldots, c_{n-1}$ полінома $P(x)$.

Приклад 1. Розв'язати інтегральне рівняння Вольтерри 2-го роду

$$y(x)+\int_0^x (x-t)y(t)dt = 2\operatorname{sh} x + 1.$$

Воно має аналітичний розв'язок $y(x)=\cos x+\sin x+\operatorname{sh} x$, і ми зможемо порівняти його з отриманим наближеним результатом.

Будемо шукати наближений розв'язок на відрізку [0,3] у вигляді кубічного полінома $P = c_0 + c_1 x + c_2 x^2 + c_3 x^3$. Підбір коефіцієнтів виконаємо з умови того, що вираз (2) обертатається на нуль в чотирьох точках $0 = x_0 < x_1 < x_2 < x_3 = 3$,

Розв'язання. Створіть одновимірну стаціонарну модель з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 1D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Створіть функцію ядра $K(x,t) = x - t$.

Global Definitions: RMB > Functions – Analytic

Function name = K

`Definition` – **Expression** =x-t

`Definition` – **Arguments** =x,t

В геометрії моделі створіть відрізок [0,3].

Geometry -> **Primitives->** Interval

Geometry – Interval 1 (i1):

`Interval` – **Right endpoint** = `3`

LMB > Build All Objects

Створіть оператор інтегрування по цьому відрізку.

Component 1 :> `Definitions` > `Component Couplings` - Integration

Component 1 - Definitions - Integration 1 (intop1):

`Source Selection` – **Selection** = `All domains`

Створіть змінну моделі *P* (вираз кубічного полінома) і по формулі (2) змінну *R*.

Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 1:

`Variables` – **Name, Expression** = `P, c0+c1*x+c2*x^2+c3*x^3`

Значення коефіцієнтів c_i визначатиметься пізніше.

`Variables` – **Name, Expression** =`R,`

`P+intop1(P*K(dest(x),x)*(x<=dest(x)))-2*sinh(x)-1`

Variables

Name	Expression
P	c0+c1*x+c2*x^2+c3*x^3
R	P+intop1(P*K(dest(x),x)*(x<=dest(x)))-2*sinh(x)-1

Зверніть увагу на те, що змінні задані на всій моделі, бо за замовчуванням встановлена опція **Geometric entity level**=`Entire model`.

Для зображення змінної R, яка обчислюється за формулою (2), спочатку було модифіковано інтеграл зі змінною верхньою межею в визначений інтеграл по відрізку

$$\int_a^X P(x)K(X,x)\,dx = \int_a^b P(x)K(X,x)(x \le X)\,dx .$$

Тут додатковий множник `(x<=X)` обнуляє ядро на ділянці `x>=X`. Щоб розрізнити просторову координату X і змінну інтегрування x, зовнішню абсцису X в операторі інтегрування `intop1(...)` по відрізку `[a,b]` було замінено оператором `dest(x)`. В результаті, останній інтеграл набув вигляду `intop1(P*K(dest(x),x)*(x<=dest(x)))`.

У вузлі `Global Equations` задайте рівняння (3), яким повинні задовольняти коефіцієнти c_i.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `c0, at1(0,R)`

– **Name, f(u, ut, utt, t)** = `c1, at1(1,R)`

– **Name, f(u, ut, utt, t)** = `c2, at1(2,R)`

– **Name, f(u, ut, utt, t)** = `c3, at1(3,R)`

Оператор `at1(x0,expr)` обчислює вираз `expr` в одновимірній геометрії в точці з координатою `x0`. Тому, наприклад, вираз `at1(0,R)`, записаний в першому рядку таблиці, представляє рівняння (3) для $x_0 = 0$, тобто є рівнянням

$$R(x_0) = P(x_0) - \int_0^{x_0} P(x)K(x_0,x)\,dx - f(x_0) = 0 .$$

Тепер можна розв'язати модель, знайшовши чисельні значення коефіцієнтів c_i.

Home -> Compute

В гілці **Derived Values** будуть створені вузли типу Global Evaluation для обчислення величин c0, c1, c2, c3. Надрукуйте їх значення в спільну таблицю.

Derived Values – State variable c0: LMB > Evaluate

Derived Values – State variable c1: LMB > Evaluate - Table 1

Derived Values – State variable c2: LMB > Evaluate - Table 1

Derived Values – State variable c3: LMB > Evaluate - Table 1

State variable c0	State variable c1	State variable c2	State variable c3
1.0000	2.6675	-1.7276	0.58082

Побудуйте графіки полінома *P* і аналітичного розв'язку

Results -> Plot Group - 1D Plot Group

1D Plot Group 1 -> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

Selection – **Selection** = All domains

y-Axis Data – **Expression** = P

x-Axis Data – **Parameter, Expression** = Expression,x

Coloring and Style –**Width** = 3

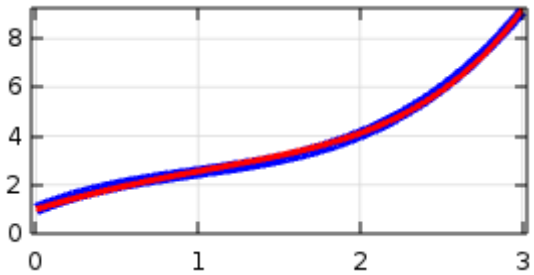
1D Plot Group 1 - Line Graph 1: RMB>Duplicate

1D Plot Group 2 - Line Graph 2:

y-Axis Data – **Expression** = cos(x)+sin(x)+sinh(x)

Coloring and Style – **Color** = Red

LMB > Plot



Як бачите, графіки полінома і аналітичного розв'язку майже співпадають.

Зауваження. В моделі поліном *P* представлено змінною, а не функцією тому, що функція не може містити невідомі параметри, а змінна – може.

■

Лінійне інтегральне рівняння Вольтерри 1-го роду має форму [2]

$$\int_a^x K(x,t)y(t)dt = f(x),$$

де $y(x)$ - невідома функція $(a \le x \le b)$, $K(x,t)$ - ядро інтегрального рівняння, $f(x)$ - деяка відома функція. Для неперервних $K(x,t), f(x)$ таких, що $K(x,x) \ne 0 \quad \forall x \in [a,b]$ і $f(a) = 0$, задача має єдиний розв'язок.

Вправа. Використовуючи поліноміальне наближення, на відрізку [0,2] розв'язати інтегральне рівняння Вольтери 1-го роду.

$$\int_0^x \left(2 + x^2 - t^2\right) y(t)dt = x^2.$$

Аналітичний розв'язок має вигляд $y(x)=x\,e^{-x^2/2}$.

Вказівка. В рівняннях $R(x_i)=\int_0^{x_i} P(\xi)K(x_i,\xi)d\,\xi - f(x_i)=0$ числа x_i не можна покладати рівними нулю. В такому випадку інтеграл і функція одночасно обертатимуться на нуль, і ми отримаємо не рівняння, а тавтологію 0=0.

Інтегральне рівняння Фредгольма 2-го роду має форму [2]

$$y(x)-\lambda\int_a^b K(x,t)y(t)d\,t = f(x), \tag{4}$$

де $y(x)$ - невідома функція $(a\le x\le b)$, $K(x,t)$ - ядро інтегрального рівняння, $f(x)$ - відома функція. Для зручності аналізу в рівнянні (4) зазвичай виділяють числовий параметр λ, який називають параметром інтегрального рівняння. Тут для спрощення покладемо $\lambda=1$.

В попередньому прикладі ми шукали наближений розв'язок інтегрального рівняння Вольтерри 2-го роду у вигляді кубічного полінома. Але можна обирати інші класи функцій, наприклад, шукати розв'язок у вигляді ламаної, кубічного сплайна, або тригонометричного полінома.

Припустимо, що ми бажаємо наблизити розв'язок задачі (4) ламаною Рівняння будь-якої ламаної можна записати у вигляді [3]

$$L(x)=a_0+a_1\,x+\sum_{i=1}^{n-1}c_i\,|\,x-x_i|, \tag{5}$$

де x_i - абсциси точок зламу. Нагадаємо, що в цій формулі абсциси $x_i\,(i=1,2,\ldots,n-1)$ повинні бути внутрішніми точками $a<x_1<x_2<\ldots<x_{n-1}<b$ відрізка [*a*,*b*] апроксимування шуканої функції $y(x)$.

Підбір коефіцієнтів апроксимуючої ламаної (5) виконується з умови, що вираз $R\equiv L(x)-\int_a^b L(\xi)K(x,\xi)d\,\xi - f(x)$ обертається на нуль в точках, кількість яких дорівнює кількості шуканих коефіцієнтів. Для цього на відрізку [*a*,*b*] обираємо n+1 точку x_i, $a=x_0<x_1<\ldots<x_{n-1}<x_n=b$, і вважаємо, що одночасно виконуються рівності

$$R(x_i)=L(x_i)-\int_a^b L(\xi)K(x_i,\xi)d\,\xi - f(x_i)=0,\ i=0,1,\ldots,n. \tag{6}$$

Це і буде системою рівнянь відносно шуканих коефіцієнтів $a_0, a_1, c_1, c_2, \ldots, c_{n-1}$.

Приклад 2. В формі ламаної знайти наближений розв'язок інтегрального рівняння Фредгольма 2-го роду

$$y(x)-\int_0^1 \frac{x}{1+t^2}\,y(t)\,d\,t=1+x^2.$$

Задача має аналітичний розв'язок $y(x)=1+x^2+\dfrac{2\,x}{2-\ln 2}$.

Розв'язання. Створіть одновимірну стаціонарну модель з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 1D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Створіть функцію ядра $K(x,t)$.

Global Definitions: RMB > Functions – Analytic

Function name = `K`

`Definition` – **Expression** =`x/(1+t^2)`

`Definition` – **Arguments** =`x,t`

В геометрії моделі створіть відрізок [0,1].

`Geometry` -> `Primitives`-> Interval

LMB > Build All Objects

Створіть оператор інтегрування по відрізку [0,1].

Component 1 :> `Definitions` > Component Couplings - Integration

Component 1 - Definitions - Integration 1 (intop1):

`Source Selection` – **Selection** = `All domains`

Будемо шукати ламану з двома внутрішніми вузлами в точках з абсцисами `0.3` і `0.7`. Створіть змінну моделі *L* (рівняння ламаної в формі (5)) і змінну *R*.

Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 1:

`Variables` – **Name, Expression** = `L`,
`a0+a1*x+c1*abs(x-0.3)+c2*abs(x-0.7)`

Коефіцієнти визначимо пізніше.

`Variables` – **Name, Expression** =`R`,
`L-intop1(L*K(dest(x),x))-1-x^2`

Variables

Name	Expression
L	a0+a1*x+c1*abs(x-0.3)+c2*abs(x-0.7)
R	L-intop1(L*K(dest(x),x))-1-x^2

У вузлі `Global Equations` для внутрішніх вузлів з абсцисами `0.3` і `0.7`, та в двох кінцевих точках відрізка `[0,1]` задайте рівняння (6).

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `a0, at1(0.0,R)`
– **Name, f(u, ut, utt, t)** = `c1, at1(0.3,R)`
– **Name, f(u, ut, utt, t)** = `c2, at1(0.7,R)`
– **Name, f(u, ut, utt, t)** = `a1, at1(1,R)`

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)	Initial value
a0	at1(0.0,R)	0
c1	at1(0.3,R)	0
c2	at1(0.7,R)	0
a1	at1(1,R)	0

Тепер можна розв’язувати задачу знаходження чисельних значень коефіцієнтів в рівнянні ламаної, і аналізувати результати.

Home -> Compute

В гілці **Derived Values** будуть створені вузли типу Global Evaluation для обчислення величин a0, a1, c1, c2. Надрукуйте їх значення в спільну таблицю.

Derived Values – State variable a0 : LMB > Evaluate

Derived Values – State variable c1: LMB > Evaluate - Table 1

Derived Values – State variable c2: LMB > Evaluate - Table 1

Derived Values – State variable a1: LMB > Evaluate - Table 1

State variable a0	State variable c1	State variable c2	State variable a1
0.65000	0.35000	0.35000	2.5540

Побудуйте графіки ламаної L і аналітичного розв'язку.

Results -> Plot Group - 1D Plot Group

1D Plot Group 1 -> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

Selection – **Selection** = All domains

y-Axis Data – **Expression** = L

x-Axis Data – **Parameter, Expression** = Expression,x

Coloring and Style –**Width** = 3

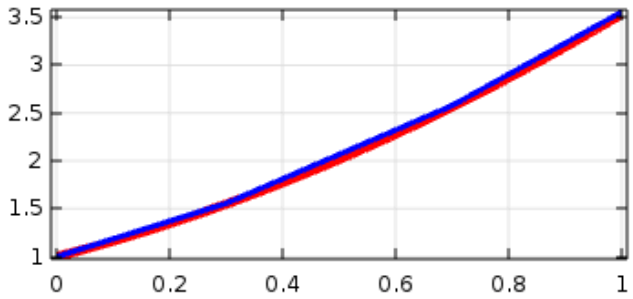
1D Plot Group 1 - Line Graph 1: RMB>Duplicate

1D Plot Group 2 - Line Graph 2:

y-Axis Data – **Expression** = 1+x^2+2*x/(2-log(2))

Coloring and Style – **Color** = Red

LMB > Plot



■

Крім полінома і ламаної, для отримання наближеного розв'язку можна використовувати кубічні сплайни. Для них загальний аналітичний вираз можна записати в формі [3]

$$S(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \sum_{i=1}^{n-1} c_1 \left| x - x_i \right|^3, \qquad (7)$$

де x_i абсциси точок стику кубічних сегментів сплайна. При апроксимуванні функцій на відрізку $[a,b]$ ці абсциси x_i $(i = 1, 2, \ldots, n-1)$ потрібно розташовувати всередині відрізка (вони не повинні співпадати з його кінцями).

Вираз (7) є двічі неперервно диференційованим. Тому його можна застосовувати навіть для наближеного розв'язання рівнянь, які містять похідні, наприклад, для інтегро-диференціальних рівнянь.

Інтегро-диференціальне рівняння Вольтерри 2-го роду має форму [4]

$$L[y(x)] - \int_a^x K(x,\xi)\, y(\xi)\, d\xi = f(x),$$

де L – деякий диференціальний оператор. Для однозначного визначення розв'язку треба додавати граничні умови в точці a.

Покладемо в останньому рівнянні $L[y(x)] = y'(x)$, і розглянемо задачу

$$y'(x) - \int_a^x K(x,\xi) y(\xi) d\xi = f(x),\ y(a) = y_0. \qquad (8)$$

На відрізку $[a,b]$ знайдемо її наближений розв'язок у вигляді кубічного сплайна, тобто $y(x) \approx S(x)$. Підбір коефіцієнтів в формулі (7) виконується з умови, що вираз $R \equiv S'_x(x) - \int_a^x S(\xi) K(x,\xi) d\xi - f(x)$ обертається на нуль в точках, кількість яких на одиницю менше кількості шуканих коефіцієнтів. Для цього на напівінтервалі $(a,b]$ обираємо n+2 точки x_i, $a < x_1 < \ldots < x_n < x_{n+1} < x_{n+2} = b$, для яких вважатимемо, що одночасно виконуються рівності

$$R(x_i) = S'(x_i) - \int_a^x S(\xi) K(x_i,\xi) d\xi - f(x_i) = 0,\ i = 1,\ldots,n+2.$$

Ще одне співвідношення ми отримаємо, враховуючи початкову умову $y(a) = y_0$, яку перенесемо на шуканий сплайн у вигляді $S(a) = y_0$. Це утворює замкнену систему рівнянь відносно коефіцієнтів $a_0, \ldots, a_3, c_1, c_2, \ldots, c_{n-1}$.

Приклад 3. Розв'язати інтегро-диференціальне рівняння

$$y'(x) - \int_0^x (x-t) y(t) dt = -1,\ y(0) = 1,$$

наблизивши розв'язок $y(x)$ на відрізку [0,6] кубічним сплайном (7). Порівняти його з аналітичним розв'язком $y(x) = e^{-x/2}\left(\cos\frac{\sqrt{3}}{2}x - \frac{1}{\sqrt{3}}\sin\frac{\sqrt{3}}{2}x\right)$,

Розв'язання. Створіть одновимірну стаціонарну модель з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 1D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Створіть функцію ядра $K(x,t)$.

Global Definitions: RMB > Functions – Analytic

Function name = K

`Definition` – Expression =x-t

`Definition` – Arguments =x,t

В геометрії моделі створіть відрізок [0,6].

Geometry -> **`Primitives`**-> Interval

Geometry – Interval 1 (i1):

`Interval` – Right endpoint = 6

LMB > Build All Objects

Створіть оператор інтегрування по цьому відрізку.

Component 1 :> `Definitions` > Component Couplings - Integration
Component 1 - Definitions - Integration 1 (intop1):

`Source Selection` – **Selection** = `All domains`

Створіть змінні моделі S і R. Враховуючи довжину формули (7), величину `S` представимо у вигляді суми двох змінних `S1` і `S2`.

Component 1(comp1) – Definitions: RMB > Variables
Definitions – Variables 1:

`Variables` – **Name, Expression** = `S1, a0+a1*x+a2*x^2+a3*x^3`

`Variables` – **Name, Expression** = `S2,`
`c1*abs(x-2)^3+c2*abs(x-3)^3+c3*abs(x-4)^3`

`Variables` – **Name, Expression** = `S, S1+S2`

Коефіцієнти в цих виразах визначатимуться пізніше.

`Variables` – **Name, Expression** =`R,`
`d(S,x)-intop1(S*K(dest(x),x)*(x<=dest(x)))+1`

Variables

Name	Expression
S1	a0+a1*x+a2*x^2+a3*x^3
S2	c1*abs(x-2)^3+c2*abs(x-3)^3+c3*abs(x-4)^3
S	S1+S2
R	d(S,x)-intop1(S*K(dest(x),x)*(x<=dest(x)))+1

Верхня межа інтегрування нашої задачі змінювана, тому під інтегралом у виразі для R ми додали множник $(x \le \mathrm{dest}(x))$. Інакше інтегрування тлумачилося б по всьому відрізку [0,6]. Отже тут ми маємо

$$S'(x) - \int_0^x S(\xi)\, K(x,\xi)\, d\xi = S'(x) - \int_0^6 S(\xi)\, K(x,\xi) \cdot (\xi < x)\, d\xi =$$
$$= d(S,x) - \mathrm{intop1}(\mathrm{S} * \mathrm{K}(\mathrm{dest}(x),x) * (x <= \mathrm{dest}(x)))$$

Зауваження. Замість множника $(x <= \mathrm{dest}(x))$ можна було б використати функцію-сходинку (step function) у вигляді $\left(1 - \mathrm{step1}(x - \mathrm{dest}(x))\right)$.

У вузлі `Global Equations` задайте потрібну кількість рівнянь $R(x_i) = 0$. Граничну умову $y(0) = 1$ теж треба додати в таблицю у вигляді `at1(0,S)-1`, тобто $S(0) = 1$.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `a0, at1(0,S)-1`
– **Name, f(u, ut, utt, t)** = `c1, at1(2,R)`
– **Name, f(u, ut, utt, t)** = `c2, at1(3,R)`
– **Name, f(u, ut, utt, t)** = `c3, at1(4,R)`
– **Name, f(u, ut, utt, t)** = `a1, at1(5,R)`
– **Name, f(u, ut, utt, t)** = `a2, at1(5.5,R)`
– **Name, f(u, ut, utt, t)** = `a3, at1(6,R)`

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)
a0	at1(0,S)-1
c1	at1(2,R)
c2	at1(3,R)
c3	at1(4,R)
a1	at1(5,R)
a2	at1(5.5,R)
a3	at1(6,R)

Тепер можна розв'язувати задачу знаходження чисельних значень коефіцієнтів сплайна (7), і аналізувати результати.

`Home` **->** Compute

Після завершення обчислень доречно побудувати графіки сплайна S і аналітичного розв'язку.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 1` -> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

`Selection` – **Selection** = `All domains`

`y-Axis Data` – **Expression** = `S`

`x-Axis Data` – **Parameter, Expression** = `Expression,x`

`Coloring and Style` –**Width** = `3`

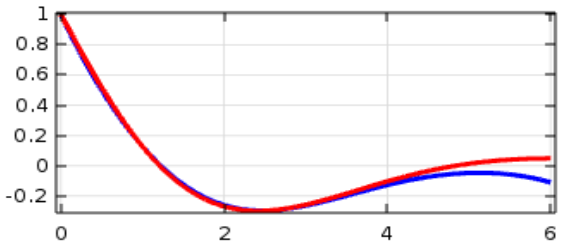
1D Plot Group 1 - Line Graph 1: RMB>Duplicate

1D Plot Group 2 - Line Graph 2:

`y-Axis Data` – **Expression** = `exp(-x/2)*`
`(cos(sqrt(3)/2*x)-1/sqrt(3)*sin(sqrt(3)/2*x))`

`Coloring and Style` – **Color** = `Red`

LMB > Plot



Ми обрали досить довгий відрізок апроксимування. Тому для кращого наближення потрібно було б обирати більшу кількість сплайнових сегментів і відповідну кількість шуканих коефіцієнтів c_i.

■

Розглянемо рівняння Вольтерри 2-го роду

$$y(x)-\int_a^x K(x,t)y(t)dt=f(x).$$

Загальна схема застосування метода послідовних наближень [2] при розв'язанні цього рівняння полягає в пошуку розв'язку у вигляді

$$y(x)=f(x)+\sum_{n=1}^{\infty}y_n(x),$$

де $y_n(x)$ визначаються за формулами

$$y_1(x)=\int_a^x K(x,t)\,f(t)\,dt$$

$$y_2(x)=\int_a^x K(x,t)\,y_1(t)\,dt \qquad (9)$$

$$y_3(x)=\int_a^x K(x,t)\,y_2(t)\,dt$$

. . . . . .

З кожним кроком вираз $Y_N(x)=f(x)+\sum_{n=1}^{N} y_n(x)$ все краще апроксимує шукану функцію $y(x)$.

Приклад 4. На відрізку [0,3] методом послідовних наближень розв'язати інтегральне рівняння Вольтерри 2-го роду

$$y(x)-\int_0^x e^{-(x-t)}\sin(x-t)\,y(t)\,dt=e^{-x}.$$

Порівняти наближений результат $Y_3(x)$ з аналітичним розв'язком $y(x)=e^{-x}\left(x^2/2+1\right)$.

Розв'язання. Створіть одновимірну стаціонарну модель з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 1D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Створіть функцію ядра $K(x,t)$.

Global Definitions: RMB > Functions – Analytic

Function name = K

`Definition` – Expression =exp(-x+t)*sin(x-t)

`Definition` – Arguments =x,t

Створіть функцію вільного члена $f(x)$.

Global Definitions: RMB > Functions – Analytic

Function name = f

`Definition` – Expression =exp(-x)

`Definition` – Arguments =x

В геометрії моделі створіть відрізок [0,3].

Geometry -> **`Primitives`**-> Interval

Geometry – Interval 1 (i1):

`Interval` – Right endpoint = 3

LMB > Build All Objects

Створіть оператор інтегрування по цьому відрізку.

Component 1 :> Definitions > Component Couplings - Integration

Component 1 - Definitions - Integration 1 (intop1):

`Source Selection` – Selection = All domains

Заповніть вузол змінних моделі, створюючи проміжні змінні $y1$, $y2$, $y3$ за формулами (9), і результуючу змінну $Y = f(x) + \sum_{i=1}^{3} y_i(x)$

Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 1:

Variables – **Name, Expression** = `y1,`
`intop1(f(x)*K(dest(x),x)*(x<=dest(x)))`

Variables – **Name, Expression** = `y2,`
`intop1(y1*K(dest(x),x)*(x<=dest(x)))`

Variables – **Name, Expression** = `y3,`
`intop1(y2*K(dest(x),x)*(x<=dest(x)))`

Variables – **Name, Expression** = `Y, f(x)+y1+y2+y3`

Variables

Name	Expression
y1	intop1(f(x)*K(dest(x),x)*(x<=dest(x)))
y2	intop1(y1*K(dest(x),x)*(x<=dest(x)))
y3	intop1(y2*K(dest(x),x)*(x<=dest(x)))
Y	f(x)+y1+y2+y3

Зверніть увагу на послідовне застосування оператора `dest(x)` в якості першого аргументу функції ядра $K(x,\xi)$.

Виконайте обчислення.

`Home` -> Compute

Зауваження. Таблиця рівнянь у вузлі **Global Equations 1** покинута порожньою.

Побудуйте графіки змінної Y та аналітичного розв'язку.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 1` -> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

Selection – **Selection** = `All domains`

y-Axis Data – **Expression** = `exp(-x)*(x^2/2+1)`

x-Axis Data – **Parameter, Expression** = `Expression,x`

Coloring and Style –**Width** = `3`

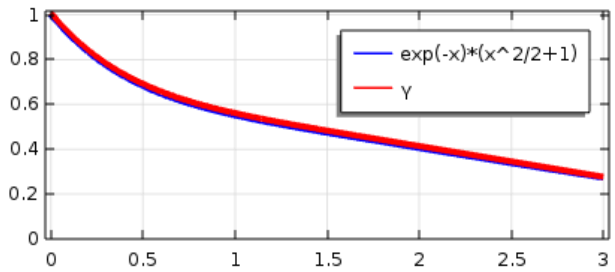
1D Plot Group 1 - Line Graph 1: RMB>Duplicate

1D Plot Group 2 - Line Graph 2:

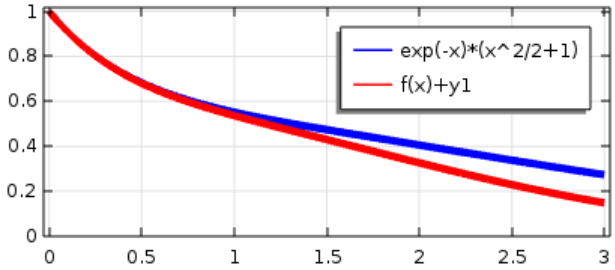
y-Axis Data – **Expression** = `Y`

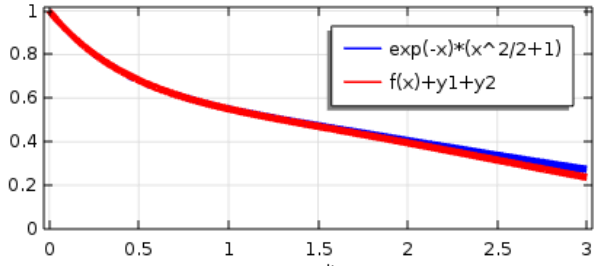
Coloring and Style – **Color** = `Red`

LMB > Plot



На наступному рисунку ліворуч наведено графік функції $Y_1 = f(x) + y_1(x)$, а праворуч – функції $Y_2 = f(x) + y_1(x) + y_2(x)$ (побудуйте їх самостійно).





4.1.6. Приклади допоміжних просторово незалежних рівнянь.

Всі задачі, які розглядалися в поточній главі, формулювалися у вигляді диференціальних рівнянь з використанням Global ODEs and DAEs інтерфейсу, який був єдиним в моделі. Але в COMSOL головним призначенням цього інтерфейсу є застосування в якості допоміжного до моделей з ДРЧП. Тут ми розглянемо приклади застосування цього інструменту, як допоміжного, а також покажемо використання деяких операторів.

Нехай в квадраті $\Omega = [-1,1] \times [-1,1]$ дано рівняння Пуассона $\Delta u = f$ з нульовими граничними умовами. Відомо, що f – стала, але така, що:

1. Інтеграл від розв'язку u по контуру кола радіуса r з центром в точці (x_0, y_0) дорівнює заданій величині a.
2. Інтеграл від розв'язку u по колу радіуса r з центром в точці (x_0, y_0) дорівнює заданій величині a.
3. Середнє значення розв'язку u по контуру кола радіуса r з центром в точці (x_0, y_0) дорівнює заданій величині a.
4. Середнє значення розв'язку u по колу радіуса r з центром в точці (x_0, y_0) дорівнює заданій величині a.
5. Середнє значення розв'язку u по області квадрата дорівнює заданій величині a.
6. Інтеграл розв'язку u по діагоналі квадрата дорівнює заданій величині a.
7. Значення розв'язку u в точці (x_0, y_0) дорівнює заданій величині a. (тобто $u(x_0, y_0) = a$).
8. Сума значень розв'язку в n точках, рівномірно розосереджених по контуру кола радіуса r з центром в точці (x_0, y_0), дорівнює заданій величині a.

Для кожної з умов потрібно знайти f (і звичайно u).

Р о з в ' я з а н н я . За допомогою майстра створіть безрозмірну 2D модель задачі

File -> New -> Model Wizard -> 2D -> Mathematics - Classical PDEs - Poisson's Equation (poeq) -> Stationary

Untitled.mph (root): `Unit System` – <> = None

Задайте параметри моделі.

Global Definitions : RMB > Parameters

Global Definitions – Parameters :

Parameters – **Name, Expression** = a, -1
– **Name, Expression** = r, 0.5
– **Name, Expression** = x0, 0.3
– **Name, Expression** = y0, 0.3

Геометрія моделі складатиметься з одного квадрата і допоміжного кола. Воно потрібно для того, щоб користувач бачив зону, по який виконується «допоміжне» обчислення. Коло можна зобразити на графіку розв'язку, але ми створимо його в геометрії моделі.

Geometry -> **Primitives** – Square

Geometry 1 - Square 1 (sq1):

Size – **Size length** = 2

Position – **Base, x, y** = Center,0,0

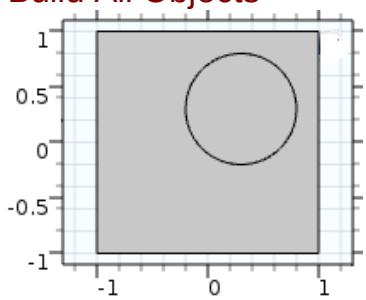
Geometry -> **Primitives** – Circle

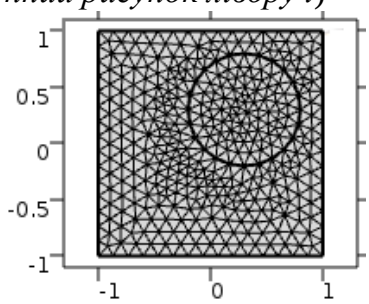
Geometry 1 - Circle 1:

Size and Shape – **Radius:** = r

Position – **Base, x, y** = Center,x0,y0

LMB > Build All Objects (*наступний рисунок ліворуч*)





На створеній геометрії генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB> Build All (*попередній рисунок праворуч*)

Налаштуйте стаціонарну задачу (Poisson's Equation), задавши коефіцієнти рівняння Пуассона і нульові граничні умови. Але спочатку перевірте, що в ПВ гілки **Poisson's Equation (poeq)** в розділі **Domain Selection** в секції вибору стоять індикатори 1, 2 квадрата і кола. Далі введіть

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = f (*зовнішнє зусилля* f *буде знайдено окремо*)

Physics -> **Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection–**Selection,**<>= Manual,1,2,3,4 (*контур квадрата*)

Dirichlet Boundary Condition – **r** = 0

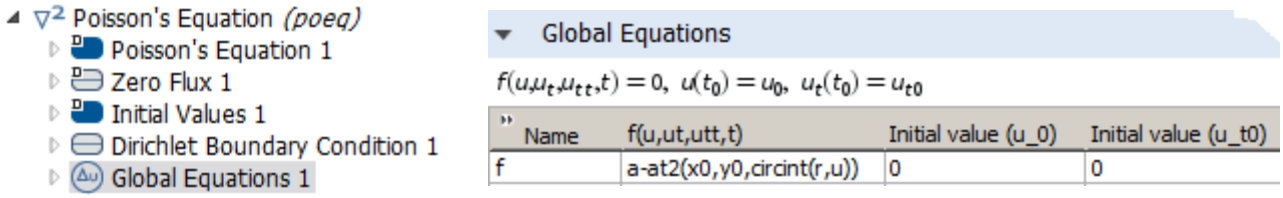
1. Додайте вузол Global Equations, в якому вкажіть невідому f, а в якості рівняння напишіть умову, сформульовану першим пунктом на початку прикладу.

Physics -> **Global** – Global Equations

Poisson's Equation (poeq) - Global Equations 1:

Global Equations – **Name** = `f`

Global Equations – **f(u,ut,utt,t)** = `a-at2(x0,y0,circint(r,u))`



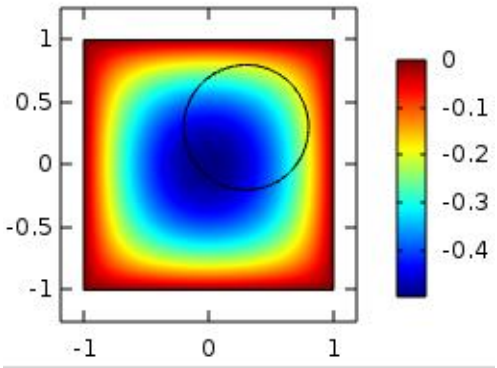
Тут застосовано оператор `circint(r,expr)`, який в 2D моделях знаходить криволінійний інтеграл від виразу `expr` по контуру кола радіуса `r` навкруги точки, в якій він обчислюється. Щоб визначити точку, оператор `circint` було використано в якості аргументу оператора `at2(x0,y0,expr)`. Нагадаємо, що цей оператор обчислює вираз `expr` в двовимірній геометрії в точці з заданими координатами. Таким чином, вираз `at2(x0,y0,circint(r,u))` обчислює інтеграл по контуру кола радіуса `r` з центром в точці `x0,y0`.

Розуміти запис в таблиці вузла **Global Equations 1** потрібно наступним чином. Задача буде розв'язуватися відносно невідомої f (зовнішнє зусилля в рівнянні Пуассона) одночасно з пошуком величини `u`, доки вираз в комірці **f(u,ut,utt,t)** не буде дорівнювати нулю, тобто доки інтеграл по контуру кола не співпаде з `a`.

Виконайте обчислення.

`Home` **->** Compute

В гілці **Results** буде створено вузол **2D Plot Group 1 - Surface 1** з графіком розв'язку.



Також створюється вузол **Derived Values - State variable f**, з якого можна надрукувати значення шуканої величини f.

Derived Values - State variable f: RMB > Evaluate

State variable f
1.6872

Отже `f=1.6872` є розв'язком першого завдання.

2. Розв'яжіть друге завдання.

Відкрийте ПВ вузла **Poisson's Equation (poeq) - Global Equations 1** і відредагуйте в ньому рівняння.

Poisson's Equation (poeq) - Global Equations 1:

Global Equations – **f(u,ut,utt,t)** = `a-at2(x0,y0,diskint(r,u))`

Оператор `diskint(r,expr)` в 2D моделях знаходить подвійний інтеграл від виразу `expr` по площі кола радіуса `r` з центром в точці, в якій він обчислюється. Для визначення точки оператор `diskint` застосовується в якості аргументу оператора `at2(...)`. Таким чином, вираз `at2(x0,y0,diskint(r,u))` обчислює інтеграл по колу радіуса `r` з центром в точці `x0,y0`.

Виконайте обчислення.

`Home` **->** Compute

Надрукуйте значення шуканої величини `f`.

Derived Values - State variable f: RMB > Evaluate – Table 1

State variable f	State variable f
1.6872	5.7867

Таким чином, `f= 5.7867` є розв'язком другого завдання.

3. Розв'яжіть третє завдання.

Середнє значення розв'язку u по контуру кола радіуса r обчислюється за допомогою оператора `circavg(r,expr)`. Використовуючи його в якості аргументу оператора `at2(...)` ми матимемо потрібний вираз для обчислення середнього значення по контуру кола з центром в точці (x_0, y_0).

Poisson's Equation (poeq) - Global Equations 1:

`Global Equations` – **f(u,ut,utt,t)** = `a-at2(x0,y0, circavg(r,u))`

`Home` **->** Compute

Надрукуйте значення шуканої величини `f`.

Derived Values - State variable f: RMB > Evaluate – Table 1

State variable f	State variable f	State variable f
1.6872	5.7867	5.3005

Таким чином, `f= 5.3005` є розв'язком третього завдання.

4. Розв'яжіть четверте завдання.

Середнє значення розв'язку u по площі кола радіуса r обчислюється за допомогою оператора `diskavg(r,expr)`. Використовуючи його в якості аргументу оператора `at2(...)` ми матимемо потрібний вираз для обчислення середнього значення по колу з центром в точці (x_0, y_0).

Poisson's Equation (poeq) - Global Equations 1:

`Global Equations` – **f(u,ut,utt,t)** = `a-at2(x0,y0, diskavg(r,u))`

`Home` **->** Compute

Надрукуйте значення шуканої величини `f`.

Derived Values - State variable f: RMB > Evaluate – Table 1

State variable f	State variable f	State variable f	State variable f
1.6872	5.7867	5.3005	4.5449

Таким чином, `f= 4.5449` є розв'язком четвертого завдання.

5. Розв'яжіть п'яте завдання.

Для обчислення виразів всередині розрахункової області, які можна застосовувати в інших місцях моделі, призначені оператори зв'язку (`Component Coupling`). Вони створюються в гілці **Definitions** компонента за допомогою

кнопки - списку Component Coupling панелі інструментів Definitions або однойменної команди з випадного меню цієї гілки. Серед цих операторів є Integration і Average. Для розв'язання нашого завдання потрібно створити оператор Average.

Component 1:

Definitions -> **Component Coupling** – Average

Component 1 (comp1) - Definitions - Average 1 (aveop1):

Source Selection – **Geometric entity level** = Domain

Остання інструкція вказує «рівень геометрії», по якій буде обчислюватися середнє значення (лінія, поверхня і т.д.).

– **Selection** = All domains (*вся зона квадрата*)

Тепер оператор aveop1 обчислення середнього значення по області квадрата створений і буде доступний під час розв'язання задачі, наприклад, у вузлі **Global Equations 1**.

Зауваження. Один і той же оператор aveop1 можна використовувати для обчислення середнього значення різних виразів.

Продемонструємо використання оператора aveop1 при розв'язанні п'ятого завдання. Відкрийте ПВ вузла **Poisson's Equation (poeq) - Global Equations 1** і відредагуйте в ньому рівняння.

Poisson's Equation (poeq) - Global Equations 1:

Global Equations – **f(u,ut,utt,t)** = a-aveop1(u)

Переобчисліть модель.

Home -> Compute

Надрукуйте значення шуканої величини f.

Derived Values - State variable f: RMB > Evaluate – Table 1

State variable f	State variable f	State variable f	State variable f	State variable f
1.6872	5.7867	5.3005	4.5449	7.1137

Таким чином середнє значення розв'язку u по області квадрата дорівнюватиме заданій величині $a = -1$ при f= 7.1137.

6. Розв'яжіть шосте завдання. Воно виконується схожим чином з попереднім – створенням оператора інтегрування Integration вздовж діагоналі квадрата.

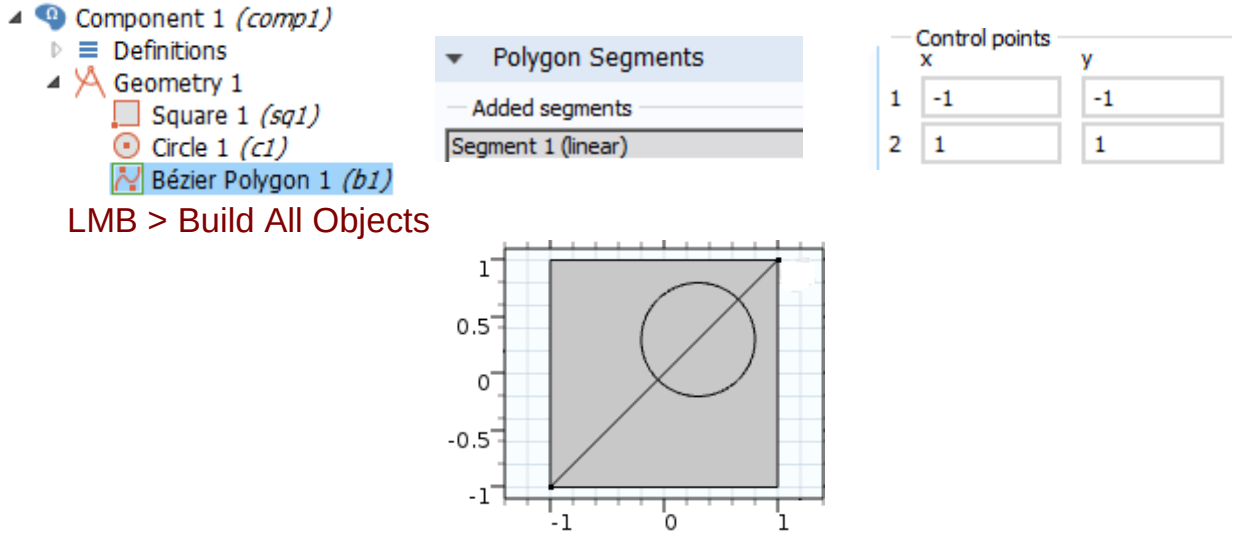
Зауваження. Використовувані тут оператори Integration і Average називаються операторами зв'язку (Coupling operators) тому, що їх можна визивати з інших фізик в межах поточного компонента, а також з інших компонентів. Таким чином, вони виконують функції зв'язку між різними фізичними моделями (Component Coupling).

Оператору інтегрування потрібно вказати геометрію, на якій він буде діяти. Тому спочатку створіть відрізок діагоналі, по якому буде виконуватися інтегрування.

Geometry -> **Primitives** – Bézier Polygon

Geometry 1 - Bézier Polygon 1 (b1): LMB > Add Linear

Введіть координати початкової (-1,-1) і кінцевої (1,1) точок відрізка.



LMB > Build All Objects

Тепер можна створювати оператор інтегрування по цій діагоналі.

Component 1 :> `Definitions` -> **`Component Coupling`** – Integration

Component 1 (comp1) - Definitions - Integration 1 (intop1):

`Source Selection` – **Geometric entity level** = `Boundary`

Остання інструкція вказує, що інтегрування буде виконуватися по лінії.

– **Selection** = `Manual,2,5,6` (*всі ділянки відрізка діагоналі*)

Оператор `intop1` інтегрування по діагоналі створений і буде доступний під час обчислення.

Відкрийте ПВ вузла **Poisson's Equation (poeq) - Global Equations 1** і відредагуйте в ньому рівняння.

Poisson's Equation (poeq) - Global Equations 1:

`Global Equations` – **f(u,ut,utt,t)** = `a-intop1(u)`

Переобчисліть модель.

`Home` -> Compute

Надрукуйте значення шуканої величини `f`.

Derived Values - State variable f: RMB > Evaluate – Table 1

State variable f	State variable f	State variable f	State variable f	State variable f	State variable f
1.6872	5.7867	5.3005	4.5449	7.1137	2.1213

Таким чином інтеграл від розв'язку u по діагоналі квадрата дорівнюватиме заданій величині $a = -1$ при `f=2.1213`.

Вправа. Створіть оператор зв'язку `Integration` по контуру кола, зображеному в геометрії. Використовуючи його, знайдіть f таке, що інтеграл від розв'язку u по контуру кола дорівнюватиме заданій величині a. (Відповідь `f=1.6872` та ж сама, що і при розв'язанні першого завдання).

7. Розв'яжіть сьоме завдання. Воно потребує застосування оператора `at2(x0,y0,expr)` – обчислення виразу `expr` в двовимірній геометрії в точці з заданими координатами. В нашому прикладі в якості виразу `expr` потрібно використати шукану величину `u`.

Poisson's Equation (poeq) - Global Equations 1:

`Global Equations` – **f(u,ut,utt,t)** = `a-at2(x0,y0,u)`

Переобчисліть модель.

Home -> Compute

Надрукуйте значення шуканої величини f.

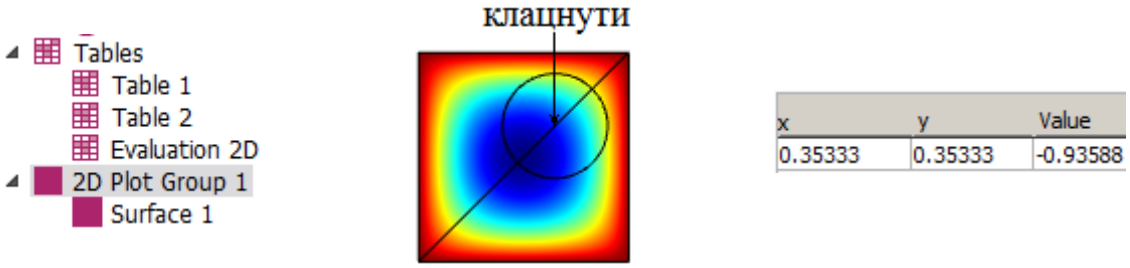
Derived Values - State variable f: RMB > Evaluate – Table 1

State variable f	State variable f	State variable f	State variable f	State variable f	State variable f	State variable f
1.6872	5.7867	5.3005	4.5449	7.1137	2.1213	3.9815

Таким чином, коли f= 3.9815 , то u(x0,y0)=-1.

Ви можете спробувати перевірити розв'язок. Нагадаємо, що після клацання мишею по бажаній точці на Surface графіку, система надрукуєте значення розв'язку u в цій точці (потрібно щоб відповідний графічний вузол в дереві моделі був активований). Проблема полягає лише в тому, щоб максимально близько поцілити в точку.

Активуйте вузол **2D Plot Group 1** і спробуйте клацнути лівою кнопкою миші по центру кола, що накреслено поверх графіка. Буде створена таблиця **Evaluation 2D** з надрукованими координатами точки і значенням розв'язку u. Оскільки важко потрапити мишею точно в точку, то ми отримаємо значення в деякій наближеній точці.



Повторні клацання будуть додавати в створену таблицю нові рядки з координатами нових точок і значеннями розв'язку у них.

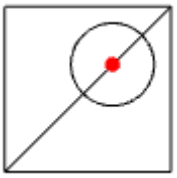
Є ще один спосіб перевірити значення отриманого розв'язку в точці. Треба створити джерело даних Cut Point 2D і, використовуючи його, обчислити вираз u.

Results – Data Sets: RMB > Cut Point 2D

Data Sets - Cut Point 2D 1:

Point Data – **x, y** = x0,y0

LMB > Plot



Derived Values: RMB > Point Evaluation

Derived Values - Point Evaluation 1:

Data – **Data set** = Cut Point 2D 1

Expression – **Expression** = u

LMB > Evaluate – New Table

Dependent variable u, Point: (0.3, 0.3)
-1.0000

8. Розв'яжіть восьме завдання. Воно потребує застосування операторів `at2(...)` і `sum(expr,index,lower,upper)`.

Створіть два нових параметра моделі `n` і `p`.

Global Definitions – Parameters :

`Parameters` – **Name, Expression** = `n, 6`

– **Name, Expression** = `p, 2*pi/n`

Тут `n` представлятиме кількість точок на колі, а `p` – центральний кут між двома сусідніми точками.

Poisson's Equation (poeq) - Global Equations 1:

`Global Equations` – **f(u,ut,utt,t)** =

`a-sum(at2(x0+r*cos(k*p),y0+r*sin(k*p),u),k,0,n-1)`

Переобчисліть модель.

`Home` **->** Compute

Надрукуйте значення шуканої величини `f`.

Derived Values - State variable f: RMB > Evaluate – Table 1

State variable f	State variable f	State variable f	State variable f	State variable f	State variable f	State variable f	State variable f
1.6872	5.7867	5.3005	4.5449	7.1137	2.1213	3.9815	0.88343

Таким чином, коли `f= 0.88343`, сума значень розв'язку в `6` точках, рівномірно розосереджених по контуру кола, дорівнює `-1`. Тобто при обраних значеннях параметрів маємо

$$\sum_{k=0}^{n-1} u\left(x_0 + r\cos(k\varphi), y_0 + r\sin(k\varphi)\right) = a \text{, де } \varphi = \frac{2\pi}{n}.$$

Можна надрукувати значення розв'язку `u` в обраних точках. Спочатку треба створити джерело даних для множини цих точок.

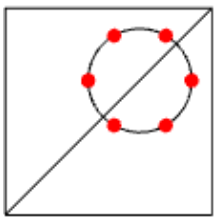
Results – Data Sets: RMB > Cut Point 2D

Data Sets - Cut Point 2D 2:

`Point Data` – **x** = `x0+r*cos({range(0,1,n-1)}*2*pi/6)`

`Point Data` – **y** = `y0+r*sin({range(0,1,n-1)}*2*pi/6)`

LMB > Plot



Потім потрібно створити вузол типу `Point Evaluation`, який опиратиметься на це джерело даних, і обчислити його.

Derived Values: RMB > Point Evaluation

Derived Values - Point Evaluation 2:

`Data` – **Data set** = `Cut Point 2D 2`

`Expression` – **Expression** = `u`

LMB > Evaluate – New Table

Корисно відредагувати назви стовпців створеної таблиці, залишивши в них лише координати точок. Активуйте вузол **Results – Tables – Table 3** і в таблиці

розділу **Column Headers** видаліть у всіх рядках зайвий текст, залишивши лише координати точок.

Column Headers

Column	Header
1	(0.8, 0.3)
2	(0.55, 0.73301)
3	(0.05, 0.73301)
4	(-0.2, 0.3)
5	(0.05, -0.13301)
6	(0.55, -0.13301)

LMB > Update

(0.8, 0.3)	(0.55, 0.73301)	(0.05, 0.73301)	(-0.2, 0.3)	(0.05, -0.13301)	(0.55, -0.13301)
-0.095966	-0.098990	-0.12987	-0.23209	-0.25587	-0.18721

Зауваження. Операції (Average, Maximum, Integral та ін.) з розділу **Data Series Operation** вузлів Derived Values застосовуються до даних в полі **Expression** за змінною часу, а не по просторовим змінним. Тому в разі статичних задач вони (операції) не приносять користі.

Розв'яжемо ще одне додаткове завдання. Нехай права частина рівняння Пуассона $\Delta u = f(x,y)$ має вигляд $f(x,y) = b + c \cdot g(x,y)$, де $g(x,y)$ є пірамідальною функцією $g(x,y) = \frac{1}{2}\left(2 - |x| - |y| - ||y| - |x||\right)$, яка дорівнює нулю на контурі квадрата. Задача полягає в підборі значень параметрів b і c таких, що середнє значення розв'язку по квадрату дорівнює a і, крім того, $u(x_0, y_0) = a$. В такій постановці ми маємо дві додаткові умови і два шуканих параметра, які можна знайти з використанням допоміжного вузла Global Equations.

Продовжить роботу з поточним файлом. Спочатку створіть функцію $g(x,y)$.

Global Definitions : RMB > Functions – Analytic

Function name = g

Definition – **Expression** =

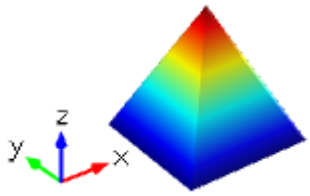
(2-abs(x)-abs(y)-abs(abs(y)-abs(x)))/2

– **Arguments** = x, y

Plot Parameters – **<x> Lower limit, Upper limit** = -1, 1

– **<y> Lower limit, Upper limit** = -1, 1

LMB > Plot



Потім змініть праву частину рівняння Пуассона.

Poisson's Equation (poeq) - Poisson's Equation 1:

Source Term – **f** = b+c*g(x,y)

Задайте умови у вузлі **Global Equations 1** (видаливши попередню умову).

Poisson's Equation (poeq) - Global Equations 1:

Global Equations – **Name, f(u,ut,utt,t)** = c, a-aveop1(u)

– **Name, f(u,ut,utt,t)** = b, a-at2(x0,y0,u)

Перша умова накладає обмеження на середнє значення, друга – представляє умову $u(x_0, y_0) = a$.

Переобчисліть модель.

Home -> Compute

Створіть вузли для обчислення величин b і c, і надрукуйте їх значення.

Derived Values - State variable f: RMB > Duplicate

Label = Variable b

Expression – **Expression** = b

RMB > Evaluate – New Table

Derived Values - State variable f: RMB > Duplicate

Label = Variable c

Expression – **Expression** = c

RMB > Evaluate – Table 4

State variable b	State variable c
39.396	-68.070

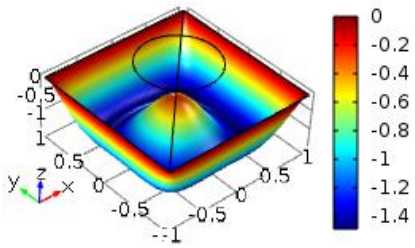
Для наочності побудуйте графік розв'язку рівняння Пуассона u.

2D Plot Group 1 - Surface 1: RMB > Height Expression

Surface 1 - Height Expression 1:

Axis – **Scale factor** = True,0.5

LMB > Plot



Зауваження. Задачі, в яких шукана величина стоятиме в аргументі оператора, будуть призводити до помилок. Наприклад, при заданому f задача пошуку абсциси x точки, в якій $u(x,x) = a$, не буде розв'язана, бо потребує використання рівняння a-at2(X,X,u)=0 відносно невідомої X. Це пов'язано з тим, що перед викликом операторів at(...) координата точки повинна бути вже обчисленою. Тобто деякі аргументи операторів не можуть бути шуканими величинами! Це зауваження стосується і функцій – функції повинні бути повністю визначені до їх першого використання. Але вирази змінних у вузлах Variables можуть містити шукані величини.

Вправа. В тривимірних моделях існують оператори ballint(r,expr), ballavg(r,expr), sphint(r,expr), sphavg(r,expr), які обчислюють об'ємні та поверхневі інтеграли, а також середні значення по кулі і сфері. Для обчислення значення в точці призначено оператор at3(x0,y0,z0,expr).

Нехай в кубі $\Omega = [-1,1]\times[-1,1]\times[-1,1]$ дано рівняння Пуассона $\Delta u = f$ з нульовими граничними умовами. Відомо, що f – стала, але така, що:

1. Інтеграл від розв'язку u по сфері радіуса r з центром в точці (x_0, y_0, z_0) дорівнює заданій величині a.
2. Інтеграл від розв'язку u по кулі радіуса r з центром в точці (x_0, y_0, z_0) дорівнює заданій величині a.
3. Середнє значення розв'язку u по сфері радіуса r з центром в точці (x_0, y_0, z_0) дорівнює заданій величині a.
4. Середнє значення розв'язку u по кулі радіуса r з центром в точці (x_0, y_0, z_0) дорівнює заданій величині a.
5. Значення розв'язку u в точці (x_0, y_0, z_0) дорівнює заданій величині a. (тобто $u(x_0, y_0, z_0) = a$).
6. Середнє значення розв'язку u по області куба дорівнює заданій величині a.

Для кожної з умов знайти f і u.

4.2. Використання Mathematical Particle Tracing інтерфейсу.

Для розв'язання систем ЗДР в COMSOL Multiphysics можна використовувати інтерфейс `Mathematical Particle Tracing` (не плутати з іншими `Particle Tracing` інтерфейсами). Головне його призначення полягає в побудові траєкторій руху множини частинок. Його можна використовувати і для інших цілей. Він має кілька редакцій, але для наших цілей цікаві дві.

Редакція `Massless` (безінерційний, безмасовий) використовується для розв'язання систем ЗДР першого порядку

$$\frac{d\,\boldsymbol{q}}{d\,t} = \boldsymbol{v}, \tag{1}$$

де **q** – вектор координат частинок, **v** – вектор їх швидкостей. В цій постановці задається векторне поле швидкостей **v**, і для кожної частинки відомо її початкове положення $\boldsymbol{q}(t_0)$. В процесі обчислень визначаються траєкторії руху $\boldsymbol{q}(t)$ частинок.

В редакції `Newtonian` розв'язується система ЗДР другого порядку

$$\frac{d}{dt}\left(m_p\,\mathbf{v}\right) = \mathbf{F}, \tag{2}$$

де **v** – вектор швидкостей, **F** – вектор сил.

В обох випадках ми повинні обрати підходящу просторову розмірність задачі (2D або 3D) і працювати з динамічним (`Time Dependent`) режимом дослідження. При цьому треба задати геометрію області, в якій відбувається рух, так, щоб всі траєкторії не виходили з цієї зони.

Приклад 1. Розв'язати систему трьох диференціальних рівнянь.

$$\begin{cases} x'_t = y - z & , x(0) = 1 \\ y'_t = z - x & , y(0) = 0 \\ z' = x - 2 \cdot y & , z(0) = 2 \end{cases}$$

Цю систему ми вже досліджували в прикладі 4.1.2.10 (параграф 4.1.2, приклад 10) з використанням динамічного Global ODEs and DAEs інтерфейсу. Перепозначивши координатні функції x, y, z через q_x, q_y, q_z, систему рівнянь можна записати у вигляді

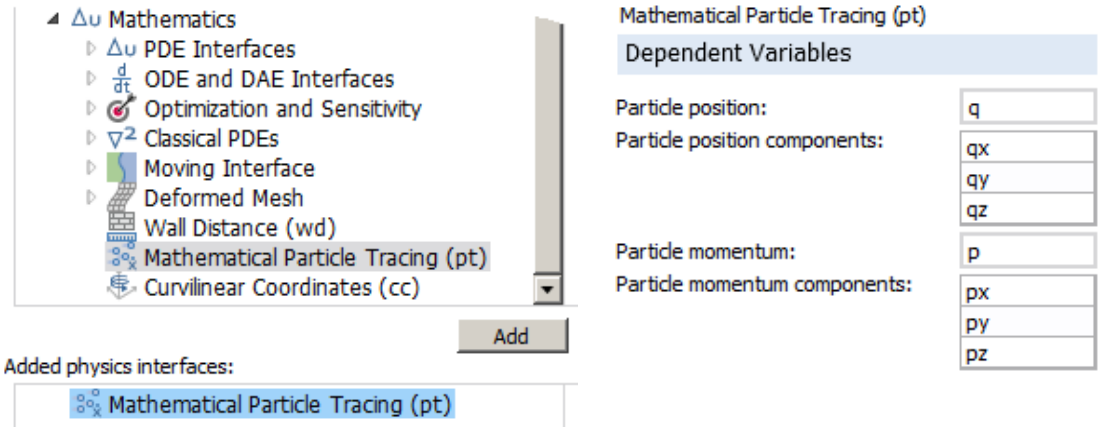
$$\frac{d}{dt}\begin{pmatrix} q_x \\ q_y \\ q_z \end{pmatrix} = \begin{pmatrix} q_y - q_z \\ q_z - q_x \\ q_x - 2q_y \end{pmatrix},$$

Якщо ввести вектори $\boldsymbol{q} = (q_x, q_y, q_z)$ і $\boldsymbol{v} = (q_y - q_z, q_z - q_x, q_x - 2q_y)$, то задача набуває вигляду (1), і її можна розв'язувати з застосуванням інтерфейсу `Particle Tracing`.

Розв'язання. Створіть динамічну 3D модель задачі з `Mathematical Particle Tracing` інтерфейсом.

File -> New -> Model Wizard -> 3D -> Mathematics -
Mathematical Particle Tracing (pt) -> Time Dependent

Обравши у вікні `Select Physics` тип моделі `Mathematical Particle Tracing(pt)` ви побачите в правій частині вікна поля для введення ідентифікаторів змінних задачі. Ці імена пізніше будуть використовуватися під час запису системи диференціальних рівнянь. Залиште вміст цих полів без змін.



Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Розрахунковою областю задачі вважатимемо циліндр. Він представлятиме зону, в якій буде розташована фазова траєкторія. Щоб не замислюватися про можливість виходу кривої за межі цієї області, для циліндра треба задати досить великі розміри.

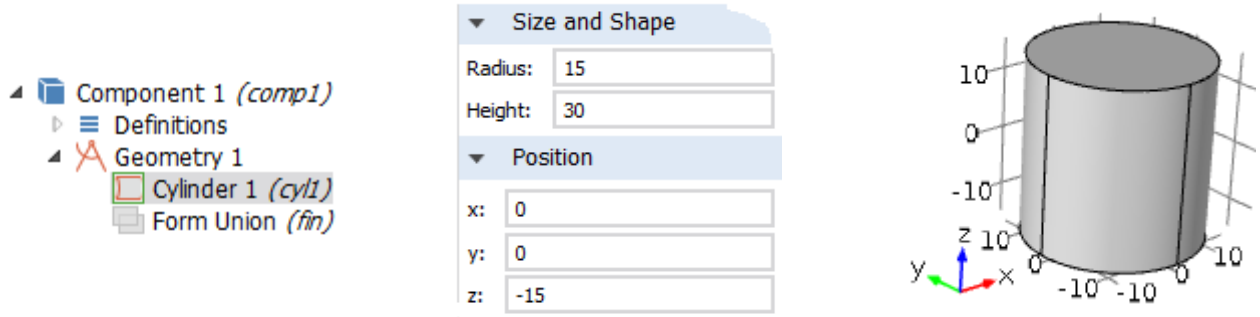
`Geometry` **-> `Primitives`** – Cylinder

Component 1 - Geometry 1 - Cylinder1:

`Size and Shape` – **Radius, Height** = `15, 30`

Position – **x, y, z** = 0,0,-15

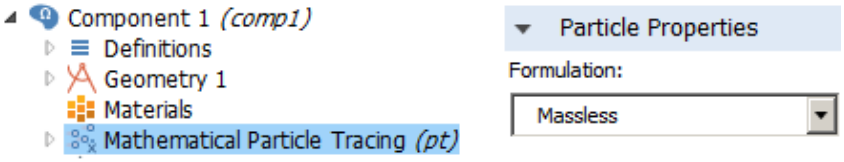
LMB > Build All Objects



В ПВ вузла **Mathematical Particle Tracing (pt)** оберіть Massless (безмасову) редакцію інтерфейсу. Також перевірте, що в розділі **Domain Selection** стоїть одиниця, тобто обрана розрахункова область з індикатором 1 (це циліндр і інших областей у нас немає).

Component 1 - Mathematical Particle Tracing (pt):

Particle Properties – **Formulation** = Massless



В розділі **Dependent Variable** ви ще раз можете проконтролювати імена шуканих величин.

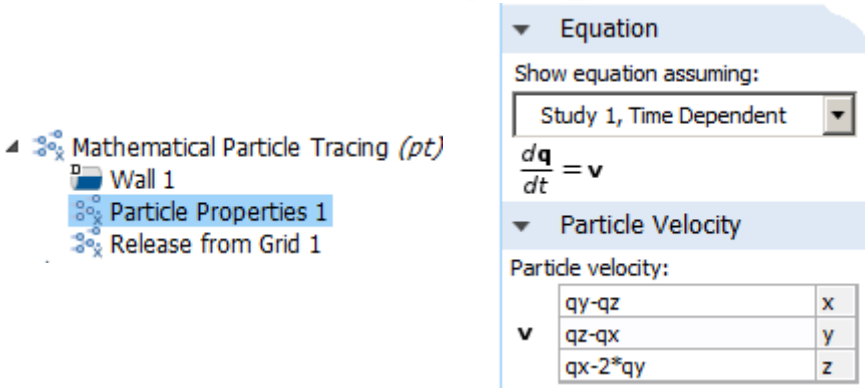
В ПВ вузла **Mathematical Particle Tracing (pt) - Particle Properties 1** в розділі **Particle Velocity** задайте праві частини системи рівнянь. Для цього використайте ідентифікатори qx, qy і qz як імена шуканих величин.

Mathematical Particle Tracing (pt) – Particle Properties 1:

Particle Velocity – **v <x>** = qy-qz

– **v <y>** = qz-qx

– **v <z>** = qx-2*qy

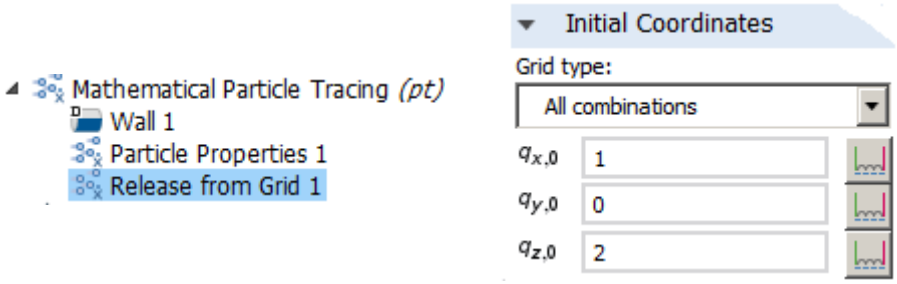


До гілки **Mathematical Particle Tracing (pt)** додайте вузол Release from Grid. Він використовується для введення початкових положень рухомих частинок. В нашій моделі положення єдиної частинки будуть представляти початкові умови системи рівнянь.

Mathematical Particle Tracing (pt) : RMB > Release from Grid

Mathematical Particle Tracing (pt) – Release from Grid 1:

Initial Coordanates – $q_{x,0}$ = 1
– $q_{y,0}$ = 0
– $q_{z,0}$ = 2



Задайте діапазон зміни параметра часу `t`.

Step 1: Time Dependent:

Study Settings – **Times** = `range(0,0.1,20)`

Виконайте обчислення.

`Home` -> Compute

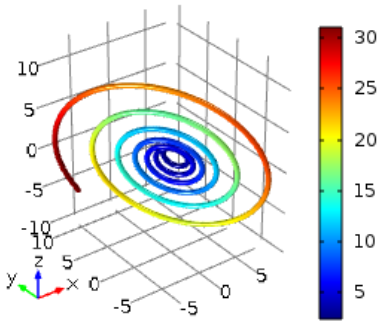
В гілці **Results - Data Sets** з'явиться вузол **Particle 1**, який представлятиме джерело даних типу `Particle`. Також в гілці **Results** з'явиться графічна група типу `3D Plot Group` з назвою **Particle Trajectories (pt)**. Зверніть увагу, що в її ПВ в списку **Data Set** обрано джерело даних **Particle 1**. Перевірте, що в розділі **Data** в полі **Time** обрано максимальний момент часу (в нашому прикладі 20), і зніміть перемикач **Plot data set edges** ☐ Plot data set edges , щоб на графіку не відображався циліндр.

Графік типу `Particle Trajectories`, який побудовано в створеній графічній групі, зазвичай відображає траєкторії частинок. В нашому випадку однієї частинки крива представлятиме фазову траєкторію системи рівнянь. Налаштуйте її стиль.

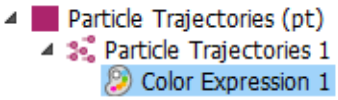
Particle Trajectories (pt) - Particle Trajectories 1:

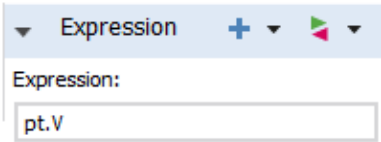
Coloring and Style – **Line style** – **Type** =`Tube`
– **Line style** – **Tube radius expression** =`0.3`
– **Line style** – **Radius scale factor** =`True,1`
– **Point style** – **Type** = `None`

LMB > Plot



Колірна палітра справа відображає величину (модуль) швидкості, з якою точка рухається по фазовій траєкторії. Це визначається налаштуваннями в ПВ вузла **Particle Trajectories 1 - Color Expression 1**.





Тут префікс `pt` є ім'ям моделі і задається в ПВ вузла **Mathematical Particle Tracing (pt)** в полі **Name**. А `V` позначає величину (модуль вектора) швидкості.

Зауваження. Якщо розміри розрахункової області (циліндра) будуть недостатніми, то траєкторія точки може перетнутися з поверхнею циліндра і, взаємодіючи з ним, змінити форму. Це призведе до неправильного розв'язку.

Щоб зобразити графіки розв'язків `qx`, `qy`, `qz` в залежності від часу, створіть графічну гілку **1D Plot Group 2** з трьома вузлами типу `Particle` для кожної невідомої.

`Results` –> Plot Group - 1D Plot Group

1D Plot Group 2 :

`Data` – **Data set** = `Particle 1`

`Legend` – **Position** = `Upper left`

`1D Plot Group 2` –> Add Plot – Particle

1D Plot Group 2 - Particle 1:

`y-Axis Data` – **Expression** = `qx`

`Coloring and Style` –**`Line style`** –**Width** =3

`Legends` – **Show Legends, Legends, Legends** = `True,Manual,qx`

1D Plot Group 2 - Particle 1: RMB > Duplicate

1D Plot Group 2 - Particle 2:

`y-Axis Data` – **Expression** = `qy`

`Legends` – **Show Legends, Legends, Legends** = `True,Manual,qy`

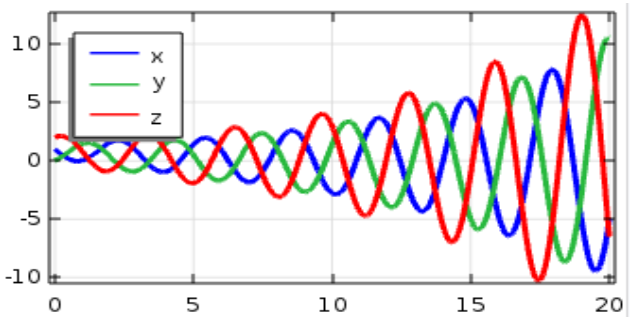
1D Plot Group 2 - Particle 2: RMB > Duplicate

1D Plot Group 2 - Particle 3:

`y-Axis Data` – **Expression** = `qz`

`Legends` – **Show Legends, Legends, Legends** = `True,Manual,qz`

LMB > Plot



Побудовані криві збігаються з графіками функцій, отриманими нами в прикладі 4.1.2.10. ■

Розглянемо ще один приклад застосування безмасової редакції інтерфейсу `Mathematical Particle Tracing`.

Приклад 2. Розв'яжемо систему трьох диференціальних рівнянь першого порядку (атрактор Ресслера).

$$\begin{cases} x_t' = -y - z \\ y_t' = x + a \cdot y \\ z_t' = b - z \cdot (c - x) \end{cases},$$

в якій покладемо $a = 0.425$, $b = 2$, $c = 4$, і візьмемо наступні початкові умови

$$x(0) = -0.04, \quad y(0) = -0.03, \quad z(0) = 0.52.$$

Вигляд системи співпадає з (1) і її можна розв'язувати з застосуванням інтерфейсу `Mathematical Particle Tracing`.

Розв'язання. Створіть безрозмірну динамічну 3D модель задачі з `Mathematical Particle Tracing` інтерфейсом.

File -> New -> Model Wizard -> 3D -> Mathematics - Mathematical Particle Tracing (pt) -> Time Dependent

Untitled.mph (root): `Unit System` – <> = None

Розрахунковою областю задачі вважатимемо циліндр.

`Geometry` -> **`Primitives`** – Cylinder

Component 1 - Geometry 1 - Cylinder1:

`Size and Shape` – **Radius, Height** = `10,10`

`Position` – **x, y, z** = `0,0,-2`

LMB > Build All Objects

В ПВ вузла **Mathematical Particle Tracing (pt)** оберіть `Massless` (безмасову) редакцію інтерфейсу.

Component 1 - Mathematical Particle Tracing (pt):

`Particle Properties` – **Formulation** = `Massless`

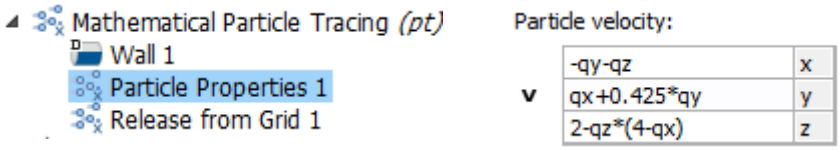
В ПВ вузла **Mathematical Particle Tracing (pt) - Particle Properties 1** в розділі **`Particle Velocity`** задайте праві частини системи рівнянь. Для цього використайте ідентифікатори `qx`, `qy` і `qz` як імена шуканих величин.

Mathematical Particle Tracing (pt) – Particle Properties 1:

`Particle Velocity` – **v <x>** = `-qy-qz`

– **v <y>** = `qx+0.425*qy`

– **v <z>** = `2-qz*(4-qx)`



В гілку **Mathematical Particle Tracing (pt)** додайте вузол `Release from Grid`, в ПВ якого введіть початкові значення.

Mathematical Particle Tracing (pt) : RMB > Release from Grid

Mathematical Particle Tracing (pt) – Release from Grid 1:

`Initial Coordanates` – $\mathbf{q_{x,0}}$ = `-0.04`

– $\mathbf{q_{y,0}}$ = `-0.3`

– $\mathbf{q_{z,0}}$ = `0.52`

Задайте діапазон зміни параметра часу `t`.

Step 1: Time Dependent:

Study Settings – **Times** = range(0,0.1,25)

Виконайте обчислення.

Home -> Compute

В гілці **Results - Data Sets** з'явиться вузол джерела даних **Particle 1**, а також графічна група **Particle Trajectories (pt)**, в ПВ якої в списку **Data Set** обрано це джерело даних **Particle 1**. Зніміть перемикач **Plot data set edges**, щоб на графіку не відображався циліндр.

Нагадаємо, що графік типу Particle Trajectories, який побудовано в створеній графічній групі призначений для відображення траєкторій частинок. В нашому випадку крива представлятиме фазову траєкторію системи рівнянь. Налаштуйте її стиль.

Particle Trajectories (pt) - Particle Trajectories 1:

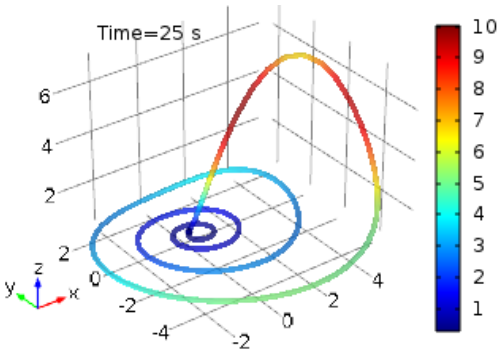
Coloring and Style – **Line style** – **Type** =Tube

– **Line style** – **Tube radius expression** =0.1

– **Line style** – **Radius scale factor** =True,1

– **Point style** – **Type** = None

LMB > Plot



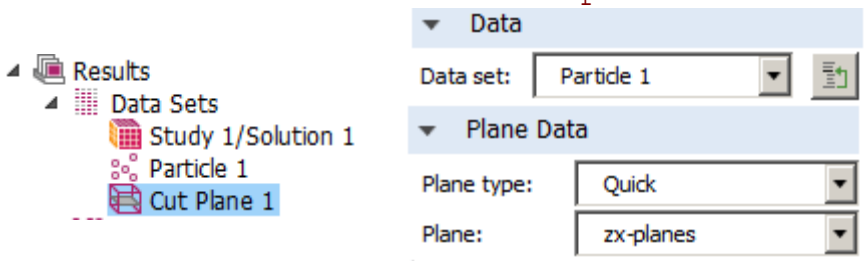
Побудуємо рисунок, на якому зобразимо точки перетину фазової траєкторії з деякою фіксованою площиною. Це називається перетином Пуанкаре. Для цього спочатку в гілці **Results - Data Sets** створіть вузол типу Cut Plane.

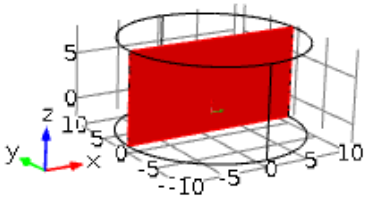
Results - Data Sets: RMB > Cut Plane

Results - Data Sets - Cut Plane:

Data – **Data set** = Particle 1

Plane Data – **Plane** = zx – planes





Щоб не пошкодити попередній графік, дублюємо його графічну групу і створимо в ній додаткові вузли.

Results - Particle Trajectories (pt): RMB > Duplicate

Results - Particle Trajectories (pt) 1: RMB > Surface

Results - Particle Trajectories (pt) 1- Surface:

Data – **Data set** = Cut Plane 1

Coloring and Style – **Coloring** = Uniform

Coloring and Style – **Color** = Gray

LMB > Plot

У лівому верхньому куті графічного вікна натисніть кнопку — Transparency. Після цього додайте до графічної групи «карту Пуанкаре» і налаштуйте її.

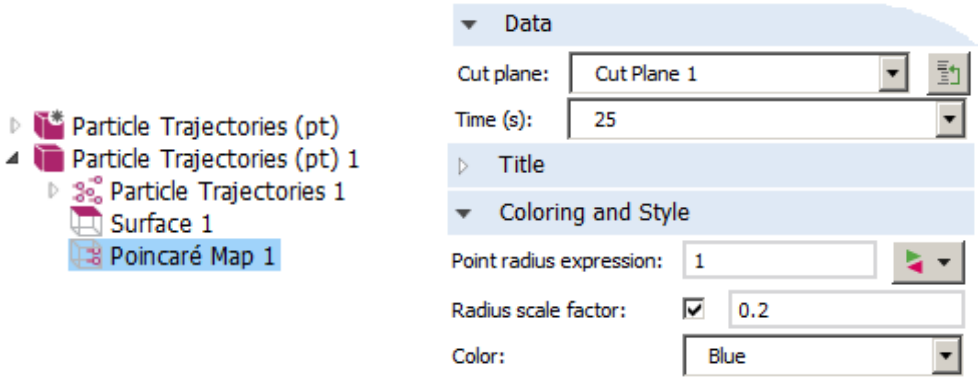
Results - Particle Trajectories (pt) 1: RMB > More Plots - Poincare Map

Data – **Cut Plane** = Cut Plane 1

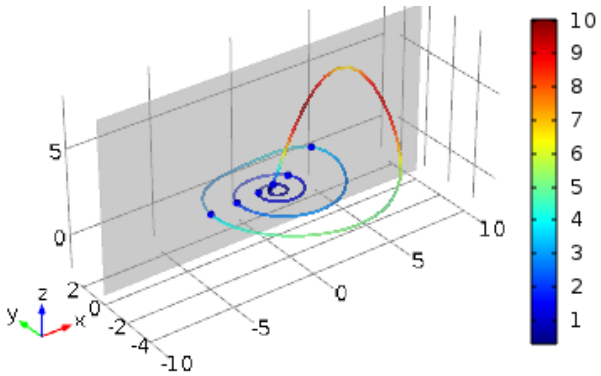
Coloring and Style – **Point radius expression** =1

– **Radius scale factor** =True,0.2

– **Color**= Blue



LMB > Plot



На цьому графіку показані точки перетину фазової траєкторії з площиною ZX.

Цікаво побудувати графік швидкості руху точки вздовж фазової траєкторії в залежності від часу. Для цього в гілку **Results** додайте одновимірну графічну групу з вузлом типу Particle і виразом pt.V.

Results **-> Plot Group** – 1D Plot Group

1D Plot Group 3 **-> Add Plot** – Particle

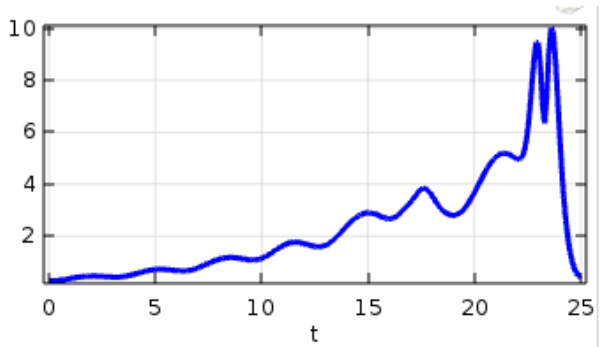
1D Plot Group 3 - Particle 1:

Data – **Data set** = Particle 1

y-Axis Data – **Expression** = pt.V

Coloring and Style –**Line style** –**Width** =3

LMB > Plot



Вправа. Розв'язати систему ЗДР

$$\frac{dx}{dt} = -y + x \cdot \left(1 - x^2 - y^2\right)$$

$$\frac{dy}{dt} = x + y \cdot \left(1 - x^2 - y^2\right)$$

з кількома різними початковими умовами. Побудувати для них фазові траєкторії.

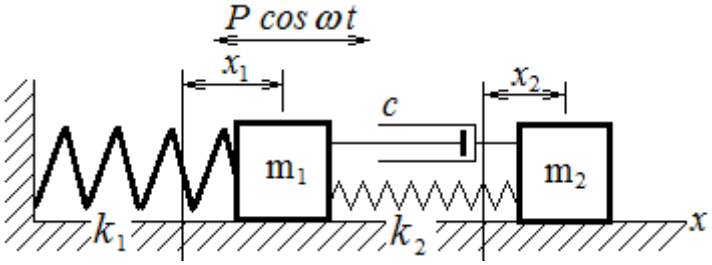
Вказівка. У системи існує граничний цикл $x^2 + y^2 = 1$, до якого всі інші фазові траєкторії наближаються по спіралі.

Продемонструємо використання `Newtonian` редакції `Mathematical Particle Tracing` інтерфейсу.

Приклад 3. Розв'язати систему диференціальних рівнянь

$$\begin{aligned} m_1 \ddot{x}_1 &= -k_1 x_1 + k_2 (x_2 - x_1) + c(\dot{x}_2 - \dot{x}_1) + P\cos\omega t \\ m_2 \ddot{x}_2 &= -k_2 (x_2 - x_1) - c(\dot{x}_2 - \dot{x}_1) \end{aligned},$$

з початковими умовами $x_1(0) = 0, x_2(0) = 0, \dot{x}_1(0) = 1, \dot{x}_2(0) = 1$. Ця система описує рух динамічного погашувача коливань [5], схематично зображеного на наступному рисунку.



Для модельної задачі оберати наступні значення параметрів: $m_1 = 3, m_2 = 2$, $k_1 = 1, k_2 = 0.25$, $c = 0.5$, $P = 2, \omega = \pi/4$.

Якщо покласти $\boldsymbol{v} = (\dot{x}_1, \dot{x}_2)$, поділити перше рівняння на m_1 і друге – на m_2, то система набуде вигляду (2), і її можна буде розв'язувати з застосуванням `Newtonian` редакції `Mathematical Particle Tracing` інтерфейсу.

Розв'язання. Створіть безрозмірну динамічну 2D модель задачі з `Mathematical Particle Tracing` інтерфейсом.

File -> New -> Model Wizard -> 2D -> Mathematics - Mathematical Particle Tracing (pt) -> Time Dependent

Untitled.mph (root): `Unit System` – <> = None

В гілці **Global Definitions** створіть вузол **Parameters** і в таблиці задайте параметри задачі.

Global Definitions : RMB > Parameters

Parameters – **Name, Expression** = m1,3
– **Name, Expression** = m2,1
– **Name, Expression** = k1,1
– **Name, Expression** = k2,0.25
– **Name, Expression** = c,0.5
– **Name, Expression** = P,2
– **Name, Expression** = w,pi/2

Розрахунковою областю задачі вважатимемо квадрат.

Geometry **-> Primitives** – Square

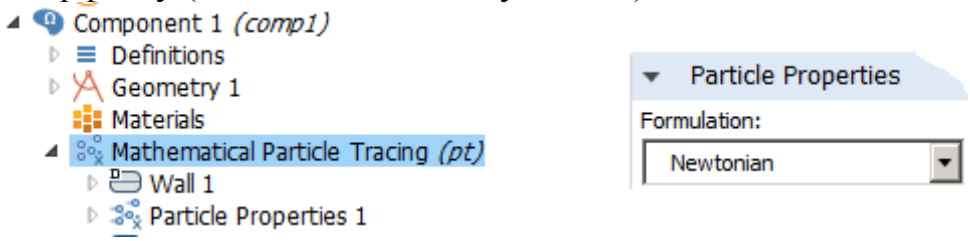
Component 1 - Geometry 1 - Square 1:

Size – **Size length** = 6

Position – **Base, x, y** = Center,0,0

LMB > Build All Objects

Перевірте, що в ПВ вузла **Mathematical Particle Tracing (pt)** обрана Newtonian редакція інтерфейсу (значення за замовчуванням).



У вузлі **Particle Properties 1** задайте одиничне значення маси.

Mathematical Particle Tracing (pt) – Particle Properties 1:

Particle Mass – $\mathbf{m_p}$ = 1

В нашій задачі безпосередньо маси будуть використані в правих частинах рівнянь, де також будемо вважати, що $(x_1, x_2) = (qx, qy)$.

В гілку **Mathematical Particle Tracing (pt)** додайте вузол Release from Grid, в ПВ якого введіть початкові значення.

Mathematical Particle Tracing (pt) : RMB > Release from Grid

Mathematical Particle Tracing (pt) – Release from Grid 1:

Initial Coordanates – $\mathbf{q_{x,0}}$ = 0
– $\mathbf{q_{y,0}}$ = 0

Initial velocity – $\mathbf{v_o}$ **<x>**= 1
– $\mathbf{v_o}$ **<y>**= 1

В гілку **Mathematical Particle Tracing (pt)** додайте вузол Force і в його ПВ введіть вирази правих частин системи.

Mathematical Particle Tracing (pt) : RMB > Force

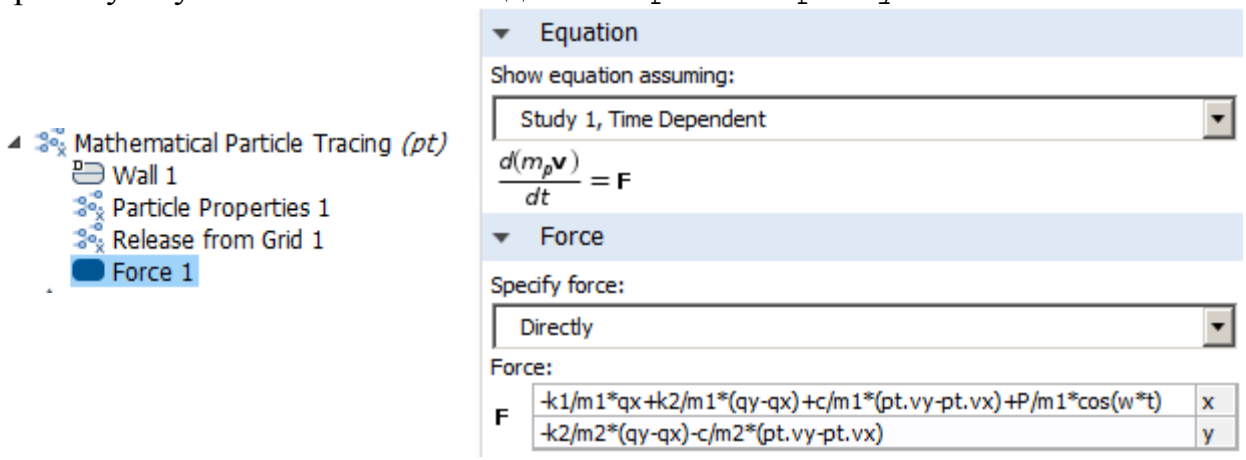
Mathematical Particle Tracing (pt) – Force 1:

Domain Selection – **Selection** = Manual, 1

Force – **Force: F <x>** = -k1/m1*qx+
k2/m1*(qy-qx)+c/m1*(pt.vy-pt.vx)+P/m1*cos(w*t)

Force – **Force: F <y>** = `-k2/m2*(qy-qx)-c/m2*(pt.vy-pt.vx)`

Зверніть увагу на позначення швидкостей `pt.vx` і `pt.vy`.



Задайте діапазон зміни параметра часу `t`.

Step 1: Time Dependent:

Study Settings – **Times** = `range(0,0.1,50)`

Виконайте обчислення.

`Home` -> Compute

В гілці **Results** з'явиться вузол джерела даних **Data Sets - Particle 1**, а також графічна група **Particle Trajectories (pt)**. Для нашого прикладу вона непотрібна, тому видалимо її.

Results - Particle Trajectories (pt): RMB > Delete

Створіть одновимірну графічну групу для побудови графіків розв'язків `qx` і `qy` (тобто x_1 і x_2).

`Results` -> **Plot Group** – 1D Plot Group

`1D Plot Group 1` -> **Add Plot** – Particle

1D Plot Group 1 - Particle 1:

Data – **Data set** = `Particle 1`

y-Axis Data – **Expression** = `qx`

Coloring and Style —**Width** =3

Legends – **Show legends, Legends, Legends** = `True, Manual,qx`

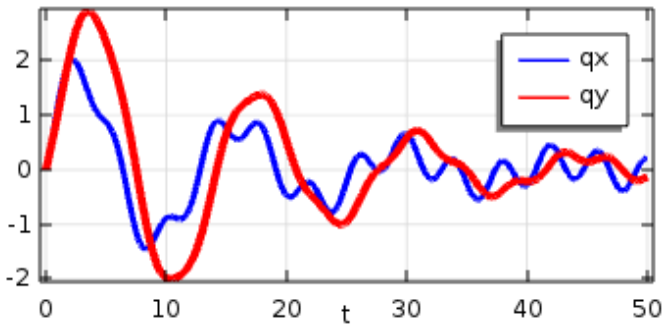
1D Plot Group 1 - Particle 1: RMB > Duplicate

1D Plot Group 1 - Particle 2:

y-Axis Data – **Expression** = `qy`

Coloring and Style —**Color** =`Red`

LMB > Plot



Зауваження. Для поточної задачі використання інтерфейсу `Mathematical Particle Tracing` є штучним. Але для деяких задач розв'язання системи ЗДР в цьому інтерфейсі виглядає більш природним і простішим.

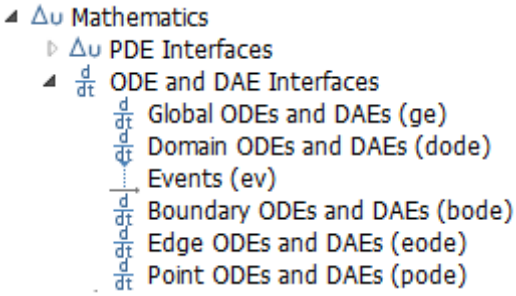
4.3. Використання Domain ODEs and DAEs інтерфейсу.

`Domain ODEs and DAEs` інтерфейс призначений для розв'язання алгебраїчних та звичайні диференціальні рівнянь, які описують неперервні моделі без просторової взаємодії, тобто такі рівняння і системи, в які просторові координати входять як параметри (по ним немає диференціювання). Це не нульвимірні задачі, оскільки їх доводиться розв'язувати в кожній точці розрахункової області. В термінології розробників моделі, які зводяться до подібних задач, називаються розподіленими. Їх математичне формулювання має наступний вигляд

$$e_a \frac{\partial^2 \boldsymbol{u}}{\partial t^2} + d_a \frac{\partial \boldsymbol{u}}{\partial t} = \boldsymbol{f}(t, x, y, z, \boldsymbol{u}),$$

де права частина $\boldsymbol{f}$ може залежати від часу t, просторових координат і шуканого розв'язку $\boldsymbol{u}$. Коефіцієнти e_a, d_a також можуть залежати від просторових координат.

Для моделювання таких задач в COMSOL Multiphysics призначені інтерфейси: `Domain ODEs and DAEs`, `Boundary ODEs and DAEs`, `Edge ODEs and DAEs` і `Point ODEs and DAEs`. На кроці `Select Physics` побудови моделі ці режими вибираються в гілці `Mathematics - ODEs and DAEs Interfaces`.



Згадані інтерфейси надають можливість розв'язувати розподілені рівняння в областях, на поверхнях тіл, на ребрах і в точках.

В розподілених моделях стан точки системи не залежить від стану сусідніх точок і, як наслідок, немає впливу граничних умов на внутрішні точки тіла. Однак в подібних моделях початковий стан (початкова умова) може залежати від просторових координат $u(\boldsymbol{x},0) = u_0(\boldsymbol{x})$. Крім того, параметри системи можуть безпосередньо або за посередництвом фізичних величин залежати від координат. Наприклад на швидкість хімічної реакції може впливати температура $T(x, y, z)$, яка залежить від точки (x, y, z).

4.3.1. Розподілені алгебраїчні задачі.

Розподілена задача може виявитися суто алгебраїчною, коли не містить диференціювання по часу.

Приклад 1. Дослідимо модель, в якій густина газу ρ позв'язана з тиском p через алгебраїчне рівняння

$$\rho^2(1-\alpha\rho)-\rho(\beta p+\gamma)+\delta p=0. \qquad (1)$$

Нехай процес розгортається в квадраті $[0,1]\times[0,1]$ і тиск визначається за формулою $p(x,y)=1+x^2-y^2/2$. Розглянемо безрозмірну модель і покладемо параметри рівняння $\alpha=3,\ \beta=1.5,\ \gamma=2.5,\ \delta=2$.

Р о з в ' я з а н н я . Створіть модель послідовністю команд

File -> New -> Model Wizard -> 2D -> Mathematics -
ODE and DAE Interfaces - Domain ODEs and DAEs(dode)-> Stationary

Для розв'язання алгебраїчного рівняння потрібно обрати стаціонарне (`Stationary`) дослідження (Study). Це одночасно приводить до вибору метода обчислення: метода Ньютона – Рафсона з автоматичним вибором коефіцієнта вгамування. Він підходить для більшості задач пошуку коренів алгебраїчних рівнянь.

Щоб розв'язувати безрозмірні рівняння, вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

В гілці **Global Definition**s створіть вузол **Parameters**

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `a,3`
– **Name, Expression** = `b,1.5`
– **Name, Expression** = `g,2.5`
– **Name, Expression** = `d,2`

В таблиці локальних змінних задайте тиск $p(x,y)$.

`Home` -> **`Definitions`** – Variables – Local Variables

Component 1 – Definitions – Variables 1:

`Variables(таблиця)` – **Name** = `p`
– **Expression** = `1+x^2-y^2/2`

Геометрія моделі складатиметься з одного квадрата.

`Geometry`->**`Pimitives`**-> Square

За замовчуванням параметри квадрата збігаються з тими, які нам потрібні, тому завершіть креслення геометрії клацанням по кнопці Build All Objects.

Побудуйте СкЕ сітку з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Відкрийте ПВ вузла **Domain ODEs and DAEs (dode) - Distributed ODE 1**. В розділі **`Equation`** цієї панелі наведено загальний вигляд розподіленого рівняння

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} = f,$$

де u є ім'ям шуканої змінної (в нашому випадку густини). Оскільки задача стаціонарна, то покладіть $e_a = 0, d_a = 0$, але це необов'язково, бо стаціонарний солвер (розв'язувач) все одно не буде приймати до уваги похідні по t. Отже співвідношення матиме вид $0 = f$. Таким чином, введіть наступні дані:

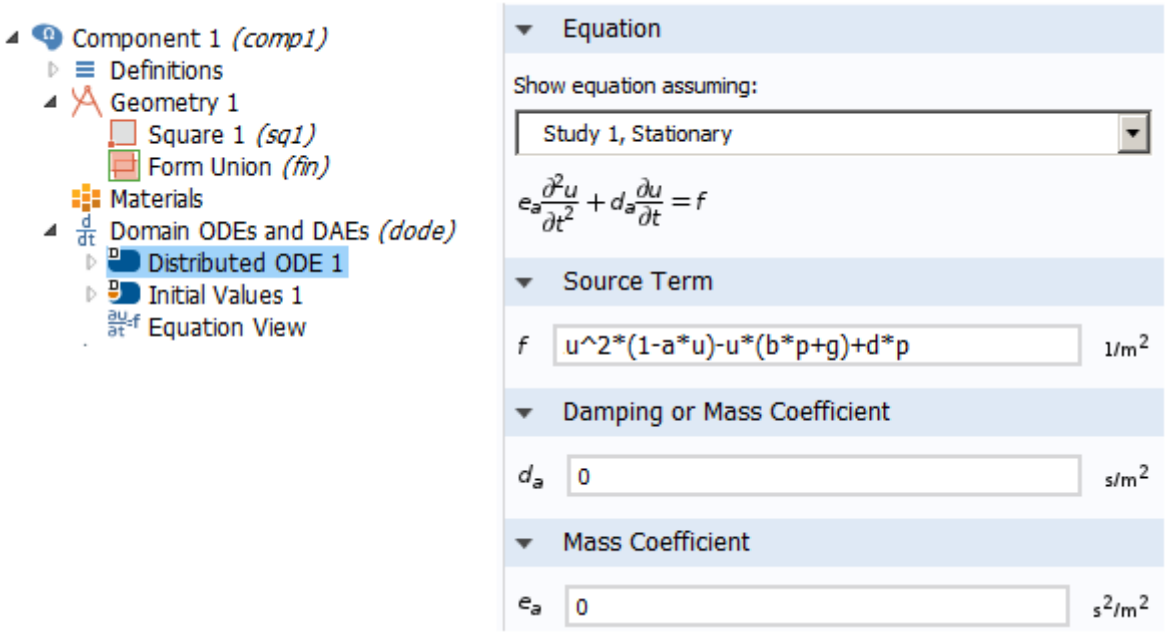
Domain ODEs and DAEs(dode) - Distributed ODE 1:

Source Term – **f** = `u^2*(1-a*u)-u*(b*p+g)+d*p`

Damping or Mass coefficient – $\mathbf{d_a}$ = 0

Mass coefficient – $\mathbf{e_a}$ = 0

Розділ **Equation** цього ВВ набуде вигляду



Оскільки похідні по просторовим координатам в рівняння не входять, то задавати граничні умови непотрібно. Початкові значення можуть вплинути на розв'язок, оскільки вони використовуються як початкові наближення в алгоритмі Ньютона. Ми залишимо їх незмінними, тобто нульовими.

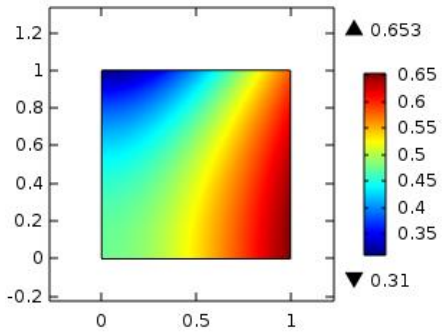
Виконайте обчислення.

`Home` -> Compute

В гілці **Results** з'явиться двовимірна графічна група **2D Plot Group 1** з вузлом **Surface 1**, що містить кольоровий графік густини.

2D Plot Group 1:

Color Legend - Show maximum and minimum values = `True`



■

В останньому прикладі ми розв'язали кубічне рівняння, яке може мати три дійсні корені (в кожній точці). Виникає питання, поле якого кореня ми отримали. Це складне питання, і на нього важко дати просту відповідь. Пояснемо проблему на простішій задачі, коли модель зводиться до обчислення поля розв'язків квадратного рівняння.

Приклад 2. Дослідимо модель, в якій в квадраті $[0,1]\times[0,1]$ розв'язується розподілене алгебраїчне рівняння $u^2-2au+a^2-p(x,y)=0$. Його розв'язком є $u=a\pm\sqrt{p(x,y)}$. Нехай змінна p визначається за формулою $p(x,y)=x^2+y^2$. Покладемо параметр рівняння $a=1$ і будемо розглядати безрозмірну модель.

Р о з в ' я з а н н я . Створіть модель послідовністю команд:

File -> New -> Model Wizard -> 2D -> Mathematics - ODEs and DAEs Interfaces - Domain ODEs and DAEs(dode) -> Stationary

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

В гілці **Global Definition**s створіть вузол **Parameters**

Global Definitions : RMB > Parameters

Parameters – **Name, Expression** = a,1

Створіть таблицю локальних змінних зі змінною *p*.

Home -> **Definitions** – Variables – Local Variables

Component 1 – Definitions – Variables 1:

Variables (таблиця) – **Name** = p

– **Expression** = x^2+y^2

Геометрія моделі складається з одного квадрата.

Geometry -> **Pimitives** -> Square

LMB > Build All Objects

Побудуйте СкЕ сітку з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Налаштуйте коефіцієнти математичної моделі.

Domain ODEs and DAEs (dode) - Distributed ODE 1:

Source Term – **f** = u^2-2*a*u+a^2-p

Damping or Mass coefficient – $\mathbf{d_a}$ = 0

Mass coefficient – $\mathbf{e_a}$ = 0

Задайте початкові значення (вони використовуються солвером як початкові наближення до розв'язку).

Domain ODEs and DAEs (dode) - Initial Values 1:

Initial Values – **Initial value for u:** = 1.1

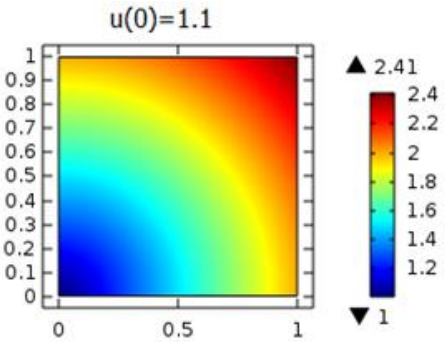
Виконайте обчислення.

Home->Compute

В гілці **Results** з'явиться гілка **2D Plot Group 1** з вузлом **Surface 1**, що містить кольоровий графік розв'язку.

2D Plot Group 1:

Color Legend - **Show maximum and minimum values** = True



Виконайте обчислення з іншим початковим значенням. Це можна зробити зміною значення u в ПВ **Domain ODEs and DAEs (dode) - Initial Values 1**, але для тренування ми зробимо це, додавши в ДМ інший вузол початкових умов і інший вузол дослідження `Study`.

Створіть додатковий вузол початкових умов з іншими значеннями.

Domain ODEs and DAEs (dode) : RMB > Initial Values

Domain ODEs and DAEs (dode) - Initial Values 2:

`Domain Selection` – **Selection** = `Manual, 1`

`Initial Values` – **Initial value for u:** = `0.9`

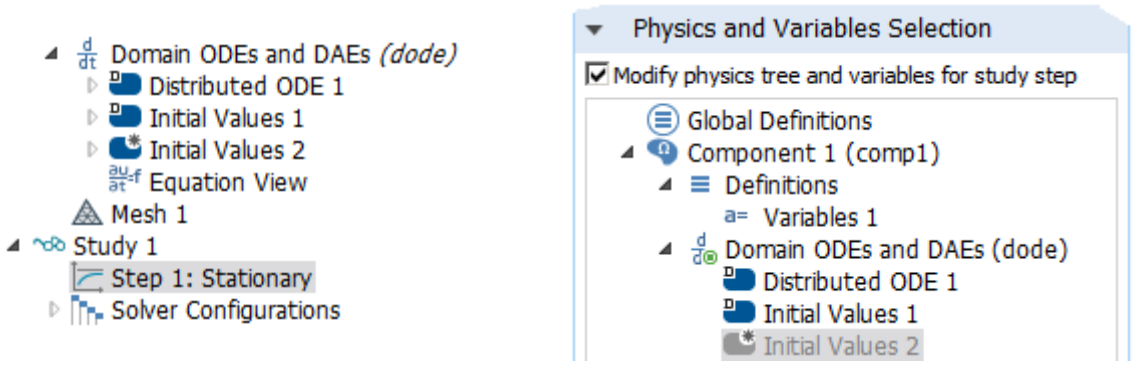
Якщо поглянути в ПВ вузла **Initial Values 1**, то побачите, що він перевантажений (overriden), тобто всі встановлені в ньому значення замінені іншими, які створені пізніше. Щоб обчислення у вузлі `Study 1` не змінилися, новий вузол **Initial Values 2** треба виключити з дерева моделі, яке «використовується» при дослідженні `Study 1`.

Study 1 – Step 1: Stationary:

`Physics and Variables Selection` –

Modify physics tree and variables for study step = `True`

В ДМ, яке відкриється нижче перемикача, виділіть вузол **Initial Values 2** і деактивуйте його, клацнувши по кнопці ⊘ Disable. Вузол стане тьмяним і виключеним з послідовності команд, які будуть виконуватися при дослідженні `Study 1`.



Тепер створіть другий вузол «досліджень» `Study`.

Untitled.mph (root) : RMB > Add Study = `Add Study ->`

Preset Studies - Stationary ->X

В ДМ з'явиться вузол **Study 2**. Обчислення, ініційовані цим вузлом, будуть використовувати початкові значення з вузла **Initial Values 2**, бо він перевантажує вузол **Initial Values 1**.

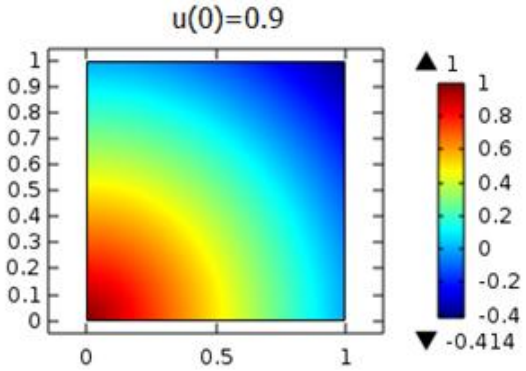
Виконайте друге дослідження.

Study 2 : RMB > Compute

В ДМ з'явиться двовимірна графічна група **2D Plot Group 2** з графічним вузлом **Surface 1**, що містить кольоровий графік розв'язку u, отриманого при початковому значенні $u_0 = 0.9$.

2D Plot Group 2:

`Color Legend` - **Show maximum and minimum values** = `True`



Точний розв'зок має вигляд $u = 1 \pm \sqrt{x^2 + y^2}$ (при $a = 1,\ p = x^2 + y^2$), тому в точці $(0,0)$ обидва розв'язки дорівнюють 1. Але в точці $(1,1)$ маємо $u_1(1,1) = 1 + \sqrt{2} \approx 2.414$ і $u_2(1,1) = 1 - \sqrt{2} \approx -0.414$. Зміна початкового значення повністю перебудувала поле кореня квадратного рівння.

Щоб порівняти отримані розв'язки, вирівняємо на них діапазони зміни кольорів.

2D Plot Group 1 - Surface 1 :

`Range` – **Manual color range** = `True`

– **Manual color range – Minimum, Maximum** = `-0.5, 2.5`

LMB > Plot (*наступний рисунок ліворуч*)

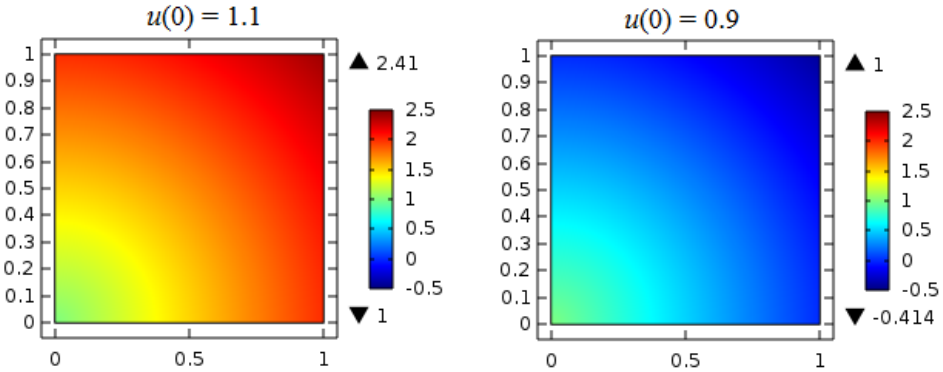
2D Plot Group 2 - Surface 1 :

`Range` – **Manual color range** = `True`

– **Manual color range – Minimum, Maximum** = `-0.5, 2.5`

LMB > Plot (*наступний рисунок праворуч*)

На наступному рисунку ці графіки наведені поряд.



Шкала кольорів на двох останніх графіках однакова. Початкові умови не дуже відрізняються. Але ліворуч побудовано поле розв'язку $u_1 \approx 1 + \sqrt{x^2 + y^2}$, а праворуч $u_2 \approx 1 - \sqrt{x^2 + y^2}$.

Можна задати неоднакові початкові умови в різних частинах розрахункової області. Тоді взагалі в них будуть обчислюватися різні гілки коренів алгебраїчного рівняння. Наприклад, додайте в нашу модель ще один вузол початкових умов (які будуть різнитися у верхній і нижній половинах квадрата).

Domain ODEs and DAEs (dode) : RMB > Initial Values

Domain ODEs and DAEs (dode) - Initial Values 3:

`Domain Selection` – **Selection** = `Manual, 1`

`Initial Values` – **Initial value for u:** = `0.9*(y<0.5)+1.1*(y>=0.5)`

Щоб перше і друге дослідження (`Study 1` і `Study 2`) не змінилися, виключіть з послідовності їх команд вузол **Initial Values 3**. Для цього в гілках досліджень **Study 1** і **Study 2** у вузлах **Step 1: Stationary** встановіть перемикач **Modify physics tree and variables for study step**, та в відповідних деревах деактивуйте вузол **Initial Values 3**, клацнувши по кнопці Disable.

Тепер додайте вузол третього дослідження і обчисліть його.

Untitled.mph (root) : RMB > Add Study = `Add Study ->`
Preset Studies - Stationary ->X

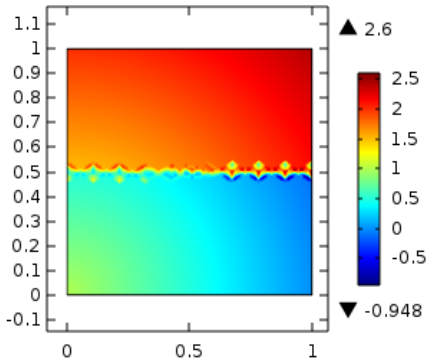
Study 3 : RMB > Compute

2D Plot Group 3:

`Color Legend` - **Show maximum and minimum values** = `True`

LMB > Plot

Буде побудований наступний графік.

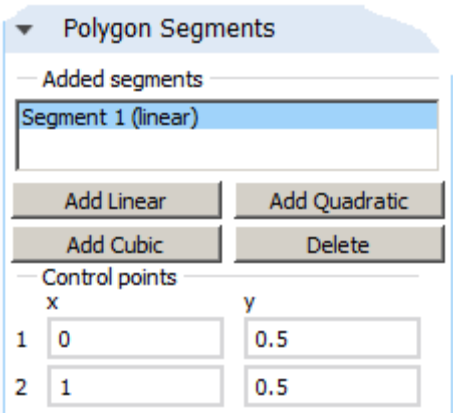


Він не досить точний в околі розділювальної лінії. В зв'язку з цим суттєва помилка з'явилася при відображенні максимального і мінімального значень. Покращити результат можна, якщо геометрію квадрата поділити навпіл лінією, яка є розділювачем кускових початкових умов.

`Geometry` **->** `Primitives` – Bézier Polygon

Geometry 1 - Bézier Polygon 1 (b1): LMB > Add Linear

Введіть координати початкової `(0,0.5)` і кінцевої `(1,0.5)` точок відрізка.



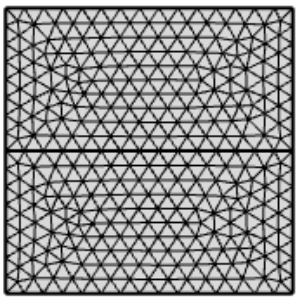
Якщо тепер побудувати нову СкЕ сітку з параметрами за замовчуванням, то її вузли потраплять на розділювальний відрізок.

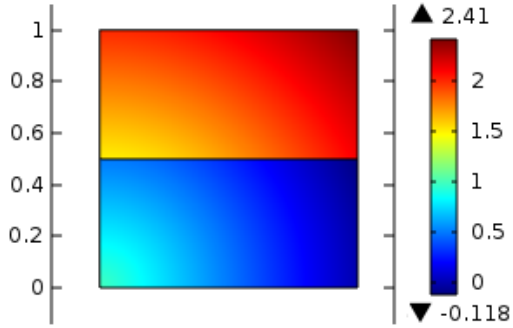
Component 1 – Mesh 1 : LMB > Build All (*наступний рисунок ліворуч*)

Виконавши зараз обчислення моделі, ви отримаєте коректний графік.

Study 3 : RMB > Compute

В графічній групі **2D Plot Group 3** буде побудований графік, зображений на наступному рисунку праворуч.





Отже навіть в одному обчисленні можна спостерігати різні гілки коренів розподіленого нелінійного рівняння. ■

Наведемо ще один приклад розподіленої алгебраїчної моделі, розглянувши нелінійне рівняння переносу

$$\frac{\partial u}{\partial t}+g(u)\frac{\partial u}{\partial x}=0, \quad -\infty<x<\infty,\ 0<t<\infty$$

з початковою умовою $u(x,0)=u_0(x)$. Ця задача має алгебраїчний розв'язок, який зображується неявним рівнянням $u=u_0(x-g(u)\cdot t)$. Щоб його реалізувати, можна створити `1D Domain ODEs and DAEs` динамічну модель, в якій використати це неявне рівняння.

Приклад 3. Дослідимо модель, в якій $g(u)=c_0+c_1u$ і розв'яжемо розподілене алгебраїчне рівняння $u=u_0(x-(c_0+c_1u)\,t)$, де функція $u_0(x)$ фінітна і має вигляд

$$u_0(x)=\begin{cases}(1-\cos 2\pi x)/4, & 0\le x\le 1\\ 0, & \text{інакше}\end{cases}.$$

Р о з в ' я з а н н я . Створіть модель послідовністю команд

File –> New –> Model Wizard –> 1D –> Mathematics - ODEs and DAEs Interfaces - Domain ODEs and DAEs(dode) –> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Створіть таблицю параметрів.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `c0,0.25`

`Parameters` – **Name, Expression** = `c1,0.1`

Створіть функцію $u_0(x)$, яка буде визначати початкові значення.

Home **-> `Definitions`** – Functions –> Local – Piecewise

Component 1 – Definitions – Piecewise 1 (pw1):

`Definition` – **Smoothing** = `Continuous second derivative`

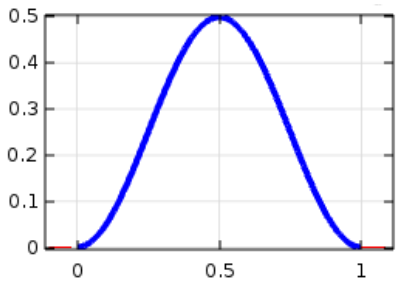
– **Size of transition zone** = `0.01`

– **Intervals(таблиця) – Start, End** = `0,1`

– **Intervals таблиця) – Function** = `0.25*(1-cos(2*pi*x))`

Опція **Extrapolation**=`Constant`, яка встановлена за замовчуванням, означає, що поза відрізком (0,1) функція буде продовжена своїми значеннями на кінцях, тобто нулями.

LMB > Plot



Зверніть увагу на те, що ми задали опцію **Smoothing** згладжування кінців.

Геометрія моделі складається з одного відрізка. Створіть його.

`Geometry`**->`Pimitives`->** Interval

Geometry – Interval 1 (i1):

`Interval` – **Right endpoint** = `3`

LMB > Build All Objects

Побудуйте дуже дрібну СкЕ сітку.

Mesh 1:

`Mesh Settings` – **Element size** = `Extremely fine`

LMB > Build All

Налаштуйте параметри математичної моделі.

Domain ODEs and DAEs (dode) - Distributed ODE 1:

`Source Term` – **f** = `u-pw1(x-(c0+c1*u)*t)`

`Damping or Mass coefficient` – $\mathbf{d_a}$ = `0`

`Mass coefficient` – $\mathbf{e_a}$ = `0`

Задайте початкові значення.

Domain ODEs and DAEs (dode) - Initial Values 1:

`Initial Values` – **Initial value for u:** = `pw1(x)`

Налаштуйте часовий інтервал і відносну похибку.

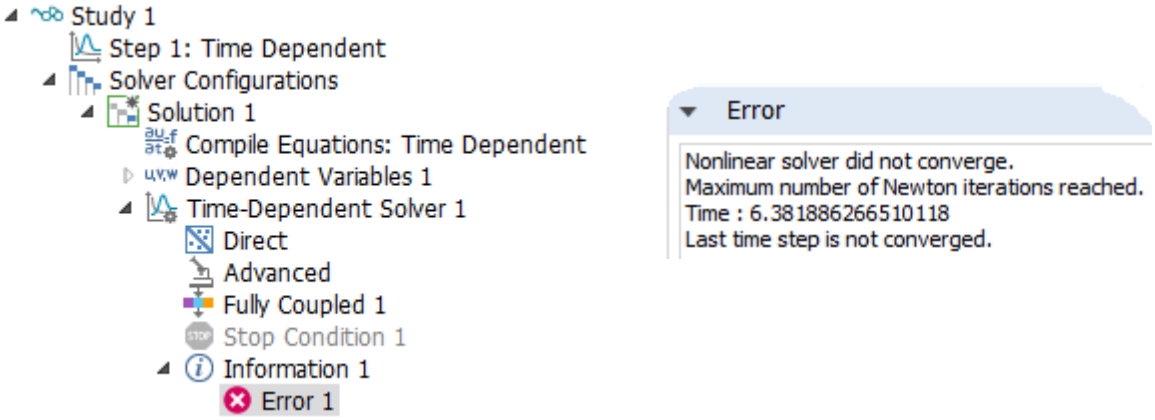
Step 1: Time Dependent:

Study Settings – **Times** = `range(0,0.1,10)`
– **Relative tolerance** = `True, 0.0001`

Виконайте обчислення.

Home->Compute

Обчислення будуть завершені з повідомленням про помилку і в дереві моделі з'явиться вузол **Error** з його текстом (після закриття вікна меседжу).



З повідомлення зрозуміло, що обчислення зупинилися в момент часу `6.381886266510118`. Щоб зрозуміти, що трапилося, бажано побудувати графіки розв'язку до цього моменту. Звичайно, що можна у вузлі **Step 1: Time Dependent** в полі **Times** вкоротити часовий діапазон до `6.3`. Але існує інше рішення – застосування вузла `Stop Condition`.

Динамічні розв'язувачі в COMSOL Multiphysics є адаптивними. Вони самостійно обирають крок по часу в залежності від заданої похибки, і не додержуються часового кроку, встановленого в часовому діапазоні поля **Times** вузла **Step 1: Time Dependent**. Коли розв'язок змінюється повільно, розв'язувач збільшує часовий крок і навпаки, коли розв'язок швидко зростає, розв'язувач зменшує крок по часу. Останнє, зокрема, трапляється коли розв'язок прямує до нескінченності. Щоб впоратися з ситуацією, розв'язувач все зменшує і зменшує крок і ми можемо зупинити обчислення, коли крок по часу стає занадто малим. Таким способом можна обробити/перехопити прямування розв'язку до сингулярності. Для цього в системі існує зарезервована змінна `timestep` (довжина часового кроку), яку можна перевіряти на кожному кроці розв'язувача у вузлі `Stop Condition` і, якщо треба, зупиняти обчислення.

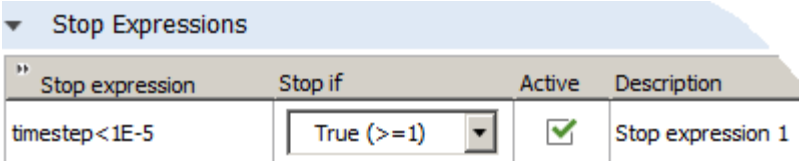
Налаштуємо розв'язувач таким чином, щоб він зупинявся, коли змінна `timestep` стає занадто малою. Для цього в гілку **Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1** потрібно додати вузол `Stop Condition`, в якому задати умову зупинки.

Time-Dependent Solver 1: RMB > Stop Condition

Time-Dependent Solver 1 - Stop Condition 1:

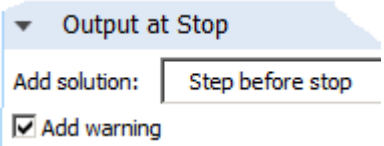
Додайте умову зупинки, натиснувши в зоні таблиці `Stop Expressions` праву кнопку миші. Оберіть з випадного списку команду Add, і введіть

Stop Expressions – **Stop Expressions** = `timestep < 1E-5`



Output at stop –**Add solution** = Steps before stop

Значення Step before stop використовується для зберігання розв'язку в останній момент часу, в який умова ще не була виконана.



Переобчисліть задачу.

Home -> Compute

В гілці **Results** з'явиться гілка **1D Plot Group 1** з вузлом **Line Graph 1**, який містить графік розв'язку. Скорегуйте його.

1D Plot Group 1:

Data – Time selection = Manual

Data – Times (1-65) = range(1,10,65)

Тут для побудови графіків ми обрали кожний десятий момент часу з 65, які запропонував розв'язувач.

Налаштуйте стиль кривих і накажіть відобразити легенду.

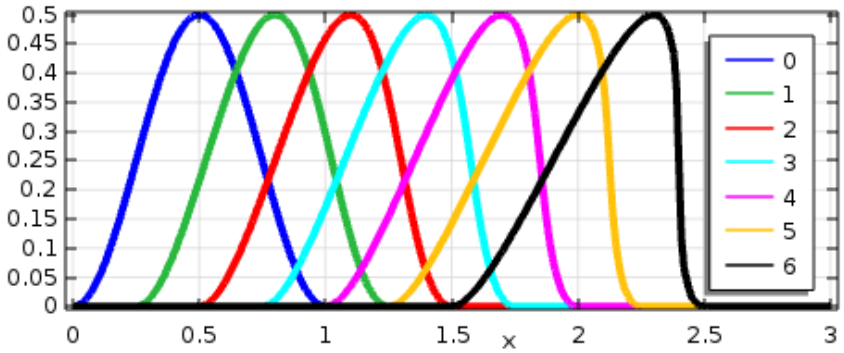
1D Plot Group 1 - Line Graph 1:

Coloring and Style –**Width** = 3

Legend – Show legends = True

LMB > Plot

В результаті ви отримаєте графіки функції $u(x,t)$ для моментів часу, наведених в легенді.



Як бачите, гребінь хвилі рухається швидше за її підніжжя і поступово його наздоганяє. Після того, як правий фронт хвилі стане вертикальним, гребінь хвилі повинен випередити її підніжжя. Але в цьому випадку порушується однозначність розв'язку і система спочатку виводила повідомлення про помилку, а зараз – зупиняє обчислення.

Для тренування побудуємо ще один графік. Для репрезентування розв'язку $u(x,t)$ можна використати джерело даних Parametric Extrusion 1D, яке дозволяє параметр часу (або будь який інший параметр моделі) зобразити як додаткову просторову координату. Створіть і налаштуйте таке ДД.

Results->More Data Sets-> Parametric Extrusion 1D

Results – Data Sets - Parametric Extrusion 1D 1:

Data – Data set = Study 1/Solution 1

Data– Time selection = Manual

Data– Time(s)(1-65) = range(1,6,65)

На базі створеного ДД можна побудувати графік. Додайте в гілку **Results** двовимірну графічну групу.

Results : RMB > 2D Plot Group

2D Plot Group 2:

Data– Data set = Parametric Extrusion 1D 1

Plot Settings – Plot data set edges = False

В групу додайте вузол типу Line.

2D Plot Group 2 : RMB > Line

2D Plot Group 2 - Line 1:

Data– Data set = From Parent

Expression – Expression = u

Coloring and Style – Line type =Tube

– Tube radius expression =0.02

– Radius scale factor =1

– Coloring, Color =Uniform, Black

У вузлі **Line 1** створіть підвузол Height Expression.

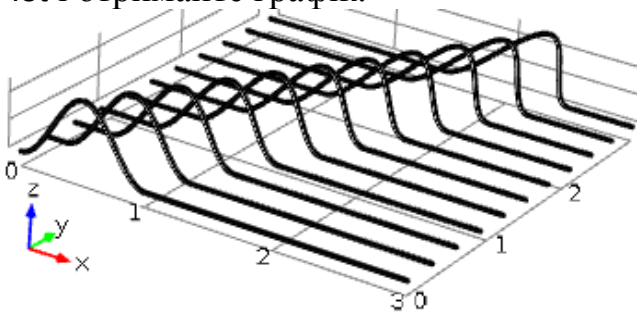
2D Plot Group 2->Attributes - Height Expression

2D Plot Group 2 - Line 1 - Height Expression 1 :

Expression – Height data = From parent

Axis – Scale factor = 1

Виконайте команду Plot і отримайте графік.



На цьому рисунку в напрямку y відкладено параметр часу і криві, які побудовані при різних значеннях $y \equiv t$, зображають графік розв'язку в різні моменти.

Процес руху хвилі виглядає наглядніше, якщо побудувати анімацію зміни розв'язку $u(x,t)$ з часом. Зробіть це самостійно.

Вправа. Розв'жіть розглянуту задачу, використавши інший початковий профіль хвилі, наприклад, такий

$$u_0(x) = \begin{cases} 2x(1-x), & 0 \le x \le 1 \\ 0, & \text{інакше} \end{cases},$$

і перевірте модель. Крім зміни виразу функції, вам доведеться підібрати часовий інтервал у вузлі **Step 1: Time Dependent**. Гребінь хвилі, як і раніше, буде рухатися швидше за її підніжжя і поступово його наздоганяти. Система COMSOL Multiphysics буде виконувати обчислення до часу, коли правий фронт хвилі не стане вертикальним (в деяких точках). Після цього обчислення неможливі через багатозначність розв'язку. Отже і в цьому випадку ви повинні або зменшити (підібравши) інтервал часу, або використати вузол `Stop Condition`. В другому випадку вам, вірогідно, потрібно буде згладити початкову функцію `pw1(x)`, задавши ширший ніж в розглянутому прикладі перехідний інтервал **Size of transition zone** =`0.1` і встановити перемикач **Smooth at endpoints** = `True`.

4.3.2. Розподілені диференціальні рівняння.

Приклад. Нехай у нас є дві перемішані речовини A і B, які в процесі хімічної реакції утворюють нову речовину D. Відомо, що з кожних двох об'ємів речовини A і одного об'єму речовини B утворюється три об'єми речовини D. Початкові концентрації C_A і C_B речовин A і B відомі, а початкова концентрація u продукту D дорівнює нулю. Процес відбувається в вертикальному циліндричному резервуарі $\left(x^2+y^2\le R\right)\times\left(0\le z\le H\right)$. Додаванння продуктів в резервуар, пермішування і диффузії речовин не відбувається. Температура в баку незмінна і розподілена за законом $T(x,y,z)=T_0+k_T z(H-z)\left(R^2-x^2-y^2\right)$.

Нехай $u(x,y,z,t)$ позначає концентрацію (відносний об'єм) речовини D. Тоді з умов задачі випливає, що в момент часу t в одиниці об'єму в хімічну реакцію вступило $\frac{2}{3}u$ об'єму речовини A і $\frac{1}{3}u$ об'єму речовини B. Це означає, що до вказаного моменту часу залишилося $C_A-\frac{2}{3}u$ відносного об'єму речовини A і $C_B-\frac{1}{3}u$ відносного об'єму речовини B. Згідно з законом діючих мас отримуємо диференціальне рівняння

$$\frac{du}{dt}=K(T)\cdot\left(C_A-\frac{2u}{3}\right)\cdot\left(C_B-\frac{u}{3}\right), \qquad (1)$$

де враховано, що швидкість хімічної реакціі залежить від температури T, тобто $K=K(T)$ є функцією температури.

Це звичайне диференціальне рівняння потрібно розв'язувати в кожній точці (x,y,z), бо воно містить просторові координати, які входять в вираз для визначення температури $T=T(x,y,z)$. Отже ми маємо розподілене звичайне диференціальне рівняння.

Розглянемо безрозмірну модель і покладемо параметри моделі $R=1, H=3$, $K(T)=k_0+k_1(T-T_0)$, $k_0=1, k_1=1$, $T_0=273^0C$, $k_T=10$, $C_A=0.4, C_B=0.2$.

Р о з в ’ я з а н н я . Створіть модель послідовністю команд

File -> New -> Model Wizard -> 3D -> Mathematics - ODEs and DAEs Interfaces - Domain ODEs and DAEs(dode) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Створіть таблицю параметрів і задайте їх значення.

Global Definitions : RMB > Parameters

Parameters – **Name, Expression** = R,1
– **Name, Expression** = H,3
– **Name, Expression** = k0,1
– **Name, Expression** = k1,1
– **Name, Expression** = kT,10
– **Name, Expression** = T0,273
– **Name, Expression** = CA,0.4
– **Name, Expression** = CB,0.2

Створіть функцію $T_S(x)$, яка буде визначати розподілення температури.

Home -> **Definitions** -> Functions -> Local -> Analytic

Component 1 – Definitions – Analytic 1:

Function name = Ts

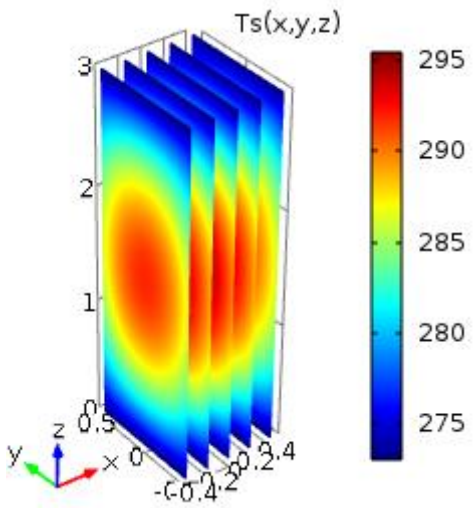
Definition – **Expression**= T0+kT*z*(H-z)*(R^2-x^2-y^2)
– **Arguments**= x,y,z

Для контролю побудуйте графік функції, визначивши тимчасові діапазони аргументів.

Plot Parameters

Argument	Lower limit	Upper limit
x	-0.5	0.5
y	-0.5	0.5
z	0	3

LMB > Plot



Побудуйте геометрію.

Geometry->**Pimitives**-> Cylinder

Geometry – Cylinder 1 (cyl1):

`Size and Shape` – **Radius, Height** = `R, H`

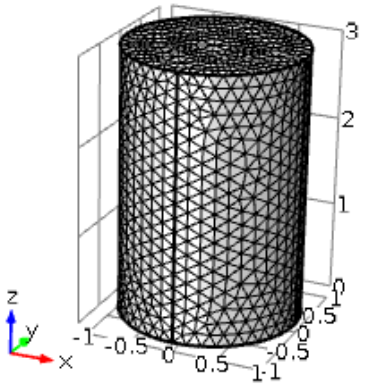
LMB > Build All Objects

Побудуйте СкЕ сітку.

Component 1 – Mesh 1 :

`Mesh Settings` – **Element size** = `Fine`

LMB > Build All



Налаштуйте параметри моделі.

Domain ODEs and DAEs (dode) - Distributed ODE 1:

`Source Term` – **f** = `(k0+k1*(Ts(x,y,z)-T0))*(CA-2*u/3)*(CB-u/3)`

`Damping or Mass coefficient` – $\mathbf{d_a}$ = `1`

`Mass coefficient` – $\mathbf{e_a}$ = `0`

Задайте нульові початкові значення (вони встановлені за замовчуванням).

Domain ODEs and DAEs (dode) - Initial Values 1:

`Initial Values` – **Initial value for u:** = `0`

Налаштуйте часовий інтервал і відносну похибку.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.2,10)`

– **Relative tolerance** = `True, 0.001`

Виконайте обчислення.

`Home` -> Compute

Майте на увазі, що обчислення триватимуть деякий час.

В гілці **Results** з'явиться графічна група **3D Plot Group 1** з вузлом **Slice 1**, який містить графік розв'язку. Замініть його вузлом типу `Multislice`. Спочатку видаліть вузол **Slice 1**.

3D Plot Group 1 - Slice 1 : RMB > Delete

Потім замість нього створіть вузол `Multislice`.

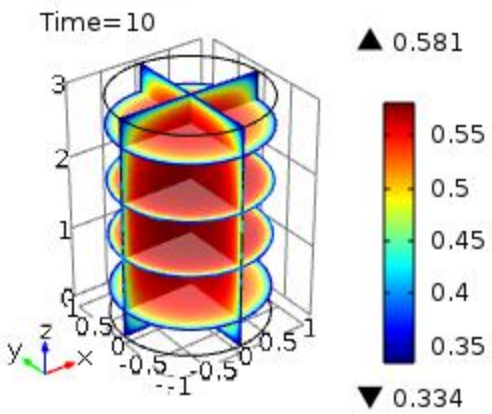
3D Plot Group 1: RMB > More Plots - Multislice

`Multiplane Data` – **z-planes - Planes** = `4`

3D Plot Group 1:

`Color Legend` - **Show maximum and minimum values** = `True`

LMB > Plot



Побудуйте графіки залежності концентрації *u* речовини D від часу в деяких точках.

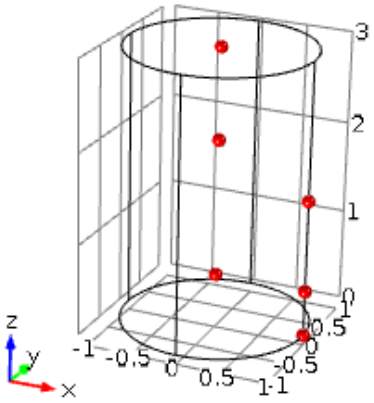
Спочатку створіть джерело даних, яке складатиметься з контрольних точок.

Data Sets: RMB > Cut Point 3D

3D Plot Group 1 - Cut Point 3D 1 :

Point Data – **Entry method** = Coordinates
– **x** = 1,1,0,0,0,1
– **y** = 0,0,0,0,0,0
– **z** = 0,0.5,2,3,0.5,1.5

LMB > Plot



Потім створіть одновимірну графічну групу з вузлом типу Point Graph, який буде використовувати це джерело даних.

Results : RMB > 1D Plot Group

1D Plot Group 2 :

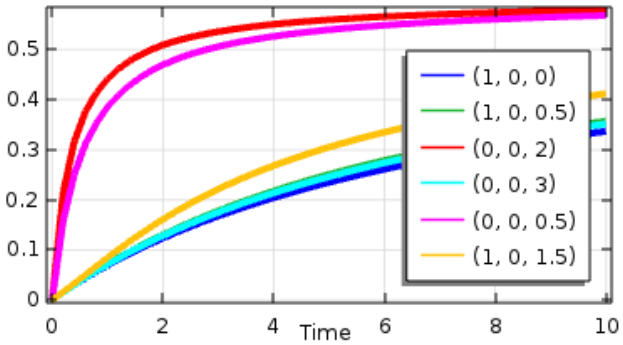
Legends – **Position** =Lower Right

1D Plot Group 2 : RMB > Point Graph

1D Plot Group 2 - Point Graph 1:

Data – **Data set** = Cut Point 3D 1
Data – **Time selection** = All
Expression – **Expression** = u
Coloring and Style – **Color, Width** =Cycle, 3
Legends – **Show legends** =True

LMB > Plot



Якщо придивитися до функції, яка обчислює температуру

$$T(x,y,z)=T_0+k_T z(H-z)(R^2-x^2-y^2),$$

то побачимо, що $T(1,0,0)=T(1,0,0.5)=T(0,0,3)=T(1,0,1.5)=T_0$. Тоді $K(T)=k_0+k_1(T-T_0)=k_0$, і рівняння (1) набуде однакового вигляду в цих точках $\frac{du}{dt}=k_0\cdot\left(C_A-\frac{2u}{3}\right)\cdot\left(C_B-\frac{u}{3}\right)$. Отже і розв'язки цього рівняння повинні бути однаковими в цих точках (чотири нижні криві повинні співпадати). Але з графіка випливає що це не так. Насправді, така різниця пояснюється похибками обчислення, а точніше розміром СкЕ сітки. Якщо СкЕ сітку зробити дрібнішою (виконайте це самостійно), то криві розташуються щільніше. Але майте на увазі, що час обчислення задачі значно зросте.

Оскільки хімічна реакція в кожній точці проходить незалежно від сусідніх точок, то концентрацію речовини D в ній можна отримати, розв'язуючи «нульвимірне» ЗДР (1) з коефіцієнтом $K(T_p)$, який обчислено при заданій в точці температурі T_p. Для тренування обчислимо функцію $u(t)$ в точці $(1,0,1.5)$, розв'язавши відповідне для цієї точки ЗДР, і побудуємо графік отриманого розв'язку.

Додамо в модель «фізику» `Global ODEs and DAEs` командою

`Physics` -> Physics -> Add Physics

У відкрившемуся праворуч вікні `Add Physics` виберіть гілку

Mathematics - ODEs and DAEs Interfaces - Global ODEs and DAEs (ge)

двічі клацнувши по ній мишею. Потім закрийте вікно `Add Physics` кнопкою-хрестиком, розташованим вгорі праворуч.

Зверніть увагу, що на головній інструментальній стрічці з'явилася панель `Global Equations`.

В ПВ створеного вузла задайте рівняння.

Component 1 (comp 1) - Global ODEs and DAEs (ge) - Global Equations 1:

`Global Equations` – **Name** = `w`

`Global Equations` – **f(u,ut,utt,t)(1)** =

```
(k0+k1*(Ts(1,0,1.5)-T0))*(CA-2*w/3)*(CB-w/3)-wt
```

Щоб змінна `w` стала доступною, знову обчисліть модель.

Study 1 : RMB > Compute

Зауваження. Задачу треба саме обчислити командою Compute, а не оновити командою Update Solution.

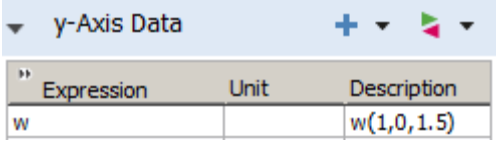
До вузла **1D Plot Group 2** додайте підвузол типу `Global`.

1D Plot Group 2 : RMB > Global

1D Plot Group 2 - Global 1:

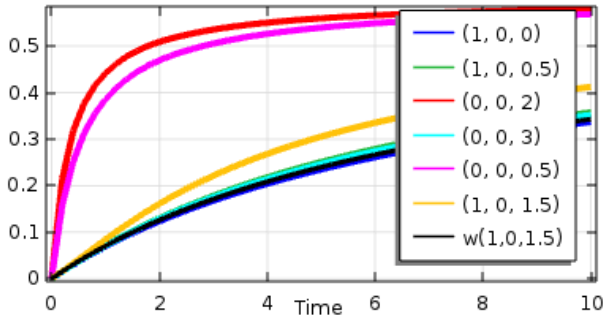
`y-Axis Data` – **Expression** = `w`

– **Description**= `w(1,0,1.5)`



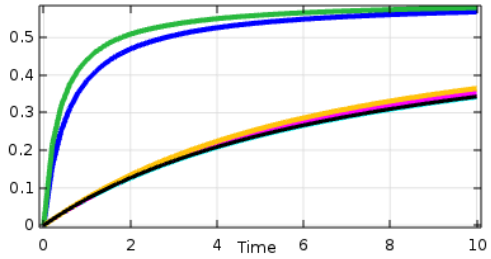
`Coloring and Style` – **Color, Width** =`Cycle, 3`

LMB > Plot



В графічному вікні з’явиться нова крива, яка накладається на одну з вже існуючих, розташованих внизу.

Зауваження. Різниця в формі кривих для точок $(1,0,0),(1,0,0.5),(0,0,3)$, $(1,0,1.5)$ зменшиться, якщо ці точки відразу створити в геометрії моделі.



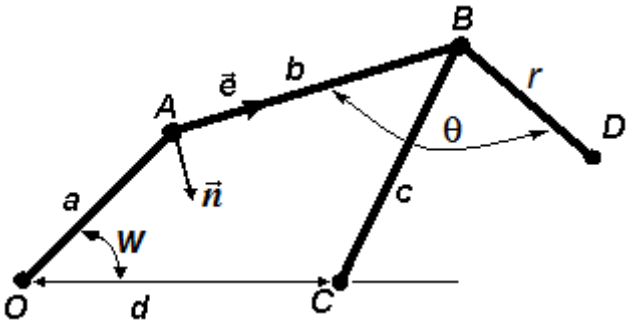
4.4. Демонстраційні приклади.

4.4.1. Триланковий механізм по Чебишеву.

Математики 19 століття приділяли багато уваги дослідженню шарнірних механізмів. Зокрема теорія Чебишева про апроксимацію функцій, які найменше відхиляються від нуля, має своє коріння в цьому питанні. Зараз його актуальність визначається розвитком робототехніки, яка передбачає створення промислових механізмів, виконавчі органи яких здійснюють рухи, визначені технологією виробництва.

Триланковий механізм складається з трьох ланок: ведуча ланка OA обертається навколо точки O зі сталою кутовою швидкістю, ланки OA, AB і BC шарнірно з'єднані в точках A і B. Ланка BC може здійснювати вільне обертання в точці C. В механізмі Чебишева ланка AB продовжена плечем BD, яке утворює фіксований кут з AB таким чином, що коліно ABD є жорстким і в точці B

шарнірно пов'язане з ланкою BC [1]. Задача полягає в визначенні траєкторії руху точки D.



Нехай точка O розташована в початку координат. Позначимо довжини ланок *a*, *b*, *c* так, як показано на рисунку. Відстань по горизонталі між точками O і C позначимо через *d*, довжину плеча BD – через *r*, кут ABD – через θ. Щоб точка A могла пройти горизонтальне положення зліва від точки O необхідно виконання співвідношення *a*+*d* <=*b*+*c*.

Для змінних координат векторів (ланок) $\vec{O}A, \vec{A}B, \vec{B}C$ введемо наступні позначення $\vec{O}A = \left(a_x, a_y\right)$ $\vec{A}B = \left(b_x, b_y\right)$, $\vec{B}C = \left(c_x, c_y\right)$. Якщо кут повороту ланки OA позначити через ω, то

$$a_x = a\cos\omega,\ a_y = a\sin\omega. \tag{1}$$

Довжини векторів $\vec{A}B, \vec{B}C$ відомі. Тому

$$b_x^{\ 2} + b_y^{\ 2} = b^2, \quad c_x^{\ 2} + c_y^{\ 2} = c^2. \tag{2}$$

Позначимо $\vec{e} = \left(b_x, b_y\right)/b$ одиничний вектор уздовж AB, а одиничний вектор йому перпендикулярний – через $\vec{n} = \left(b_y, -b_x\right)/b$. Тоді

$$\vec{B}D = \left(BD_x, BD_y\right) = r\left(-\vec{e}\cos\omega + \vec{n}\sin\omega\right),$$

або

$$BD_x = \frac{r}{b}\left(-b_x\cos\omega + b_y\sin\omega\right), \tag{3}$$

$$BD_y = \frac{r}{b}\left(-b_y\cos\omega - b_x\sin\omega\right). \tag{4}$$

Крім того, маємо $\vec{O}A + \vec{A}B + \vec{B}C = \vec{O}C = \left(d, 0\right)$. В покоординатному запису це має вигляд

$$a_x + b_x + c_x = d, \tag{5}$$

$$a_y + b_y + c_y = 0. \tag{6}$$

Отже, величини b_x, b_y, c_x, c_y пов’язані в нелінійну систему алгебраїчних рівнянь (2), (5), (6), розв’язавши яку, можна отримати координати точки $D\left(D_x, D_y\right)$ за формулами

$$D_x = a_x + b_x + BD_x,\ D_y = a_y + b_y + BD_y. \tag{7}$$

На перший погляд здається, що для моделювання цієї задачі потрібно використати динамічний (Time Dependen) `Global ODEs and DAEs` інтерфейс. Але система рівнянь нелінійна і для її розв’язання буде застосовуватися

ітераційний алгоритм, якому потрібні «гарні» початкові наближення. Задати їх водночас для всіх моментів часу буде важко (або неможливо). Та в системі існує допоміжний параметричний аналіз (`Auxiliary Sweep`). При його застосуванні вважається, що параметр задачі змінюється неперервно, а результат, отриманий на попередньому кроці, розглядається як початкове значення для нової ітерації. В нашій задачі в якості параметра можна використати кут повороту ω ланки OA і підібрати вдалі початкові наближення для стартового положення механізму. Початкові наближення для нових значень b_x, b_y, c_x, c_y будуть обиратися як розв'язки системи на попередньому кроці при попередньому значенні кута ω.

Приклад. Створіть модель задачі з Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 2D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

Зверніть увагу на те, що будується двовимірна стаціонарна модель.

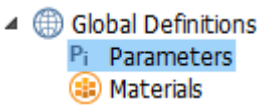
Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

В гілці **Global Definitions** створіть вузол параметрів і заповніть його таблицю.

Global Definitions: RMB > Parameters

`Parameters` – **Name, Expression** = `w, pi/4`
– **Name, Expression** = `a, 2`
– **Name, Expression** = `b, 3`
– **Name, Expression** = `c, 3`
– **Name, Expression** = `d, 3.1`
– **Name, Expression** = `r, 1.5`
– **Name, Expression** = `theta, pi/2`



Name	Expression	Value	Description
w	pi/4	0.7854	Кут повороту плеча OA
a	2	2	Довжина OA
b	3	3	Довжина AB
c	3	3	Довжина BC
d	3.1	3.1	Відстань OC
r	1.5	1.5	Довжина BD
theta	pi/2	1.5708	Кут ABD

В ПВ вузла **Global Equations 1** заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** = `bx, bx^2+by^2-b^2`
– **Initial value** = `a/sqrt(2)+b`

`Global Equations` – **Name, f(u, ut, utt, t)** = `by, ay+by+cy`
– **Initial value** = `a/sqrt(2)`

`Global Equations` – **Name, f(u, ut, utt, t)** = `cx, ax+bx+cx-d`
– **Initial value** = `-a/sqrt(2)`

`Global Equations` – **Name, f(u, ut, utt, t)** = `cy, cx^2+cy^2-c^2`
– **Initial value** = `-a/sqrt(2)`

Name	f(u,ut,utt,t)	Initial value (u_0)
bx	bx^2+by^2-b^2	a/sqrt(2)+b
by	ay+by+cy	a/sqrt(2)
cx	ax+bx+cx-d	-a/sqrt(2)
cy	cx^2+cy^2-c^2	-a/sqrt(2)

Створіть таблицю локальних змінних.

Home **-> Definitions** – Variables –> Local Variables

Component 1 – Definitions – Variables 1:

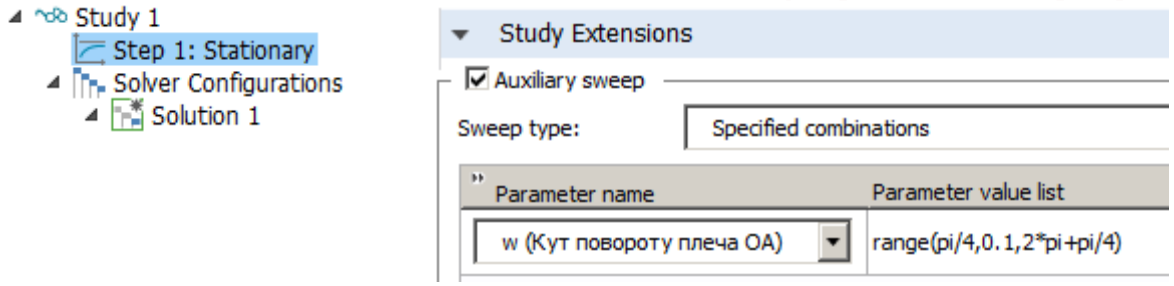
Variables – **Name, Expression** = ax, a*cos(w)
– **Name, Expression** = ay, a*sin(w)
– **Name, Expression** = ex, bx/b
– **Name, Expression** = ey, by/b
– **Name, Expression** = nx, ey
– **Name, Expression** = ny, -ex
– **Name, Expression** =
bdx,r*(-ex*cos(theta)+nx*sin(theta))
– **Name, Expression** =
bdy, r*(-ey*cos(theta)+ny*sin(theta))
– **Name, Expression** = pdx, ax+bx+bdx
– **Name, Expression** = pdy, ay+by+bdy

Name	Expression	Unit	Description
ax	a*cos(w)		абсциса вектора OA
ay	a*sin(w)		ордината вектора OA
ex	bx/b		
ey	by/b		
nx	ey		
ny	-ex		
bdx	r*(-ex*cos(theta)+nx*sin(theta))		абсциса вектора BD
bdy	r*(-ey*cos(theta)+ny*sin(theta))		ордината вектора BD
pdx	ax+bx+bdx		абсциса точки D
pdy	ay+by+bdy		ордината точки D

Ввімкніть в моделі застосування Auxiliary Sweep аналізу.

Study 1 – Step 1: Stationary:

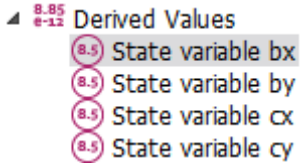
Study Extensions – **Auxiliary sweep** = True
– **таблиця:** RMB > Add
– **Parameter name** = w
– **Parameter value list** = range(pi/4,0.1,2*pi+pi/4)



Виконайте обчислення.

Home **-> Compute**

В гілці **Derived Values** будуть створені вузли типу `Global Evaluation` для обчислення величин bx, by, cx, cy.



Побудуйте графіки залежності цих величин від параметра ω. Для цього створіть одновимірну графічну групу з вузлом `Global`, в якому намалюйте їх графіки.

`Results` –> Plot Group - 1D Plot Group

1D Plot Group 1 : **`Legend`** – **Position** = `Lower left`

`1D Plot Group 1` –> Add Plot – Global

1D Plot Group 1 - Global 1:

`y-Axis Data` – **Expression, Description** = `bx,bx` (row1)

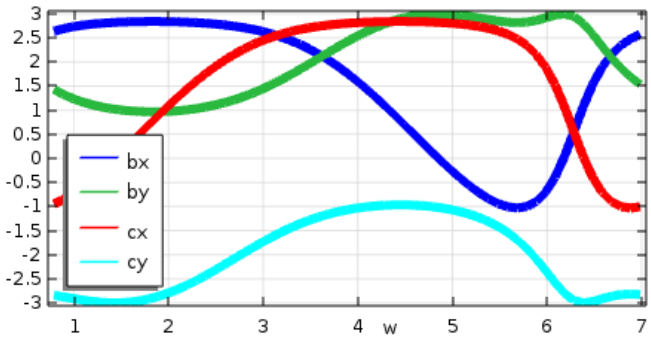
– **Expression, Description** = `by,by` (row2)

– **Expression, Description** = `cx,cx` (row3)

– **Expression, Description** = `cy,cy` (row4)

Expression	Unit	Description
bx		bx
by		by
cx		cx
cy		cy

LMB > Plot



Для побудови схематичного зображення механізму знадобиться джерело даних `Grid 1D`.

Results - Data Sets: RMB> More Data Sets -> Grid 1D

Data Sets - Grid 1D 1 :

`Data` – **Source** = `Data set`

– **Data set** = `Study 1/Solution 1`

`Paramemter Bounds` – **Name** = `u`

– **Minimum, Maximum** = `0,1` (*два поля*)

Тепер можна зобразити ескіз механізму в окремій графічній групі.

`Results` –> Plot Group - 1D Plot Group

1D Plot Group 2 :

`Legend` – **Show legends** = `False`

У вузлах лінійних графіків цієї групи побудуйте зображення ланок механізму по їх параметричним рівнянням, де в якості параметра використайте поздовжню змінну u джерела даних **Grid 1D 1**.

1D Plot Group 2 -> Add Plot – Line Graph

1D Plot Group 2 - Line Graph 1:

Label = Line Graph OA

Data – **Data set** = Grid 1D 1

Зображати механізм будемо наприкінці періоду його руху.

Data – **Parameter selection(w)** = Last

y-Axis Data – **Expression** = ay*u

x-Axis Data – **Parameter, Expression** = Expression,ax*u

Coloring and Style – **Color, Width** = Red, 4

1D Plot Group 2 - Line Graph OA : RMB > Duplicate

Label = Line Graph AB

y-Axis Data – **Expression** = ay+by*u

x-Axis Data – **Parameter, Expression** = Expression, ax+bx*u

1D Plot Group 2 - Line Graph AB : RMB > Duplicate

Label = Line Graph BC

y-Axis Data – **Expression** = ay+by+cy*u

x-Axis Data – **Parameter, Expression** = Expression, ax+bx+cx*u

1D Plot Group 2 - Line Graph AB : RMB > Duplicate

Label = Line Graph BD

y-Axis Data – **Expression** = ay+by+bdy*u

x-Axis Data – **Parameter, Expression** = Expression, ax+bx+bdx*u

Додайте зображення кінцевої точки D. Її намалюйте теж за допомогою вузла типу Line Graph.

1D Plot Group 2 - Line Graph AB : RMB > Duplicate

Label = Line Graph PD

y-Axis Data – **Expression** = ay+by+bdy

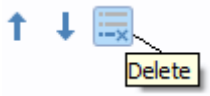
x-Axis Data – **Parameter, Expression** = Expression, ax+bx+bdx

Coloring and Style – **Color, Width** = Black, 6

Зображення траєкторії руху кінцевої точки D механізму побудуйте в цій же графічній групі, використовуючи вузол типу Global.

1D Plot Group 2 -> Add Plot – Global

В розділі **y-Axis Data** створеного вузла **Global 1** очистіть рядки таблиці виразів за допомогою кнопки Delete, що розташована внизу таблиці.



Потім введіть вирази.

1D Plot Group 2 - Global 1:

y-Axis Data – **Expression** = pdy

x-Axis Data – **Parameter, Expression** = Expression, pdx

Задайте межі зміни просторових координат цієї графічної групи.

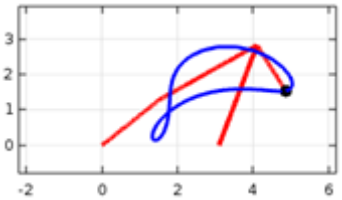
1D Plot Group 2:

Axis – Manual axis limits = True

– x minimum, x maximum = -2.2,6.2

– y minimum, y maximum = -3,5

LMB > Plot



Побудуйте вузол анімації.

Results –> Animation –> Player

Export – Animation 1:

Scene – Subject = 1D Plot Group 2

Frames– Frame selection = All

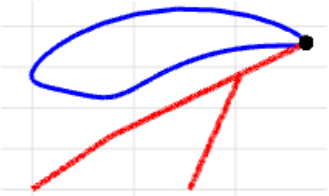
Playing – Repeat = True

LMB > Generate Frame.

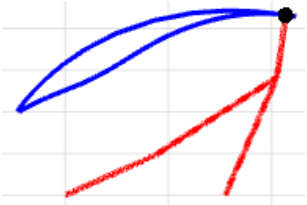
Запустіть анімацію кнопкою Play. На попередньому рисунку зображено кадр анімації наприкінці періоду руху для механізму Чебишева з $\vartheta = \pi/2$. На наступних рисунках зображено кадри в той же кінцевий момент часу для механізмів з $\theta = \frac{3\pi}{4}, \pi, \frac{5\pi}{4}$.

$\vartheta = 3\pi/4$

$\vartheta = \pi$



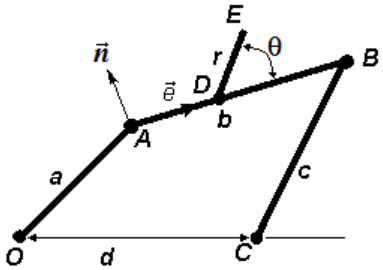
$\vartheta = 5\pi/4$



Як бачимо, криві значно відрізняються. Змінюючи значення кута θ, довжини плечей *a*, *b*, *c*, і відстань *d* можна отримати безліч різних кривих.

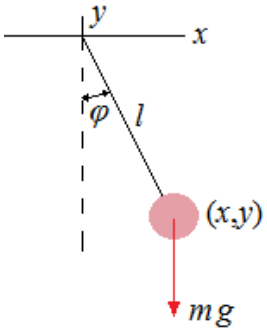
Зауваження. У розглянутої системи алгебраїчних рівнянь існує явний формульний розв'язок (достатньо складний). Але нашою метою було продемонструвати можливості Auxiliary Sweep параметричного аналізу при дослідженні нелінійних задач.

Вправа. Дослідити триланковий механізм з плечем DE, яке жорстко закріплено всередині ланки AB під кутом θ.



4.4.2. Рух маятника.

Дослідимо поведінку математичного маятника.



Нехай маса вантажу дорівнює одиниці, а стрижень, на якому підвішена маса, невагомий. Тоді диференціальне рівняння руху вантажу має вигляд

$$\varphi'' + k\varphi' + \omega^2 \sin\varphi = 0,$$

де $\varphi(t)$ кут відхилення маятника від положення рівноваги, параметр k характеризує опір середовища, $\omega^2 = g/l$ (g прискорення вільного падіння, l - довжина стрижня). Для визначення конкретного руху до рівняння треба додати початкові умови, $\varphi(0) = \varphi_0$, $\varphi'(0) = v_0$.

Для модельної задачі покладемо $k=0.5$, $\omega = \sqrt{10}$, і використаємо наступні початкові значення $\varphi_0 = 0, v_0 = 5$.

Розв'язання 1. Створіть безрозмірну нульвимірну модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Untitled.mph (root): `Unit System` – <> = None

В гілці **Global Definition**s створіть вузол **Parameters**, і в його таблиці задайте параметри `k`, `w`, `v0`.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `k, 0.5`
– **Name, Expression** = `w, sqrt(10)`
– **Name, Expression** = `v0, 5`

Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** =`p, ptt+k*pt+w^2*sin(p)`
– **Initial value(u_0), Initial value(u_t0)** = `0, v0`

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
p	ptt+k*pt+w^2*sin(p)	0	v0

Задайте діапазон зміни параметра `t` і відносну похибку.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.05,20)`
– **Relative tolerance** = `True, 0.000001`

Виконайте обчислення.

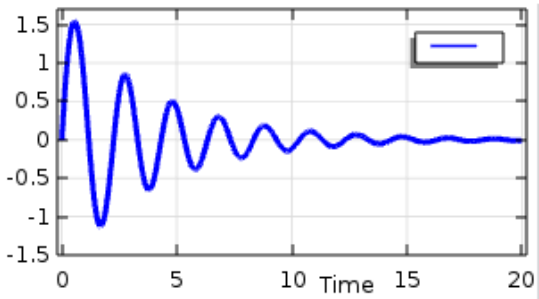
`Home` -> Compute

В гілці **Results** буде створено вузол **1 D Plot Group 1 - Global 1** з графіком розв'язку.

1 D Plot Group 1 - Global 1:

Coloring and Style –**Width** = 3

Legends – **Description** = False



Як бачимо з графіка, максимальний кут відхилення маятника не перевищує $\pi/2$ і коливання маятника загасають.

Побудуйте фазову траєкторію

Results -> Plot Group – 1D Plot Group

1D Plot Group 2 -> Add Plot – Global

1D Plot Group 2 - Global 1:

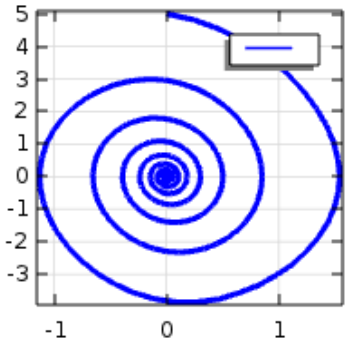
y-Axis Data(таблиця) – **Expression** = comp1.pt

– **Description** = *видалити все*

x-Axis Data – **Parameter, Expression** = Expression, comp1.p

Coloring and Style – **Width** =3

LMB > Plot



Виконаємо дослідження руху маятника в залежності від початкової кутової швидкості v_0. Для цього застосуємо допоміжний аналіз (Auxiliary Sweep) по цьому параметру. Наприклад, розв'яжемо задачу при v0=5,7,9,11.

Study 1 – Step 1: Time Dependent:

Study Extensions – **Auxiliary sweep** = True

Study Extensions(таблиця) – **Parameter name** = v0

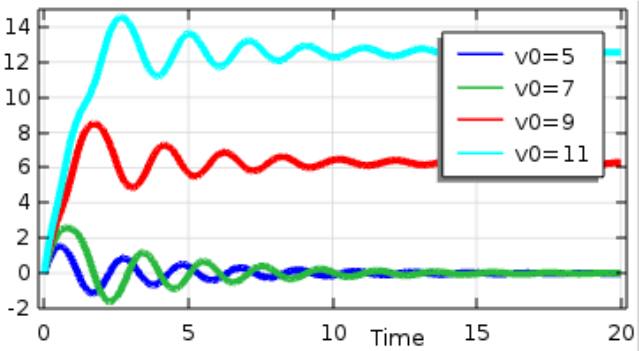
– **Parameter value list** = 5,7,9,11

Переобчисліть модель.

Home -> Compute

Обидва графіка зміняться.

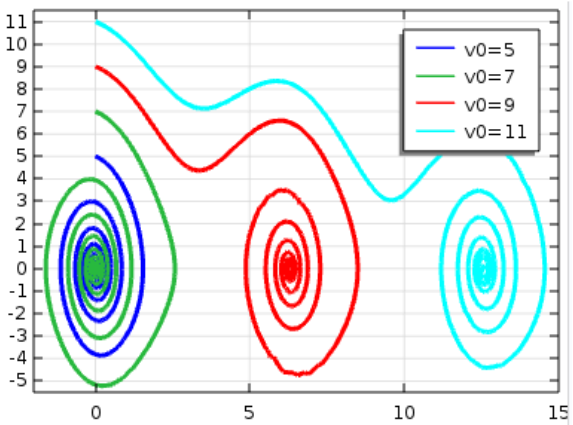
1 D Plot Group 1 : LMB > Plot



Як бачимо, при початковій швидкості v0=5 і 7 коливання загасають в околі кута $\varphi = 0$. Тобто цих початкових швидкостей недостатньо, щоб маятник пройшов верхню точку і зробив хоча б один повний оберт. При початковій швидкості v0=9 маятник робить один повний оберт, а потім його коливання загасають в околі кута $\varphi = 2\pi$ (для маятника кут повороту 2π представляє те ж саме, що і 0 радіан, тобто положення рівноваги). При v0=11 маятник виконав два повних оберти і тільки після цього його коливання почали затухати навколо положення рівноваги $\varphi = 4\pi$.

Ту ж поведінку, але по-іншому, демонструють фазові криві.

1D Plot Group 2 : LMB > Plot



Побудуємо анімацію руху маятника. Для цього потрібно зобразити стрижень і масу (точку) на його кінці в деякий момент часу, а потім виконати анімацію, змінюючи час. Для побудови такого рисунка нам знадобиться двовимірна геометрія. Тому додайте в модель двовимірний компонент з геометрією у формі квадрата. В цьому компоненті ми не будемо розв'язувати ніякої задачі, але його розрахункова область знадобиться для побудови зображення маятника.

Home **-> Model ->** Add Component - 2D

Активуйте гілку нового компонента і побудуйте в ньому квадрат.

Component 2 - Geometry 1 :> Geometry->**Primitives –** Square

Component 2 - Geometry 1 - Square 1 (sq1):

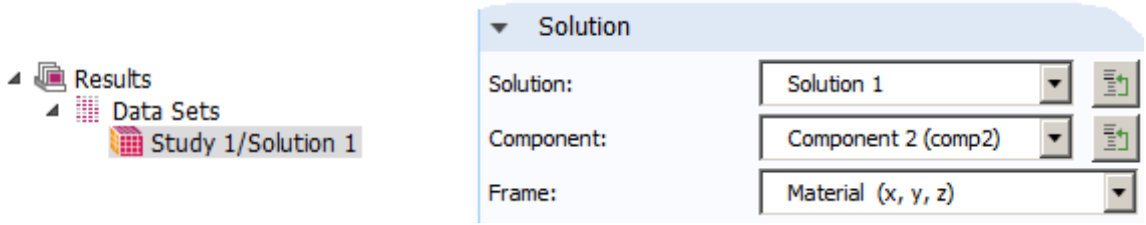
Size – **Size length** = 2.5

Position – **Base, x, y** = Center,0,0

Переобчисліть модель.

Home **->** Compute

Оновлене джерело даних **Study 1/Solution 1** містить розв'язок задачі з першого компонента, а розрахункову область – з другого. Це можна побачити з налаштувань поля **Component** вузла **Study 1/Solution 1**.



Створіть джерело даних «параметрична крива». Воно знадобиться при побудові зображення стрижня маятника.

Results – Data Sets: RMB > More Data Sets - Parameterized Curve 2D

Data Sets – Parameterized Curve 2D 1:

Data – **Data set** = Study 1/Solution 1

Parameter – **Name** = s

– **Minimum** = 0

– **Maximum** = 1

Expressions – **x:** = s*sin(comp1.p)

– **y:** = -s*cos(comp1.p)

LMB > Plot

Тепер створіть двовимірну графічну групу з вузлами типу Line (для зображення стрижня) і Point Trajectories (для зображення маси).

Results -> Plot Group – 2D Plot Group

2D Plot Group 3:

Data – **Data set** = Study 1/Solution 1

RMB > Line

2D Plot Group 3 - Line 1:

Data – **Data set** = Parameterized Curve 2D 1

Expression – **Expression** = 1

Coloring and style – **Line type** = Tube

– **Tube radius expression** = 0.015

– **Coloring, Color** = Uniform, Red

Для зображення маси у вигляді точки зручно використати вузол типу Point Trajectories.

2D Plot Group 3 -> **More Plots** -> Point Trajectories

2D Plot Group 3 - Point Trajectories 1:

Data– **Data set** = From Parent

Trajectory Data– **Plot data** = Coordinates

Значення опції Coordinates зазвичай означає, що траєкторія точки буде задаватися параметричними рівняннями в полях виразів. Але у нас зображатися буде лише точка.

Trajectory Data – **x-expression** = sin(comp1.p)

– **y-expression** = -cos(comp1.p)

Останні дві формули враховують факт, що координатна вісь Y спрямована на рисунку вертикально вгору, а кут φ відраховується супротив годинникової стрілки від вертикального напрямку униз.

Налаштуйте стиль вузла так, щоб в ньому відображалася лише точка.

Coloring and Style –Line style –Type =None

– **Point style –Type** = Point

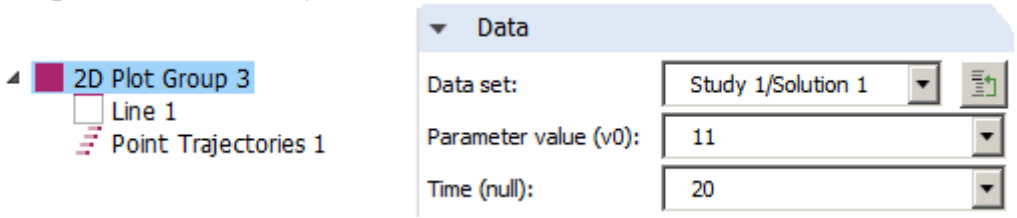
– **Point style –Point radius expression** = 0.1

– **Point style – Radius scale factor** = True,1

– **Point style – Color** = Red

LMB > Plot

Зверніть увагу на те, що це зображення отримано для останнього моменту часу **Time**=20 при початковій кутовій швидкості v0=11.



Побудуйте анімацію на базі створеної графічної групи.

Results –> Animation –> Player

Animation 1:

Scene – Subject = 2D Plot Group 3

Animation Editing – Loop over = Time

Анімація будуватиметься для випадку v0=9

– **Parameter value (vo)** = 9

– **Time selection** = Manual

– **Times (1-401)** = range(1,1,40) range(41,2,200)

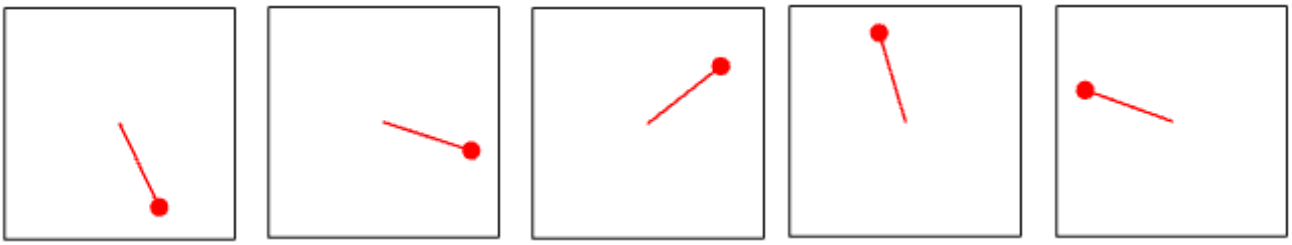
Для кадрів анімації використаємо перші 40 моментів часу, а подальші моменти братимемо через один.

Frames – Frame selection = All

LMB > Generate Frame

Graphics (графічне вікно) : LMB > Play

На наступному рисунку наведені зображення маятника, отримані при початковій кутовій швидкості v0=9 в моменти часу t=0.05,0.15,0.3, 0.55,0.75.

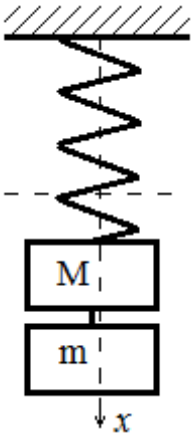


0.05 0.15 0.3 0.55 0.75

Зауваження. Можна було б не використовувати другий компонент. Але тоді з самого початку перший компонент треба було б створити не нульвимірним, а двовимірним.

4.4.3. Пружина з відривним вантажем.

На кінці невагомої пружини жорсткості c висить вантаж масою M. В момент t_1 $(t_1 > 0)$ до вантажу M раптово підвішується вантаж масою m, так що система починає коливатися. В момент t_2 $(t_2 > t_1)$ вантаж m зривається і пружина з масою M на кінці продовжує виконувати вільні коливання.



Якщо відраховувати зміщення $x(t)$ маси M від положення рівноваги вантажу, то воно задовольнятиме диференціальному рівнянню

$$\frac{d^2x}{dt^2} + k^2 x = \frac{1}{M} f(t) \ , \ t > 0 \quad \left(k^2 = \frac{c}{M}\right), \tag{1}$$

де $f(t)$ кусково задана функція. Для початкових умов $x(0) = x_0$, $x'_t(0) = v_0$ розв'язок задачі має вигляд

$$x(t) = x_0 \cos kt + \frac{v_0}{k} \sin kt + \frac{1}{M\,k} \int_0^t f(\tau) \sin k(t-\tau) d\tau \,.$$

В нашому прикладі збурювальний член $f(t)$ є функцією сходинкою, яка дорівнює 0 при $t \le t_1$, дорівнює mg при $t_1 < t \le t_2$, і дорівнює 0 для $t > t_2$. Тоді після інтегрування останній вираз набуває вигляду

$$x(t) = x_0 \cos kt + \frac{v_0}{k} \sin kt + \frac{m\,g}{M\,k^2} \left(\cos k \left(t - \frac{t_1 + t_2 + |t - t_1| - |t - t_2|}{2} \right) - \cos k(t - t_1) \right), \tag{2}$$

який можна буде порівняти з розв'язком задачі (1), отриманим в COMSOL.

Найпростіший спосіб розв'язати задачу полягає в створенні кускової функції $f(t)$, і розв'язанні ЗДР (1) за допомогою `Global ODEs and DAEs` інтерфейсу. Але ми збираємося продемонструвати «інструменти» розв'язання задачі по крокам. На першому кроці $0 \le t \le t_1$ буде розв'язуватися рівняння (1) з нульовою правою частиною і початковими умовами $x(0) = x_0$, $x_t'(0) = v_0$. На другому, при $t_1 < t \le t_2$, розв'язуватиметься рівняння (1) з $f(t) = mg$ і початковими умовами, отриманими з кінцевих значень (в момент часу t_1) розв'язку першої задачі. На третьому кроці розв'язуватиметься рівняння (1) з нульовою правою частиною і початковими умовами, які співпадатимуть з кінцевими значеннями (в момент часу t_2) розв'язку другої задачі.

Розв'язання 1. Створіть модель задачі з `Global ODEs and DAEs` інтерфейсом та «порожнім» дослідженням (`Empty Study`).

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) ->
Custom Studies - Empty Study

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

В гілці **Global Definitions** створіть вузол **Parameters**.

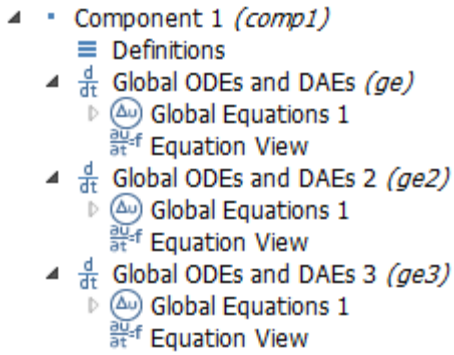
Global Definitions: RMB > Parameters

`Parameters` – **Name, Expression** = `m, 1`
– **Name, Expression** = `M, 1`
– **Name, Expression** = `k, 3`
– **Name, Expression** = `g, 10`
– **Name, Expression** = `x0,1`
– **Name, Expression** = `v0,1`
– **Name, Expression** = `t1,1`
– **Name, Expression** = `t2,4`

Додайте в компонент **Component 1** ще дві «фізики» - два `Global ODEs and DAEs` вузли.

Component 1: RMB > Add Physics = `Add Physics` -> Mathematics –
ODE and DAE Interfaces - Global ODEs and DAEs (ge)

Component 1: RMB > Add Physics = `Add Physics` -> Mathematics –
ODE and DAE Interfaces - Global ODEs and DAEs (ge) ->X



Маючи три «фізики», в гілці **Study 1** для кожної з них ми зможемо створити власний вузол/крок обчислювань. Але спочатку у вузлах `Global ODEs and DAEs` задайте власні рівняння.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u,ut,utt,t)** = `X1, X1tt+k^2*X1`

– **Initial value(u_0), Initial value(u_t0)** `=x0,v0`

Global ODEs and DAEs 2(ge2) - Global Equations 1:

`Global Equations` – **Name, f(u,ut,utt,t)** = `X2, X2tt+k^2*X2-m*g/M`

– **Initial value(u_0), Initial value(u_t0)** `=X1,X1t`

Останнім ми сказали, що початкові значення для величини `X2` будуть братися з розв'язку `X1` (але щоб так було цього замало).

Global ODEs and DAEs 3(ge3) - Global Equations 1:

`Global Equations` – **Name, f(u,ut,utt,t)** = `X3, X3tt+k^2*X3`

– **Initial value(u_0), Initial value(u_t0)** `=X2,X2t`

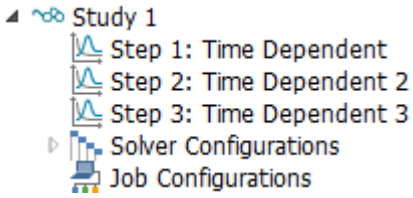
Початкові значення для величини `X3` будуть братися з розв'язку `X2`.

Створіть в гілці **Study 1** три вузли кроків досліджень.

Study 1: RMB > Study Steps - Time Dependent - Time Dependent

Study 1: RMB > Study Steps - Time Dependent - Time Dependent

Study 1: RMB > Study Steps - Time Dependent - Time Dependent



В кожному з вузлів-кроків потрібно задати діапазон часу, на якому він діятиме, та вказати фізику, яку він обчислюватиме. Для другого і третього кроків потрібно також вказати, що їх початкові значення братимуться з кінцевих значень попереднього кроку.

Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.1,t1)`

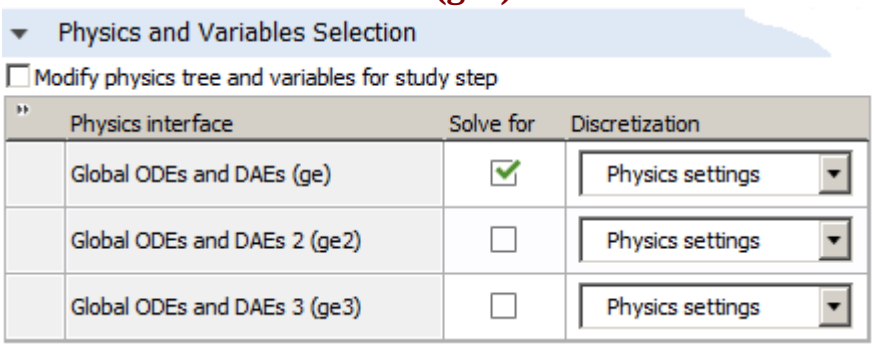
– **Relative tolerance** = `True, 0.0001`

`Physics and Variables Selection (таблиця)`-

Global ODEs and DAEs(ge) – Solve for = `True`

Global ODEs and DAEs 2(ge2) – Solve for = `False`

Global ODEs and DAEs 3(ge3) – Solve for = `False`



Останні налаштування означають, що на першому кроці розв'язуватися буде лише перша задача.

Step 2: Time Dependent 2:

Study Settings – **Times** = range(t1,0.1,t2)

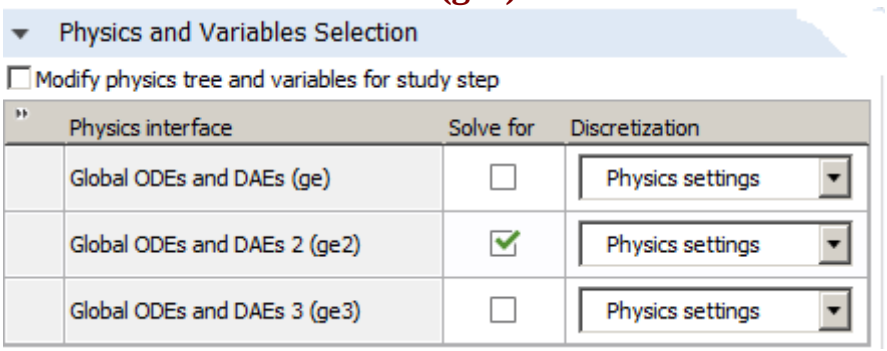
– **Relative tolerance** = True, 0.0001

Physics and Variables Selection (таблиця)-

Global ODEs and DAEs(ge) – Solve for = False

Global ODEs and DAEs 2(ge2) – Solve for = True

Global ODEs and DAEs 3(ge3) – Solve for = False



Задані налаштування вказують, що на поточному кроці розв'язуватися буде друга задача **Global ODEs and DAEs 2(ge2)**.

Щоб початкове значення другого кроку вибиралося по результатам роботи першого кроку, треба встановити прапорець

Values of Dependent Variables –

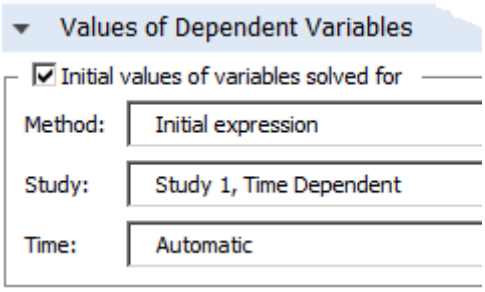
– **Initial values of variables solved for** = True

Щоб визначити, як використовувати дані попереднього кроку, в розділі **Values of Dependent Variables** є список **Method**. Обране в ньому значення Initial expression визначає, що дані будуть використовуватися для початкових значень поточного кроку обчислень. В списку **Study** визначається з якого кроку дослідження ці дані будуть обиратися.

Values of Dependent Variables – **Study** =

Study 1: Time Dependent

В списку **Time** (при динамічному дослідженні) вказується в який момент часу будуть обиратися дані попереднього кроку. Вибір Automatic (за замовчуванням) визначає вибір даних в останній момент кроку, вказаного в списку **Study**.



Нагадаємо, що у вузлі **Global ODEs and DAEs 2(ge2) - Global Equations 1** в таблиці рівнянь для шуканої змінної X2 ми вказали початкові значення X2(t0)=X1, X2'(t0)=X1t, які беруться з першого кроку обчислень.

Налаштування розділу **Values of Dependent Variables** визначають, що це будуть значення X1 і X1t, взяті в останній момент часу цього кроку.

Налаштуйте третій крок дослідження.

Step 3: Time Dependent 3:

Study Settings – **Times** = range(t2,0.1,10)

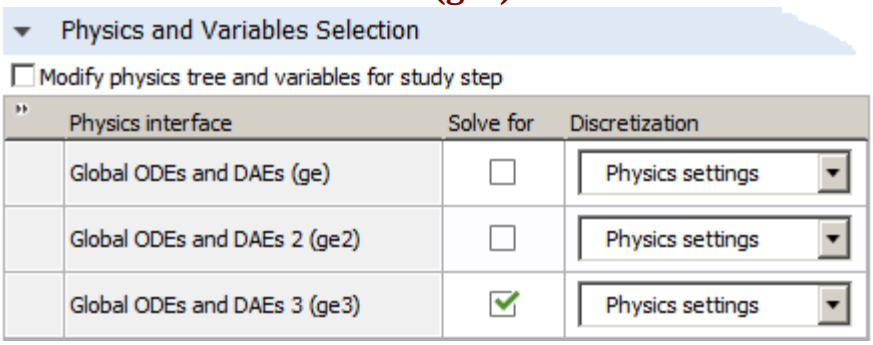
– **Relative tolerance** = True, 0.0001

Physics and Variables Selection (таблиця)-

Global ODEs and DAEs(ge) – Solve for = False

Global ODEs and DAEs 2(ge2) – Solve for = False

Global ODEs and DAEs 3(ge3) – Solve for = True

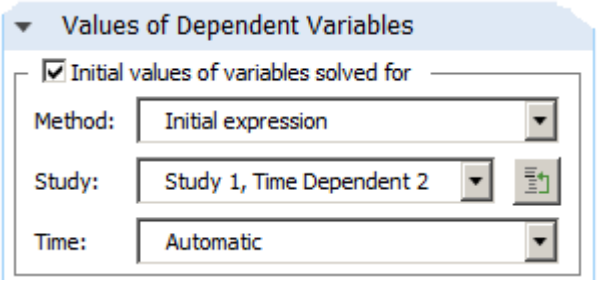


Values of Dependent Variables –

– **Initial values of variables solved for** = True

Values of Dependent Variables – **Study** =

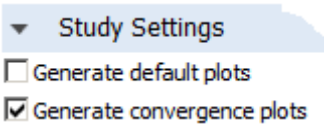
Study 1: Time Dependent 2



Якщо зараз викликати команду Compute, то будуть виконані обчислення всіх трьох кроків, але система залишить собі (і нам) лише дані останнього кроку. Але ми бажаємо побудувати графік розв'язку на всьому часовому інтервалі $0 \le t \le 10$. Тому будемо розв'язувати задачу по крокам, і для кожного кроку зберігатимемо дані розв'язку в таблицях. Маючи три таблиці, ми побудуємо по ним графіки, які в спільній графічній групі поєднаються в один графік розв'язку.

Щоб в гілці **Results** система не створювала зайвих вузлів, зніміть прапорець автоматичної побудови графіків в ПВ вузла **Study 1**.

Study 1: Study Settings – **Generate default plots** = False



Виконайте обчислення першого кроку.

Step 1: Time Dependent: RMB > Compute Selected Step

Після завершення обчислень в гілці **Result** не з'явиться жодного вузла (бо ми відмінили автоматичне генерування графіків).

Для запам'ятовування значень розв'язку X1 в гілці **Derived Values** створіть і обчисліть вузол типу Global Evaluation.

Derived Values : RMB > Global Evaluation

Label = Evaluation X1

Derived Values - Evaluation X1:

Expression – **Expression** = comp1.X1

LMB > Evaluate – New Table

Змініть назву створеної таблиці.

Label = Table X1

Дані, які надруковані в таблиці, залишаться і після виконання наступних кроків обчислень (а у вузлі **Evaluation X1** зміняться). Тому згодом ми зможемо використати їх для побудови графіка.

Виконайте обчислення другого кроку, для змінної X2 створіть вузол глобальних обчислень, і надрукуйте значення в таблицю.

Step 2: Time Dependent 2: RMB > Compute Selected Step

Derived Values : RMB > Global Evaluation

Label = Evaluation X2

Derived Values - Evaluation X2:

Expression – **Expression** = comp1.X2

LMB > Evaluate – New Table

Label = Table X2

Нарешті виконайте обчислення третього кроку, для невідомої X3 створіть вузол глобальних обчислень, і надрукуйте значення змінної в таблицю.

Step 3: Time Dependent 3: RMB > Compute Selected Step

Derived Values : RMB > Global Evaluation

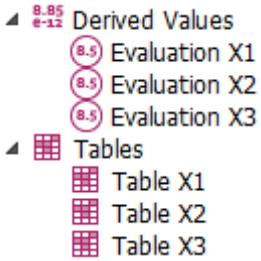
Label = Evaluation X3

Derived Values - Evaluation X3:

Expression – **Expression** = comp1.X3

LMB > Evaluate – New Table

Label = Table X3



Побудуйте графіки змінних X1, X2, X3, значення яких збережено в таблицях.

Results -> Plot Group - 1D Plot Group

1D Plot Group 1 -> Add Plot – Table Graph

1D Plot Group 1 - Table Graph 1:

Data – **Table** = Table X1

1D Plot Group 1 -> Add Plot – Table Graph

1D Plot Group 1 - Table Graph 2:

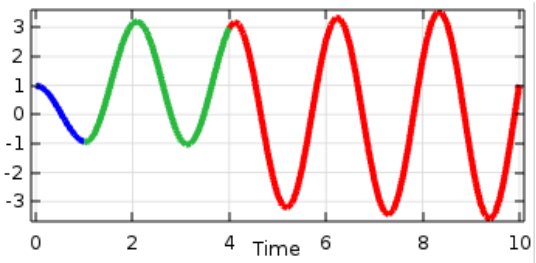
Data – **Table** = Table X2

1D Plot Group 1 –> Add Plot – Table Graph

1D Plot Group 1 - Table Graph 3:

Data – **Table** = Table X3

LMB > Plot



Побудуйте в цій графічній групі графік аналітичного розв'язку (2). Але спочатку створіть функцію для обчислення виразу, який записано в правій частині формули (2).

Global Definitions: RMB > Functions – Analitic.

Function name = f

Definition – **Expression** = x0*cos(k*t)+v0/k*sin(k*t)+ m*g/(M*k^2)*(cos(k*(t-(t1+t2+abs(t-t1)- abs(t-t2))/2))- cos(k*(t-t1)))

Definition – **Arguments** = t

Для побудови графіка функції нам знадобиться Grid 1D джерело даних. Створіть його.

Results - Data Sets: RMB> More Data Sets -> Grid 1D

Data Sets - Grid 1D 1 :

Data – **Source** = Function

– **Data set** = All

Paramemter Bounds – **Name** = t

– **Minimum, Maximum** = 0,10 (*два поля*)

Тепер додайте вузол лінійного графіка до нашої графічної групи

1D Plot Group 1 –> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

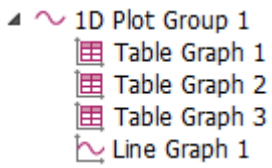
Data – **Data set** = Grid 1D 1

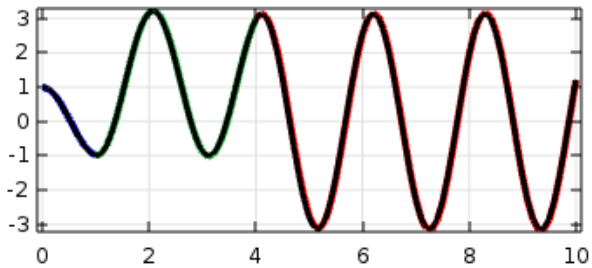
y-Axis Data – **Expression** = f(t)

x-Axis Data – **Parameter, Expression** = Expression,t

Coloring and Style – **Color** = Black

LMB > Plot (*наступний рисунок праворуч*)





Як бачите, криві співпадають.

Зауваження. Якщо у вузлах `Step X: Time Dependent X` залишити незмінною відносну похибку **Relative tolerance** = `0.01`, то наприкінці часового інтервалу буде спостерігатися невелике розходження кривих.

■

В інструментарії системи існує рішення, яке дозволяє вказати солверу моменти часу, в які він обов'язково повинен обчислювати розв'язок і, по потребі, стрибкоподібно змінити деякі змінні. Цим інструментом є інтерфейс `Events`. Продемонструємо його використання на поточному прикладі пружини з відривним вантажем.

Розв'язання 2 (використання інтерфейсу Events). Створіть модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

В гілці **Global Definitions** створіть вузол **Parameters**.

Global Definitions: RMB > Parameters

`Parameters` – **Name, Expression** = `m, 1`
– **Name, Expression** = `M, 1`
– **Name, Expression** = `k, 3`
– **Name, Expression** = `g, 10`
– **Name, Expression** = `x0,1`
– **Name, Expression** = `v0,1`
– **Name, Expression** = `t1,1`
– **Name, Expression** = `t2,4`

Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** =`X, Xtt+k^2*X-ZZ`
– **Initial value(u_0), Initial value(u_t0)** = `x0, v0`

Тут в якості правої частини рівняння (1) ми використали змінну `ZZ`, яку ще не створили. Вона буде міняти своє значення в моменти `t1` і `t2`.

Додайте в компонент інтерфейс `Events(ev)`. Він призначений для визначення в моделі контрольованих подій і їх обробки. Якщо моменти часу подій відомі заздалегідь (як в поточному прикладі), то в інтерфейсі `Events` для кожної події потрібно створити і налаштувати власний вузол `Explicit Event`. В цьому ж інтерфейсі можна (потрібно) створювати змінні, які будуть стрибкоподібно змінювати свої значення/вирази при наставанні події.

Додайте в компонент інтерфейс `Events(ev)`.

Component 1 (comp1): RMB > Add Physics = `Add Physics:` Events(ev) : X

В цьому інтерфейсі створіть вузол `Discrete States`, який призначено для визначення змінюваних змінних при наставанні подій. В нашому прикладі такою змінною буде `ZZ` - права частина рівняння (1).

Component 1 (comp1) - Events(ev) : RMB > Discrete States
Events(ev) - Discrete States 1 :

`Discrete States (таблиця)` – Name, Initial value (u0), = `ZZ, 0`

Discrete States

$u(t_0) = u_0$

Name	Initial value (u0)	Description
ZZ	0	0

Отже на початку процесу змінна `ZZ` дорівнюватиме нулю.

Тепер додайте в інтерфейс `Events(ev)` два `Explicit Event` вузли. Вони будуть містити вказівки на два моменти зовнішніх подій – моменти стрибків функції $f(t)$.

Component 1 (comp1) - Events(ev) : RMB > Explicit Event
Component 1 (comp1) - Events(ev) - Explicit Event 1 :

`Event Timing` – Start of event - t_i = `t1` (*момент першого стрибка*)
– Period of event - t_i = `10`

`Reinitialization (таблиця)` – Variable, Expression = `ZZ, m*g/M`

Останньою опцією в момент часу `t1` задається нове значення змінної `ZZ`.

Event Timings

Start of event:

t_i t1

Period of event:

T 10

Reinitialization

Variable	Expression
ZZ	m*g/M

Поле періоду нам не потрібно, але його не можна залишати порожнім. Тому ми вказали в ньому значення, яке співпадає з досліджуваним інтервалом часу.

Component 1 (comp1) - Events(ev) - Explicit Event 1 : RMB > Duplicate
Component 1 (comp1) - Events(ev) - Explicit Event 2 :

`Event Timing` – Start of event - t_i = `t2` (*момент другого стрибка*)

`Reinitialization (таблиця)` – Variable, Expression = `ZZ, 0`

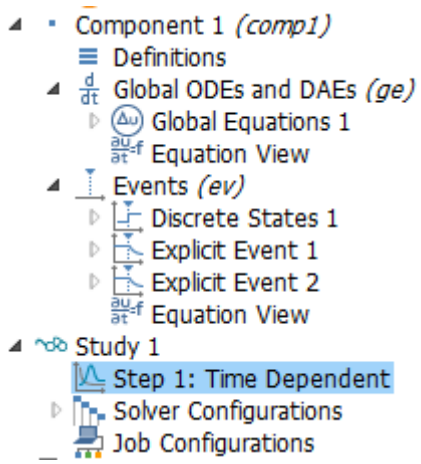
В момент часу `t2` значення змінної `ZZ` знову встановлюється в нуль.

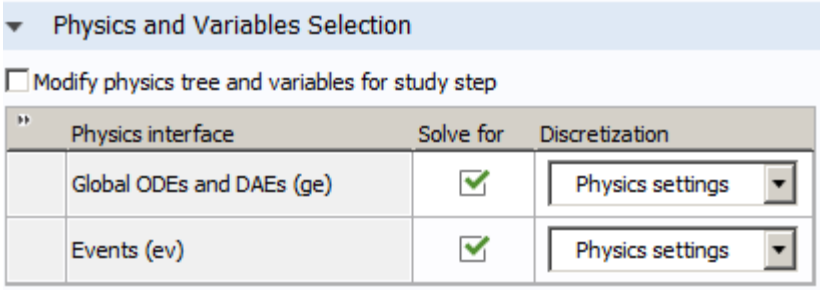
Нарешті задайте діапазон зміни параметра часу `t` і відносну похибку.

Step 1: Time Dependent:

`Study Settings` – Times = `range(0,0.1,10)`
– Relative tolerance = `True, 0.001`

Зверніть увагу на налаштування в розділі **`Physics and Variables Selection`**. Ними зазначено, що в обчислюваннях будуть використовуватися обидва інтерфейси: `Global ODEs and DAEs (ge)` і `Events(ev)`.





Виконайте обчислення.

`Home` -> Compute

В гілці **Results** створюється вузол **1 D Plot Group 1 - Global 1** з графіком розв'язку. Його стиль налаштуйте самостійно.

Тут же побудуйте графік аналітичного розв'язку (2). Для цього створіть функцію *f*, яка його представлятиме.

Global Definitions: RMB > Functions – Analitic.

Function name = `f`

Definition – **Expression** = `x0*cos(k*t)+v0/k*sin(k*t)+ m*g/(M*k^2)*(cos(k*(t-(t1+t2+abs(t-t1)- abs(t-t2))/2))- cos(k*(t-t1)))`

Definition – **Arguments** = `t`

Щоб функція стала доступною треба оновити розв'язок.

Study 1: LMB > Update Solution

Тепер побудуйте графік.

`1D Plot Group 1` -> Add Plot – Global

1D Plot Group 1 - Global 2:

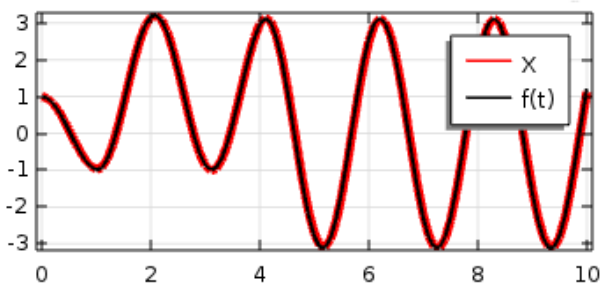
y-Axis Data(таблиця) – **Expression, Description** = `f(t),f(t)`

x-Axis Data – **Parameter, Expression** = `Expression, t`

Coloring and Style – **Line style** – **Color** = `Black`

– **Width** =2

LMB > Plot

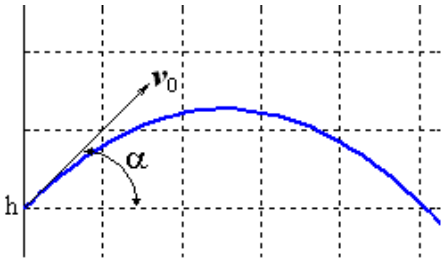


4.4.4. Рух тіла, кинутого під кутом до горизонту.

Розв'яжемо задачу Коші, яка описує рух тіла, кинутого з початковою швидкістю v_0 під кутом α до горизонту в припущенні, що опір повітря пропорційний квадрату швидкості. У векторній формі рівняння руху має вигляд

$$m\ddot{\mathbf{r}} = -\gamma \cdot \mathbf{v}\,|\,\mathbf{v}\,| - m\mathbf{g},$$

де $\mathbf{r}(t)$ – радіус-вектор, $\mathbf{v} = \dot{\mathbf{r}}(t)$ – вектор швидкості, γ – коефіцієнт опору, $m\mathbf{g}$ – вектор ваги тіла маси m, $\mathbf{g}$ – прискорення вільного падіння.



Особливість цієї задачі полягає в тому, що рух закінчується в заздалегідь невідомий момент часу, коли тіло падає на землю.

Якщо позначити $k = \gamma / m$, то в координатній формі ми маємо систему рівнянь

$$\ddot{x} = -k\,\dot{x}\sqrt{\dot{x}^2 + \dot{y}^2}$$

$$\ddot{y} = -k\,\dot{y}\sqrt{\dot{x}^2 + \dot{y}^2} - g\,,$$

до якої слід додати початкові умови: $x(0) = 0$, $y(0) = H$, (H початкова висота), $\dot{x}(0) = v_0 \cos\alpha$, $\dot{y}(0) = v_0 \sin\alpha$. Для модельної задачі покладемо `H=0, k=0.01, g=9.81, v0=10, α=π/4`.

Р о з в ' я з а н н я . Створіть нульвимірну модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics - ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

В гілці **Global Definition**s створіть вузол **Parameters**, і в його таблиці задайте параметри `K`, `v0`, `H`, *a* (кут) .

Global Definitions : RMB > Parameters

`Parameters`	– **Name, Expression**	= `K, 0.01`	(row1)
	– **Name, Expression**	= `v0, 10`	(row2)
	– **Name, Expression**	= `H, 0`	(row3)
	– **Name, Expression**	= `a, pi/4`	(row4)

Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – Name =`X`

– **f(u, ut, utt, t)=** `Xtt+K*Xt*sqrt(Xt^2+Yt^2+eps)`

– **Initial value(u_0), Initial value(u_t0)** = `0, v0*cos(a)`

`Global Equations` – Name =`Y`

– **f(u, ut, utt, t)=** `Ytt+K*Yt*sqrt(Xt^2+Yt^2+eps)+g_const`
– **Initial value(u_0), Initial value(u_t0)** = `H, v0*sin(a)`

Global ODEs and DAEs *(ge)*
Global Equations 1

Global Equations

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
X	Xtt+K*Xt*sqrt(Xt^2+Yt^2+eps)	0	v0*cos(a)
Y	Ytt+K*Yt*sqrt(Xt^2+Yt^2+eps)+g_const	H	v0*sin(a)

Зауваження. З урахуванням того, що внаслідок похибок обчислень під квадратним коренем може з'явитися від'ємне число, ми до його аргументу додали машинне епсілон $\approx 2.2204\cdot 10^{-16}$. Це (майже) не змінює результат обчислень, але запобігає виникненню випадкових помилок.

Задайте діапазон зміни параметра `t` і відносну похибку.

Step 1: Time Dependent:

Study Settings – **Times** = `range(0,0.1,1.5)`
– **Relative tolerance** = `True, 0.0001`

Виконайте обчислення.

`Home` -> Compute

Змініть налаштування створеного графічного вузла **Global 1** так, щоб він зображав траєкторію тіла.

1D Plot Group 1 :

Legend – **Show legends** = `False`

1D Plot Group 1 - Global 1:

y-Axis Data – **Expression** = `comp1.Y`

Рядок в таблиці виразів з формулою `comp1.X` видаліть, натиснувши кнопку Delete , яка розташована під таблицею.

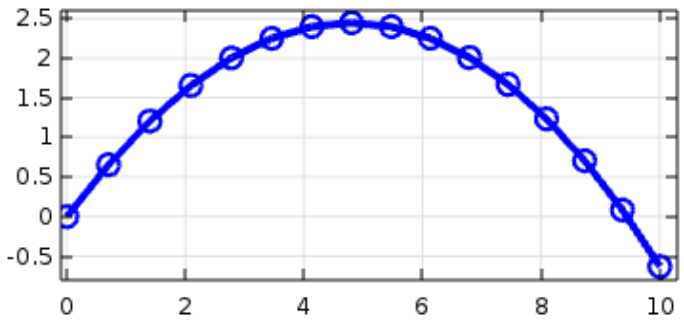
x-Axis Data – **Parameter, Expression** = `Expression, comp1.X`

Окрім кривої, буде зручно побудувати положення тіла в моменти часу, які визначені у вузлі **Step 1: Time Dependent** в полі **Times**.

Coloring and Style – **Line markers** – **Marker** = `Circle`
– **Line markers** – **Positioning** = `In data points`

Цим ми наказали системі відобразити положення тіла у вигляді кружків в моменти часу послідовності `range(0,0.1,1.5)`. Але пам'ятайте, що їх положення (кружків) все одно підбирається з інтерполяції по розрахунковим точкам, які солвер обирає самостійно. Дійсно, при налаштуванні солверів, призначених для розв'язання динамічних задач, ми зазвичай задаємо інтервал часу у вигляді діапазону з заданим кроком, наприклад, `range(0,0.1,1.5)`. Однак введений крок `0.1` не є визначальним. В першу чергу розмір кроку залежить від відносної похибки в полі **Relative tolerance**. Якщо з часом розв'язок змінюється швидко, то крок зменшується. В моменти повільної зміни розв'язку часовий крок збільшується.

LMB > Plot



Коли у вузлі **Time-Dependent Solver 1** в списку **Output - Times to store** обрано значення Specified values, то розв'язувач за замовчуванням синхронізовано з налаштуваннями вузла **Step 1: Time Dependent**. З цими налаштуваннями побудовано попередній графік, і точки на ньому зображені у моменти часу з послідовності range(0,0.1,1.5) (коли у графічному вузлі обрано опцію **Line markers – Positioning** = In data points). Якщо в списку **Output - Times to store** обрати значення Steps taken by solver, то будуть зберігатися (і відображатися) дані в моменти часу, обрані розв'язувачем.

Розрахунковий час 1.5 тривалості руху підбирався нами «на око». Також «на око» доводиться визначати максимальну висоту і дальність польоту. Звичайно, ці значення слід обчислювати.

Спочатку налаштуємо розв'язувач таким чином, щоб він зупинявся, коли координата тіла Y стає від'ємною. Для цього в гілку **Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1** потрібно додати вузол Stop Condition, в якому задати умови зупинки.

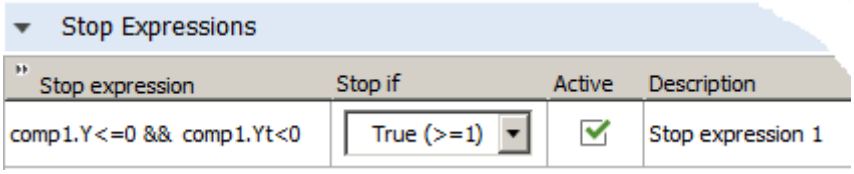
Time-Dependent Solver 1: RMB > Stop Condition

Time-Dependent Solver 1 - Stop Condition 1:

Додайте умову зупинки, натиснувши в зоні таблиці Stop Expressions праву кнопку миші. Оберіть з випадного списку команду Add, і введіть

Stop Expressions – Stop Expressions =comp1.Y<=0 && comp1.Yt<0

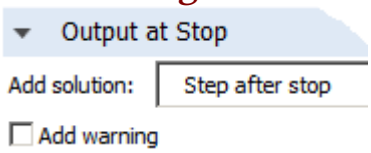
В кінцевій точці траєкторії величина Y дорівнює нулю, але вона дорівнює нулю і на початку руху. Тому, щоб обчислення не зупинилися відразу, доречно перевіряти дві умови $y \le 0$ і $y'_t < 0$ (ордината тіла зменшується).



Output at stop –Add solution = Steps after stop

Значення Step after stop використовується для зберігання (і відображення) розв'язку в момент часу, в який умова була виконана.

Output at stop –Add warning = False



Переобчисліть задачу.

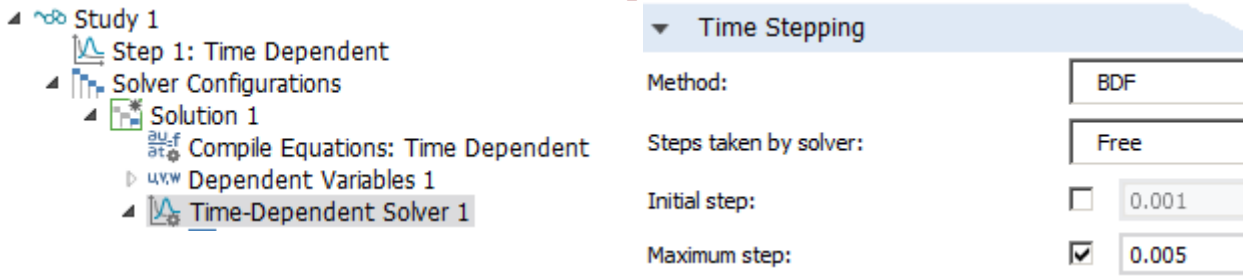
Home -> Compute

На графіку траєкторії мало що змінилося. Справа в тому, що при розв’язанні динамічних задач солвер підбирає часовий крок самостійно, виходячи з відносної похибки, а не з величини кроку, який задається користувачем. Там, де розв’язок змінюється швидко, крок зменшується. На ділянках повільної зміни розв’язку часовий крок збільшується. У зв'язку з цим солвер може «проскочити» важливі для розв’язку задачі моменти часу. Наприклад, інтервал часу, на якому відбувається приземлення тіла, може бути занадто великим. Саме це спостерігається у нашому випадку.

У користувача є можливість управління максимальною довжиною часових інтервалів, які використовує динамічний розв’язувач. Застосуємо її.

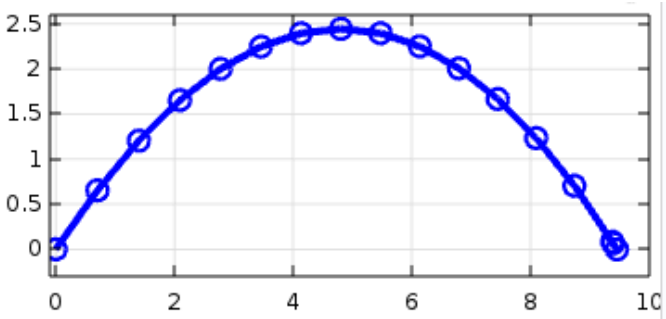
Time-Dependent Solver 1:

Time Stepping – **Maximum step** = True, 0.005



Знову обчисліть задачу.

Home -> Compute



Оскільки довжина часових інтервалів не перевищує 0.005, то відстань між двома останніми положеннями тіла (коли умова ще не виконалася, і коли вже виконалася) мала. Будь яке з цих положень можна вважати останнім, тобто тим, коли тіло впало на землю (тіло впало на землю в проміжок часу між цими двома моментами).

Надрукуйте дальність польоту.

Results -> **Derived Values** – Global Evaluation

Label = Distance

Derived Values - Distance:

Data – **Time selection** = Last

Expression – **Expression** = comp1.X

– **Description** = True, Distance

LMB > Evaluate

Буде надруковано час і координата X в останній запам'ятований момент (перший, в який виконалася умова `comp1.Y<=0 && comp1.Yt<0`), тобто тривалість руху і відстань.

Time	Distance
1.4164	9.4782

Надрукуйте максимальну висоту. Для цього можна використати оператор `timemax(t1,t2,expr)`, який визначає максимальне значення виразу `expr` на відрізку часу $t_1 \le t \le t_2$, а в якості виразу використати ординату $y(t)$ тіла.

`Results` -> **`Derived Values`** – Global Evaluation

Label = `Height`

Derived Values - Height:

`Data` – **Time selection** = `Last`

`Expression` – **Expression** = `timemax(0,t,comp1.Y)`

– **Description** = `True, Height`

LMB > Evaluate – Table 1

Time	Distance	Height
1.4164	9.4782	2.4463

Інший спосіб надрукувати максимальну висоту полягає у виборі операції `Maximum` в розділі **`Data Series Operation`** вузла глобальних обчислень.

Derived Values - Height: RMB > Duplicate

Derived Values – Height 1:

`Data` – **Time selection** = `All`

`Expression` – **Expression** = `comp1.Y`

`Data Series Operation` – **Operation** = `Maximum`

LMB > Evaluate – New Table

Maximum: Height
2.4463

Використовуючи оператор `attimemax(...)` можна надрукувати момент часу, в який досягається найвища точка траєкторії. Нагадаємо, що оператор `attimemax(t1,t2,expr1,expr2)` шукає момент часу t_0 $(t_1 \le t_0 \le t_2)$, в який вираз `expr1` досягає максимуму, і в цей момент t_0 обчислює вираз `expr2`.

`Results` -> **`Derived Values`** – Global Evaluation

Label = `MaxHeightTime`

Derived Values - MaxHeightTime:

`Data` – **Time selection** = `Last`

`Expression` – **Expression** = `attimemax(0,t,comp1.Y,t)`

Зверніть увагу на різницю тлумачення ідентифікатора `t` в останньому виразі: перший `t` – це час польоту (`Last`), другий `t` – це вираз, який буде обчислюватися.

– **Description** = `True, Moment of maximum height`

`Data Series Operation` – **Operation** = `None`

LMB > Evaluate – Table 1

Time	Distance	Height	Moment of maximum height
1.4164	9.4782	2.4463	0.69984

Виконаємо дослідження залежності відстані польоту від коефіцієнта опору середовища. Для цього застосуємо допоміжний аналіз (`Auxiliary Sweep`) по параметру `K` - коефіцієнту опору середовища. Наприклад, розв'яжемо задачу при `K=0.1,0.2,0.3,0.5`.

Study 1 – Step 1: Time Dependent:

`Study Extensions` – Auxiliary sweep = `True`

В розділі **`Study Extensions`** активується таблиця вибору параметрів. Клацніть по таблиці правою кнопкою миші і з випадного меню виберіть команду Add. Зі списку, який з'явиться в стовпці **Parameter name**, виберіть параметр `K`.

`Study Extensions` – Parameter name = `K`

У другому стовпці таблиці задайте діапазон зміни цього параметра.

`Study Extensions` – Parameter value list = `0.1 0.2 0.3 0.5`

Переобчисліть модель.

`Home` -> Compute

Графік в гілці **1D Plot Group 1** стане іншим. Змініть в ньому налаштування.

1D Plot Group 1 :

`Legend` – Show legends = `True`

`Axis` – Manual axis limits = `True`

– x maximum = `6.5`

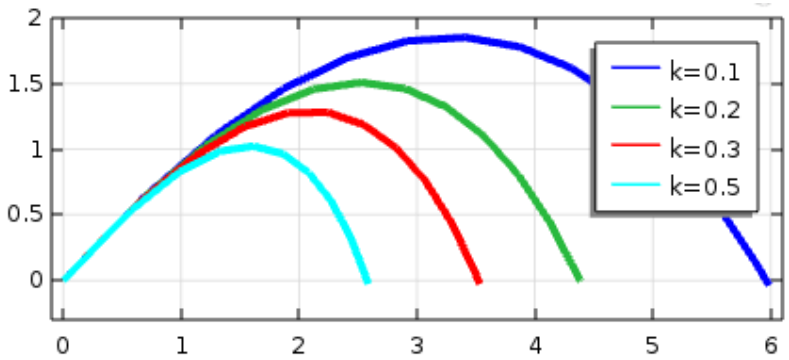
– y maximum = `2`

1D Plot Group 1 - Global 1:

`y-Axis Data` (таблиця) – Description = (*все видалити*)

`Coloring and Style` – Line markers – Marker = `None`

LMB > Plot



Надрукуйте для цих коефіцієнтів тривалість і дальність польоту.

Derived Values - Distance: LMB > Evaluate - New Table

K	t	Distance
0.10000	1.2264	5.9653
0.20000	1.1014	4.3826
0.30000	1.0164	3.5206
0.50000	0.90640	2.5837

Надрукуйте максимальну висоту польоту. Це потрібно робити окремо/послідовно для кожного значення параметра K.

Derived Values – Height 1:

Data – **Parameter selection (K)** = Manual

Data – **Parameter values (K) (1-4)**= 1

LMB > Evaluate – New Table

Data – **Parameter values (K) (1-4)**= 2

LMB > Evaluate –Table 4

Data – **Parameter values (K) (1-4)**= 3

LMB > Evaluate –Table 4

Data – **Parameter values (K) (1-4)**= 4

LMB > Evaluate –Table 4

Maximum: Height	Maximum: Height	Maximum: Height	Maximum: Height
1.8520	1.5091	1.2803	1.0238

Зауваження. Вузли **Derived Values – Height** і **Derived Values – MaxHeightTime** використати для обчислення максимальної висоти і відповідного моменту часу неможливо, бо оператори timemax і attimemax з розв'язками, які окрім часу залежать від параметрів моделі, не працюють.

Дізнатися час, в який досягається максимальна висота, можна, якщо розв'язати у вузлі **Global Equations 1** відносно невідомої *Th* задачу $\frac{dT_h}{dt} = (Y_t{}' > 0)$ з вільним членом, заданим логічним виразом, і нульовою початковою умовою. Доки траєкторія піднімається, тобто $Y_t{}' > 0$, вільний член дорівнюватиме одиниці, а змінна *Th* дорівнюватиме часу *t*. Але коли функція *Y*(*t*) починає спадати (тобто $Y_t' < 0$) вільний член дорівнюватиме нулю, і величина *Th* перестає змінюватися. Отже наприкінці процесу *Th* дорівнюватиме тому значенню, яке вона набула в момент зміни знака Y_t', тобто моменту максимальної висоти польоту (насправді останньому моменту часу, коли траєкторія ще зростала).

Додайте в таблицю рівнянь вказану задачу.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name** =Th

– **f(u, ut, utt, t)=** Tht-(Yt>0)

– **Initial value(u_0)** = 0

Global Equations

$f(u,u_t,u_{tt},t) = 0,\ u(t_0) = u_0,\ u_t(t_0) = u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
X	Xtt+K*Xt*sqrt(Xt^2+Yt^2+eps)	0	v0*cos(a)
Y	Ytt+K*Yt*sqrt(Xt^2+Yt^2+eps)+g_const	H	v0*sin(a)
Th	Tht-(Yt>0)	0	0

Переобчисліть модель і надрукуйте знайдені значення.

Home -> Compute

Derived Values - MaxHeightTime:

Data – **Parameter selection(K)** = All

Data – **Time selection** = Last

Expression – **Expression** = comp1.Th

– **Description** = True, MaxHeightTime

Data Series Operation – **Operation** = None
LMB > Evaluate – New Table

K	t	MaxHeightTime
0.10000	1.2278	0.57459
0.20000	1.1028	0.49959
0.30000	1.0178	0.44959
0.50000	0.90280	0.38459

В другій колонці надрукована тривалість польоту, а в третьому – момент часу, в який досягається максимальна висота.

Визначення моментів часу, в які досліджувані величини набувають зумовленого значення, зустрічається досить часто. Фактично це є деякими подіями, які потрібно відстежувати під час розв'язання задач. Для цього в системі існує допоміжний інтерфейс Events (Події). Його можна застосовувати для тимчасової зупинки розв'язувача, і модифікації або запам'ятовування змінних при подальших обчисленнях. При чому сам момент здійснення події може бути заданий явно (у відомий момент часу) або неявно (деякою логічною умовою).

Для тренування користувача покажемо використання цього інтерфейсу для визначення максимальної висоти траєкторії і моменту часу, в який вона досягається.

Додайте в дерево моделі интерфейс Events(ev).

Component 1 (comp1): RMB > Add Physics = Add Physics: Events (ev) : X

Якщо моменти часу подій невідомі заздалегідь, але відомі умови, при яких вони відбуваються, то в інтерфейсі Events для кожної з них потрібно створити і налаштувати власний вузол Implicit Event. Але спочатку для цього в інтерфейсі Events у вузлі Indicator States потрібно створити змінні, які розв'язувач буде контролювати на кожному часовому кроці. Їх (змінні) солвер буде використовувати при перевірці умов у вузлах Implicit Event неявно визначених подій.

Створіть і налаштуйте вузол Indicator States. В нашому прикладі контрольованою змінною буде Z1 – вертикальна швидкість. Звичайно, що вона буде співпадати з обчислюваною в моделі величиною Yt.

Component 1 (comp1) - Events(ev) : RMB > Indicator States

Events(ev) - Indicator States 1 :

В таблиці **Indicator Variables** в полі **Name** задаються імена контрольованих змінних, в полі **g(v,vt,vtt,t)** вводяться вирази для їх обчислення, а в полі **Initial value(uo)** задаються початкові значення

Indicator Variables (таблиця) – **Name, g(v,vt,vtt,t)** = Z1, Yt
– **Initial value(u0)**= v0*sin(a)

Indicator States

Label: Indicator States 1

Indicator Variables

$\mathrm{nojac}(g(v,v_t,v_{tt},t)) - u = 0,\ u(t_0) = u_0$

Name	g(v,vt,vtt,t)	Initial value (u0)
Z1	Yt	v0*sin(a)

Тепер створіть і налаштуйте вузол `Implicit Event` (Неявні події). Ми використовуємо цей вузол тому, що нам невідомий точний час, коли буде досягнута найвища точка траєкторії, однак ми знаємо умову, при якій це трапиться ($Y_t' = 0$).

Component 1 (comp1) - Events(ev) : RMB > Implicit Event

Events(ev) - Implicit Event 1 :

В момент найвищої висоти вертикальна швидкість Y_t' тіла дорівнює нулю, але при чисельних розрахунках це неможливо точно перевірити. Та можна перевірити умову зміни знака вертикальної швидкості Y_t'.

`Event Conditions` – Condition = `Z1<0`

Майте на увазі, що умова в полі **Condition** може бути розпізнана лише при пересіченні створеною змінною `Z1` нуля, коли логічна умова змінює своє значення з хибного на істинне. Це означає, наприклад, що умова типу `Z1<0` може бути ідентифікована лише при пересіченні змінною `Z1` нуля, а умова `Z1<1` (або `Z1-1<0`) може взагалі бути не розпізнаною.

Повторимо ще раз. Неявна подія розпізнається тільки при пересіканні Indicator States змінних нульового значення. В нашому випадку для `Z1=Yt` це трапиться лише один раз в точці максимальної висоти.

Коли подія трапляється, то у вузлах `Implicit Event` в таблиці розділу **`Reinitialization`** можна запам'ятати значення деяких змінних. В нашому прикладі ці змінні слід ще створити. Змінні, які стрибкоподібно змінюють свої значення при виникненні неявної події, можна створювати у вузлах типу `Discrete States` інтеерфейсу `Events`.

Component 1 (comp1) - Events(ev) : RMB > Discrete States

Events(ev) - Discrete States 1 :

`Discrete States(таблиця)` – Name, Initial value(u0) = `Ymax,0`

– Name, Initial value(u0) = `TYmax,0`

Цим ми створили дві змінні Ymax – максимальна висота, і TYmax – момент часу, в який досягається максимальна висота. На початку руху вони дорівнюють нулю.

Discrete States

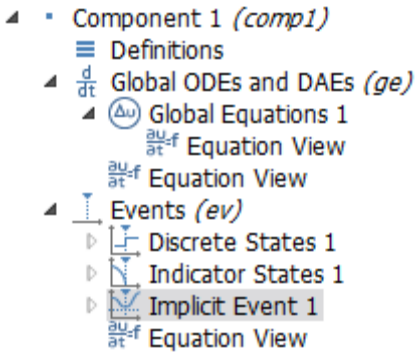
$u(t_0) = u_0$

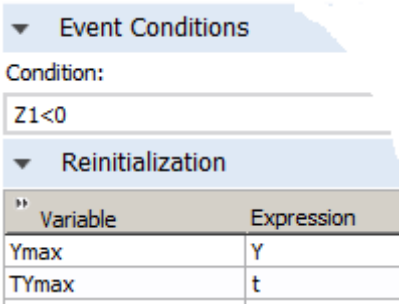
Name	Initial value (u0)	Description
Ymax	0	0
TYmax	0	0

Повернемося до ПВ вузла **Implicit Event 1** і введемо вирази, які отримують змінні `Ymax` і `TYmax` при відбуванні події.

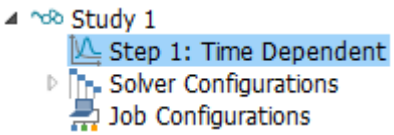
Events(ev) - Implicit Event 1 :

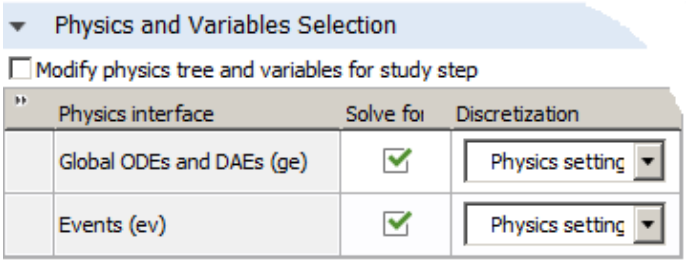
Reinitialization(таблиця) – **Variable, Expression** = Ymax,Y
– **Variable, Expression** = TYmax,t





Відкрийте ПВ вузла **Step 1: Time Dependent** і зверніть увагу на налаштування в розділі **Physics and Variables Selection**. Ними зазначено, що одночасно будуть виконуватися обидва інтерфейси: Global ODEs and DAEs (ge) і Events (ev).





Виконайте обчислення.

Home -> Compute

Для друку максимальної висоти і відповідного моменту часу в гілці **Derived Values** створіть два Global Evaluation вузли, в яких задайте вирази comp1.Ymax і comp1.TYmax.

Results -> **Derived Values** – Global Evaluation
Label = MaxHeightTime 2

Derived Values – MaxHeightTime 2:

Data – **Parameter selection(K)** = All
Data – **Time selection** = Last
Expression – **Expression** = comp1.TYmax
– **Description** = True, THmax

LMB > Evaluate – New Table

Derived Values - MaxHeightTime 2: RMB > Duplicate
Label = MaxHeight

Derived Values – MaxHeight:

Expression – **Expression** = comp1.Ymax
– **Description** = True, MaxHeight

LMB > Evaluate –Table 6

K	t	THmax	MaxHeight
0.10000	1.2289	0.57650	1.8548
0.20000	1.1050	0.50204	1.5091
0.30000	1.0199	0.45153	1.2918
0.50000	0.90352	0.38964	1.0246

4.4.5. Рух пружного м’яча.

Для пружного м'яча відоме початкове положення і швидкість. Опір повітря нехтовно малий, але енергія руху витрачається при відскоку м'яча від землі. Нехай при відскоку від землі вертикальна швидкість м'яча становить 90% від вертикальної швидкості в момент падіння. Змоделюємо такий рух.

Рівняння руху (без урахування опору повітря) має вигляд

$$m\ddot{\mathbf{r}} = -m\mathbf{g},$$

де $\mathbf{g}$ – вектор прискорення вільного падіння з напрямком вертикально вниз. У покоординатній формі рівняння набирає вигляду $\ddot{x} = 0$, $\ddot{y} = -g$. Задачу слід доповнити початковими умовами $x(0) = x_0, y(0) = y_0$, $\dot{x}(0) = v_0^x$, $\dot{y}(0) = v_0^y$. Система розпадається на два незалежних рівняння, перше з яких має розв’язок $x(t) = x_0 + v_0^x \cdot t$. Це вказує на те, що в горизонтальному напрямку рух відбувається зі сталою швидкістю v_0^x. Друге рівняння також інтегрується, але ми хочемо показати, як його можна розв’язати чисельно враховуючи, що в момент відскоку t_k вертикальна швидкість $\dot{y}$ змінює знак так, що $\dot{y}_{after}(t_k) = -0.9 \cdot \dot{y}_{before}(t_k)$. Ми будемо моделювати рух при багаторазовому відскоку і приземленні м'яча.

В нашому прикладі покладемо $x_0 = 0, y_0 = 0$, а також `g=9.81, v0x=1, v0y=10.`

Р о з в ’ я з а н н я . Створіть нульвимірну модель задачі з динамічним Global ODEs and DAEs інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

В гілці **Global Definition**s створіть вузол **Parameters**, і в його таблиці задайте параметри `x0, g, v0x` (початкова горизонтальна швидкість), `v0y` (початкова вертикальна швидкість)

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `x0,0` (row1)
– **Name, Expression** = `g,9.81` (row2)
– **Name, Expression** = `v0x,1` (row3)
– **Name, Expression** = `v0y,10` (row4)

Заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – **Name, f(u, ut, utt, t)** =`X, Xtt`
– **Initial value(u_0), Initial value(u_t0)** = `x0, v0x`

`Global Equations` – **Name, f(u, ut, utt, t)** =`Y, Ytt+g`
– **Initial value(u_0), Initial value(u_t0)** = `0, v0y`

$f(u,u_t,u_{tt},t) = 0,\ u(t_0) = u_0,\ u_t(t_0) = u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
X	Xtt	x0	v0x
Y	Ytt+g	0	v0y

Щоб аналізувати відскік м’яча від землі, в модель потрібно додати інтерфейс `Events(ev)`. Він застосовується для обробки в динамічних обчислюваннях контрольованих подій. Якщо моменти часу заздалегідь невідомі, а події відбуваються при виконанні деяких умов, то в інтерфейсі `Events` для кожної з них потрібно створити і налаштувати власний вузол `Implicit Event`. Але перед цим в інтерфейсі `Events` у вузлі `Indicator States` потрібно створити змінні, значення яких розв’язувач буде контролювати на кожному кроці.

Додайте в компонент інтерфейс `Events(ev)`.

Component 1 (comp1): RMB > Add Physics = `Add Physics`: Events (ev) : X

Створіть і налаштуйте вузол `Indicator States`. Його призначено для визначення контрольованих змінних. В нашому прикладі такою змінною буде `Z` – вертикальна координата м’яча. Вона буде співпадати з обчислюваною в моделі величиною `Y`.

Component 1 (comp1) - Events(ev) : RMB > Indicator States

В ПВ створеного вузла в таблиці **`Indicator Variables`** в полі **Name** задаються імена контрольованих змінних, в полі **g(v,vt,vtt,t)** вводяться вирази для їх обчислення, а в полі **Initial value(uo)** задаються початкові значення.

Component 1 (comp1) - Events(ev) - Indicator States 1 :

`Indicator Variables (таблиця)` – **Name, g(v,vt,vtt,t)** = `Z, Y`

– **Initial value(u0)**= `0`

Indicator States

Label: Indicator States 1

Indicator Variables

nojac$(g(v,v_t,v_{tt},t))$ - $u = 0,\ u(t_0) = u_0$

Name	g(v,vt,vtt,t)	Initial value (u0)
Z	Y	0

Створіть і налаштуйте вузол неявних подій `Implicit Event`.

Component 1 (comp1) - Events(ev) : RMB > Implicit Event

Events(ev) - Implicit Event 1 :

В момент відскоку м’яча від землі його вертикальна координата дорівнює нулю, але при чисельних розрахунках це неможливо перевірити, бо завжди є похибка. Тому наша модель буде реагувати на подію становлення вертикальної координати від’ємною (точніше на зміну її знака)..

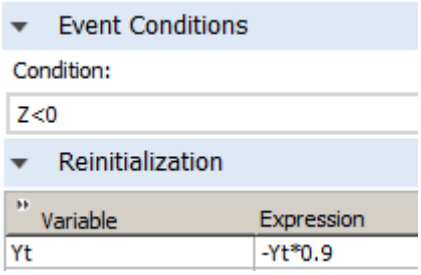
`Event Conditions` – **Condition** = `Z<0`

Нагадаємо, що умови в полі **Condition** розпізнаються лише при пересіченні контрольованими змінними нульового значення.

При досягненні м'ячем землі солвер призупиняє роботу, попередньо запам'ятавши всі поточні дані розв’язку. Щоб продовжити роботу в наступні моменти часу, ми повинні відновити обчислення з новими даними, що відповідають моменту відскоку. Останні запам’ятовані солвером значення

коректуються відповідно до таблиці реініціалізації і використовуються для обчислення наступної гладкої ділянки траєкторії м'яча. При цьому вертикальна швидкість м'яча в момент відскоку стає позитивною і становить 90% від абсолютного значення вертикальної швидкості в момент падіння. В нашому прикладі в момент відскоку вертикальна швидкість змінює знак і величину по формулі $\dot{y}_{after}(t_k) = -0.9 \cdot \dot{y}_{before}(t_k)$, яка реалізується наступною інструкцією.

`Reinitialization(таблиця)` – Variable, Expression = `Yt, -Yt*0.9`

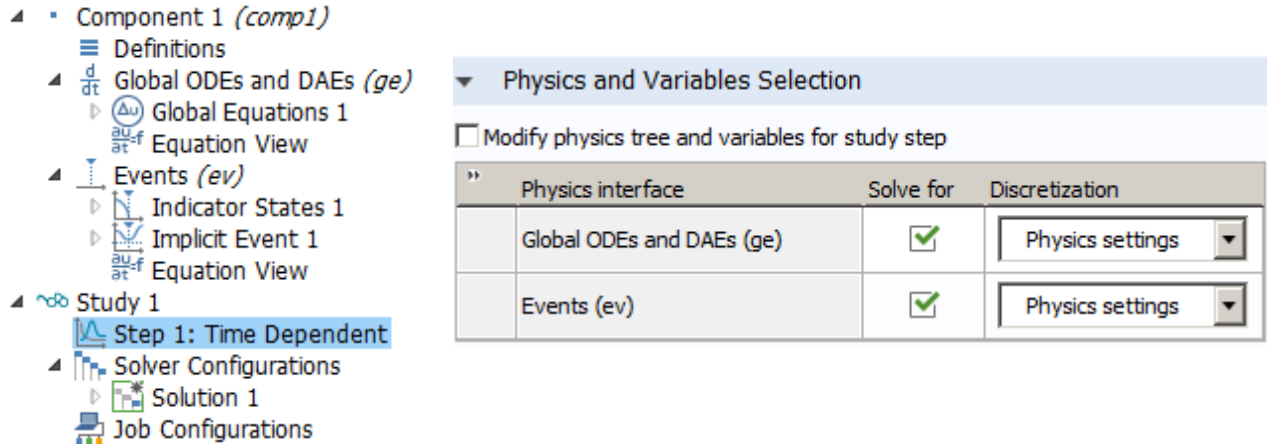


Залишилося налаштувати розв'язувач.

Step 1: Time Dependent:

`Study Settings` – Times = `range(0,0.1,10)`

Зверніть увагу на налаштування в розділі **`Physics and Variables Selection`**. Ними зазначено, що в обчислюваннях будуть використовуватися обидва інтерфейси: `Global ODEs and DAEs (ge)` і `Events(ev)`.



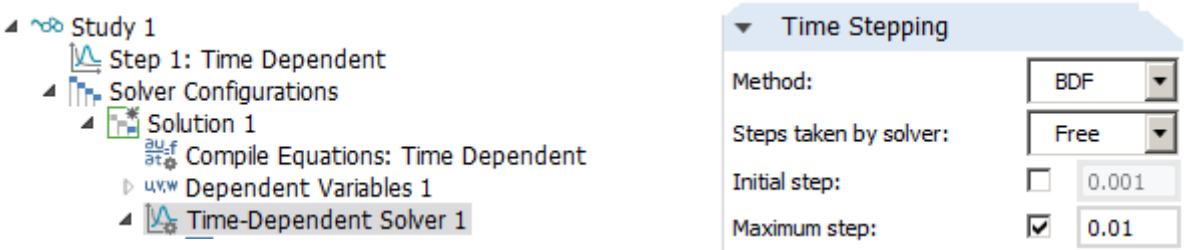
Щоб часові інтервали розв'язувача були короткими, зменшіть максимальну довжину його кроку по часу.

`Study` –> Solver - Show Default Solver.

У вузлі **Study 1** з'явиться кілька дочірніх вузлів і можна буде виконати інструкцію

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

`Time Stepping` – Maximum step = `True, 0.01`



Зауваження. Існує опція `Event tolerance` (значення за замовчуванням 0.01), яка визначає точність перевірки неявної умови у вузлах типу `Implicit Event`.

Виконайте обчислення.

`Home` **->** Compute

В гілці **Results** створюється вузол **1D Plot Group 1 - Global 1** з графіками величин X і Y. Але нас цікавить форма траєкторії. Змінимо налаштування цього вузла так, щоб він показував траєкторію.

1D Plot Group 1 :

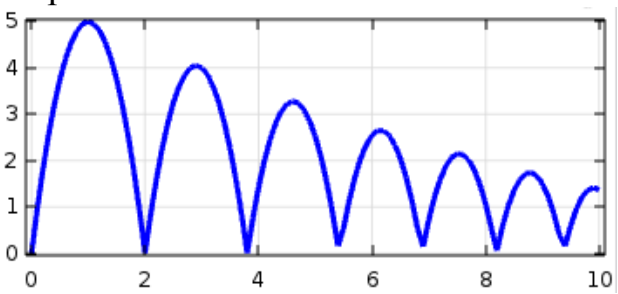
`Legend` – **Show legends** = `False`

1D Plot Group 1 - Global 1:

`y-Axis Data` – **Expression** = `comp1.Y`

`x-Axis Data` – **Parameter, Expression** = `Expression, comp1.X`

Останні вирази кажуть, що вертикальні координати кривої відповідають значенням `Y`, а горизонтальні - значенням `X`.



Показаний на попередньому рисунку графік траєкторії руху м'яча отримано при векторі початкової швидкості $\boldsymbol{v}(0)=\left(v_0^x, v_0^y\right)=(1,10)$.

4.4.6. Модель Лотки – Вольтерри.

Дослідимо двохвидову модель «хижак–жертва», вперше побудовану Вольтерра для пояснення коливань рибних уловів. Є два біологічні види, чисельністю в момент часу `t`, відповідно `x(t)` і `y(t)`. Особини першого виду є їжею для особин другого виду (хижаків). Чисельності популяцій в початковий момент часу відомі. Потрібно визначити чисельність видів в майбутні моменти часу. Математичною моделлю задачі є система диференціальних рівнянь Лотки – Вольтерра

$$\begin{cases} \dfrac{d\,x}{d\,t}=(a-b\cdot y)\,x \\ \dfrac{d\,y}{d\,t}=(-c+d\cdot x)\,y \end{cases},$$

де `a,b,c,d` – додатні сталі.

Проведемо визначення чисельності популяцій при $a=3, b=3, c=1, d=1$, для декількох варіантів початкових умов: $x(0)=2,\ y(0)=1$; $x(0)=1,\ y(0)=2$; $x(0)=1,\ y(0)=3$; $x(0)=3,\ y(0)=2$.

Для дослідження можна використати модель з динамічним Global ODEs and DAEs інтерфейсом. Але вигляд системи відповідає безмасовій редакції `Mathematical Particle Tracing` інтерфейсу, тому використаємо його.

Розв'язання. Створіть двовимірну динамічну модель з `Mathematical Particle Tracing` інтерфейсом.

File -> New -> Model Wizard -> 2D -> Mathematics -
Mathematical Particle Tracing (pt) -> Time Dependent

Обравши тип інтерфейсу `Mathematical Particle Tracing(pt)` (після натискання на кнопку Add) ви побачите в правій частині вікна `Select Physics` поля для введення назв змінних задачі. Ці імена пізніше будуть використовуватися під час запису системи диференціальних рівнянь. Залиште вміст `qx`, `qy`, `qz` цих полів без змін.

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Розрахунковою областю задачі вважатимемо прямокутник. Він представлятиме зону, в якій будуть розташовані фазові траєкторії. Щоб не замислюватися про можливість виходу кривих за межі цієї області, для прямокутника треба задати досить великі розміри.

`Geometry` -> **`Primitives`** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

`Size` – **Width, Height** = `8,5` (*два поля*)

`Position` – **Base, x, y** = `Corner,-1,-1` (*три поля*)

LMB > Build All Objects

В гілці **Global Definitions** створіть вузол параметрів.

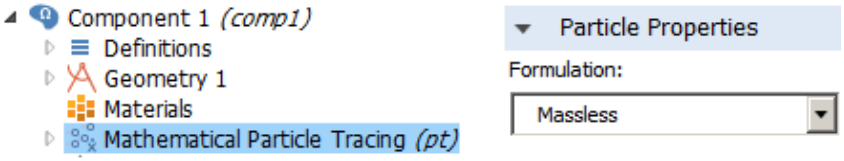
Global Definitions: RMB > Parameters

`Parameters` – **Name, Expression** = `a, 3`
– **Name, Expression** = `b, 3`
– **Name, Expression** = `c, 1`
– **Name, Expression** = `d, 1`

Виберіть безмасовий (`Massless`) варіант інтерфейсу.

Component 1 - Mathematical Particle Tracing (pt):

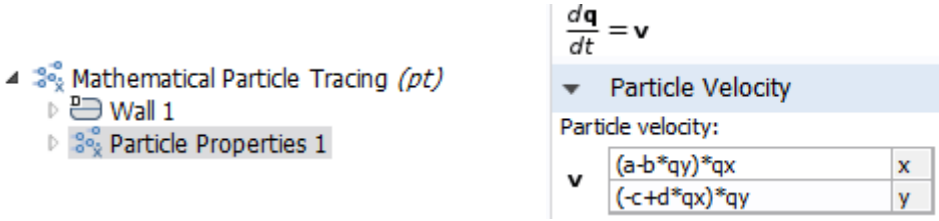
`Particle Properties` – **Formulation** = `Massless`



Введіть систему рівнянь. В ролі шуканих величин `x`, `y` виступатимуть `qx` і `qy`.

Mathematical Particle Tracing (pt) – Particle Properties 1:

`Particle Velocity` – **v <x>** = `(a-b*qy)*qx`
– **v <y>** = `(-c+d*qx)*qy`



На наступному кроці в гілці **Mathematical Particle Tracing (pt)** створіть вузол **Release from Grid 1**, в якому задайте координати чотирьох початкових точок.

Mathematical Particle Tracing (pt) : RMB > Release from Grid

Mathematical Particle Tracing (pt) – Release from Grid 1:

Initial Coordanates – **Grid type**= Specified combinations

Initial Coordanates – $\mathbf{q}_{x,0}$ = 2,1,1,3

– $\mathbf{q}_{y,0}$ = 1,2,3,2

Вибір Specified combinations означатиме, що точки матимуть координати $(2,1),(1,2),(1,3),(3,2)$, а не всі можливі комбінації введених чисел.

Залишилося ввести діапазон зміни часу і відносну похибку обчислень.

Step 1: Time Dependent:

Study Settings – **Times** = range(0,0.05,5)

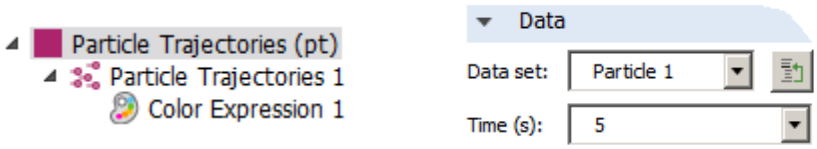
– **Relative tolerance** = True, 0.001

Виконайте обчислення.

Home -> Compute

В гілці **Results - Data Sets** з'явиться вузол **Particle 1**, який представлятиме джерело даних типу Particle. Також в гілці **Results** з'явиться графічна група типу 2D Plot Group з назвою **Particle Trajectories (pt)**. Зверніть увагу, що в її ПВ в списку **Data Set** обрано джерело даних **Particle 1**.

В графічній зоні вузла **Particle Trajectories (pt)** відобразилися тільки чотири точки. Вони розташовані на фазових траєкторіях, і їхнє положення визначається моментом часу, обраним в списку **Time** розділу **Data** ПВ вузла **Particle Trajectories (pt)**.



Налаштуємо цю графічну групу для відображення фазових траєкторій.

Particle Trajectories (pt):

Plot Settings – **Plot data set edges** =False

Particle Trajectories (pt) - Particle Trajectories 1:

Coloring and Style – **Line style** – **Type** =Tube

– **Line style** – **Tube radius expression** =0.03

– **Line style** – **Radius scale factor** =True,1

Coloring and Style – **Point style** – **Type** = None

Додатково зобразимо на графіку початкові точки фазових траєкторій. Для цього дублюємо вузол **Particle Trajectories 1** і його копію налаштуємо для відображення цих точок.

Particle Trajectories (pt) - Particle Trajectories 1: RMB > Duplicate

Particle Trajectories (pt) - Particle Trajectories 2 - Color Expression 1:

RMB > Delete

Particle Trajectories (pt) - Particle Trajectories 2:

Data –Data set = Particle 1

Data –Time = 0

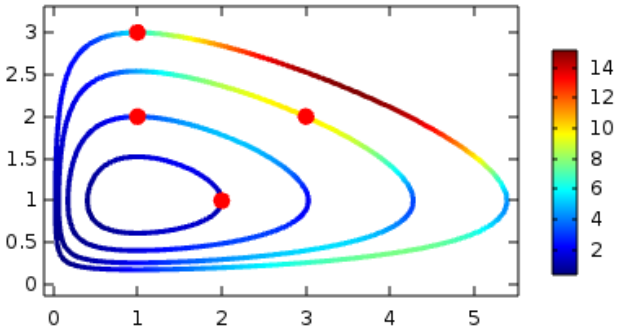
Coloring and Style –Line style –Type =None

Coloring and Style –Point style –Type = Point

– Point style –Point radius expression = 0.1

– Point style – Radius scale factor = True,1

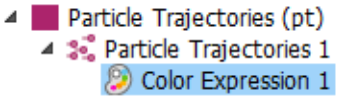
LMB > Plot

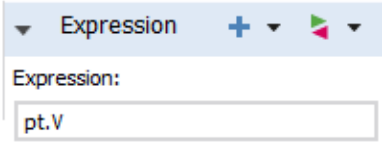


Момент часу, для якого побудовані фазові траєкторії, визначається значенням в полі **Time** вузла **Particle Trajectories (pt)**. У нас воно дорівнює 5 і ми спостерігаємо фазові траєкторії цілком. Червоні точки відповідають моменту часу t=0, тобто є початковими точками фазових кривих.

З вигляду траєкторій зрозуміло, що чисельність популяцій змінюється періодично.

Колірна палітра справа відображає величину (модуль) швидкості, з якою точка рухається по фазовій траєкторії. Це визначається налаштуваннями в ПВ вузла **Particle Trajectories 1 – Color Expression 1** в полі **Expression**.





Тут префікс pt є ім'ям моделі і задається в ПВ вузла **Mathematical Particle Tracing (pt)** в полі **Name**. А V позначає величину (модуль вектора) швидкості.

Щоб побудувати криві залежності розв'язків від часу, створіть в ДМ графічну групу **1D Plot Group 2** і додайте в неї два підвузли типу Particle, в яких зобразіть графіки величин qx і qy.

Results **-> Plot Group** – 1D Plot Group

1D Plot Group 2:

Data – Data set= Particle 1

RMB > Particle

1D Plot Group 2 – Particle 1:

y-Axis Data – Expression = qx

x-Axis Data – Expression = Time

Coloring and Style – Width = 3

Data Series Operation – Operation = None

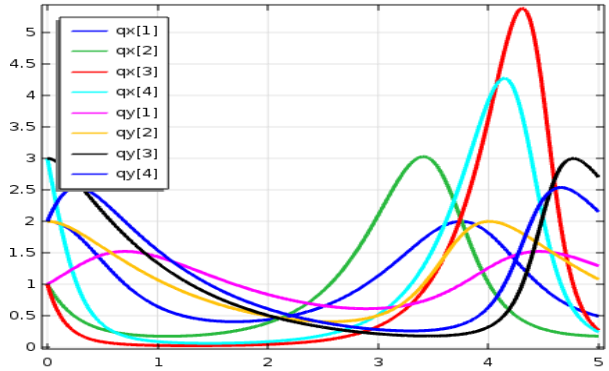
Останнє означає, що будуть будуватися окремі графіки горизонтальних координатних функцій $qx_i(t)$, $i = 1,2,3,4$.

1D Plot Group 2 – Particle 1: RMB > Duplicate

1D Plot Group 2 – Particle 2:

y-Axis Data – Expression = qy

LMB > Plot



Ми ще налаштували легенду цього графіка.

Протабулюйте значення швидкості (модуля вектора швидкості) в різні моменти часу.

Derived Values: RMB > Particle Evaluation

Derived Values - Particle Evaluation 1:

В полі **Expression** клацніть мишею по кнопці Replace Expression. Відкривається вікно, в якому можна вибрати різні вирази. Знайдіть вираз `pt.V-Particle velocity magnitude` і зробіть по ньому подвійне клацання миші. В полі **Expression** з'явиться вираз `pt.V`.

Expression – Expression = `pt.V`

Натисніть кнопку Evaluate вгорі ПВ вузла **Particle Evaluation 1** і отримайте таблицю швидкостей для кожної з чотирьох фазових кривих. Для зміни точності відображення чисел використайте кнопку Full Precision, розташовану над таблицею.

Time	Particle velocity magnitu	Particle 2	Particle 3	Particle 4
0	1	3	6	9.848857801796104
0.025	1.0345359625039168	2.7820186473668596	5.16795926971753	9.851051078393851
0.05	1.0870053053886823	2.5795852733063436	4.471120004338723	9.66171437307983
0.07500000000...	1.1547252314128496	2.3943473898775043	3.9035467051905304	9.306973458160206
0.1	1.2342104764192054	2.2271836514800483	3.454250692833082	8.820269168629988
0.125	1.3217166751830798	2.0783000847041584	3.1086846060538607	8.238269784835683
0.15000000000		[illegible]45535	2.8500947098779017	7.597241443016[illegible]

4.4.7. Рух планети навкруги сонця.

Розглянемо задачу руху планети навкруги Сонця під дією сили тяжіння.

Закон всесвітнього тяжіння Ньютона у векторній формі має вигляд $m\mathbf{a}=F\mathbf{e}$, де $\boldsymbol{a}$ – вектор прискорення, $\mathbf{e}$ – одиничний вектор, спрямований від планети до сонця, $F=\gamma\dfrac{mM}{R^2}$ – величина сили тяжіння, R – відстань між тілами, m – маса планети, M – маса сонця, γ – гравітаційна стала. Зв'яжемо початок координат з масивним тілом (сонцем). У будь-який момент часу вектор-функція $(x(t),y(t))$ представлятиме радіус-вектор планети. Сила, з якою сонце притягує планету, буде спрямована вздовж одиничного вектора

$$\boldsymbol{e}=\left(\frac{-x(t)}{\sqrt{x^2(t)+y^2(t)}},\frac{-y(t)}{\sqrt{x^2(t)+y^2(t)}}\right),$$

який має протилежний напрямок радіус-вектору планети. Тоді рівняння руху матимуть вид

$$m\begin{Bmatrix}x''(t)\\ y''(t)\end{Bmatrix}=-\gamma\frac{mM}{(x^2(t)+y^2(t))^{3/2}}\begin{Bmatrix}x(t)\\ y(t)\end{Bmatrix}.$$

Введемо позначення $\gamma M=k$. Тоді отримаємо наступну систему звичайних диференціальних рівнянь відносно невідомих функцій $x(t)$, $y(t)$ – координат рухомої планети:

$$\begin{cases}x''(t)=-\dfrac{k\,x(t)}{(x^2(t)+y^2(t))^{3/2}}\\[2ex] y''(t)=-\dfrac{k\,y(t)}{(x^2(t)+y^2(t))^{3/2}}\end{cases}. \qquad (1)$$

Для модельної задачі виберемо k=1 і задамо початкові умови $x(0)=1,\ x'(0)=0,\ y(0)=0,\ y'(0)=0.4$.

Вигляд системи відповідає `Newtonian` редакції `Mathematical Particle Tracing` інтерфейсу, тому використаємо його.

`Розв'язання.` Створіть двовимірну динамічну модель з `Mathematical Particle Tracing` інтерфейсом.

File -> New -> Model Wizard -> 2D -> Mathematics - Mathematical Particle Tracing (pt) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Розрахунковою областю задачі вважатимемо еліпс.

`Geometry` -> **`Primitives`** – Ellipse

Component 1 - Geometry 1 - Ellipse1:

Size and Shape – **a-semiaxes, b-semiaxes,** = 1,0.5

Position – **Base, x, y** = Center,0.5,0

LMB > Build All Objects

Перевірте, що в ПВ вузла **Mathematical Particle Tracing (pt)** обрана Newtonian редакція інтерфейсу. В розділі **Dependent Variable** ви можете подивитися імена шуканих функцій (в нашому прикладі в якості шуканих функцій виступають змінні qx і qy). Перевірте також, що в розділі **Domain Selection** в секції вибору стоїть одиниця, тобто обрана область з індикатором 1 (це еліпс).

У вузлі **Particle Properties 1** задайте одиничне значення маси.

Mathematical Particle Tracing (pt) – Particle Properties 1:

Particle Mass – $\mathbf{m_p}$ = 1

В гілку **Mathematical Particle Tracing (pt)** додайте вузол Release from Grid, в ПВ якого введіть початкове положення $q_{x,0}$, $q_{y,0}$ і початкову швидкість $\mathbf{v_0}$ планети.

Mathematical Particle Tracing (pt) : RMB > Release from Grid

Mathematical Particle Tracing (pt) – Release from Grid 1:

Initial Coordanates – $\mathbf{q_{x,0}}$, $\mathbf{q_{y,0}}$ = 1, 0

Initial velocity – $\mathbf{v_o}$ **<x>**= 0

– $\mathbf{v_o}$ **<y>**= 0.4

Додайте вузол Force. В його ПВ заповніть таблицю з виразами правих частин системи (1), а також в секцію вибору розділу **Domain Selection** додайте індикатор 1 - номер області, в якій буде діяти векторне силове поле ***F***. Для додавання номера області просто клацніть по еліпсу мишкою.

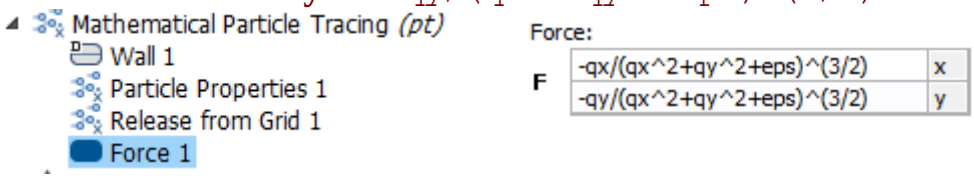
Mathematical Particle Tracing (pt) : RMB > Force

Mathematical Particle Tracing (pt) – Force 1:

Domain Selection – **Selection** = Manual, 1

Force – **Force: F <x>** = -qx/(qx^2+qy^2+eps)^(3/2)

Force – **Force: F <y>** = -qy/(qx^2+qy^2+eps)^(3/2)



Зверніть увагу, що в знаменник виразів ми додали сталу eps – машинне епсілон $\approx 2.2204 \cdot 10^{-16}$. Це потрібно, щоб запобігти обертанню знаменника в нуль.

Задайте діапазон зміни параметра часу t і відносну похибку обчислень.

Step 1: Time Dependent:

Study Settings – **Times** = range(0,0.01,20)

– **Relative tolerance** = True, 1E-5

Виконайте обчислення.

Home -> Compute

В гілці **Results – Data Sets** буде створено два джерела даних **Study 1/Solution 1** і **Particle 1**, а також двовимірна графічна група **Particle Trajectories (pt)** з вузлом **Particle Trajectories 1**. Перевірте, що в ПВ вузла **Particle**

Trajectories (pt) в розділі **Data** в полі **Time** обраний максимальний момент часу (в нашому прикладі 20). Також зніміть перемикач **Plot data set edges**, щоб в графічному вікні не малювався контур розрахункової області (еліпса).

Particle Trajectories (pt):

Plot Settings – **Plot data set edges** =False

Щоб побачити траєкторію руху планети, відкоригуйте налаштування вузла **Particle Trajectories 1**.

Particle Trajectories (pt) - Particle Trajectories 1:

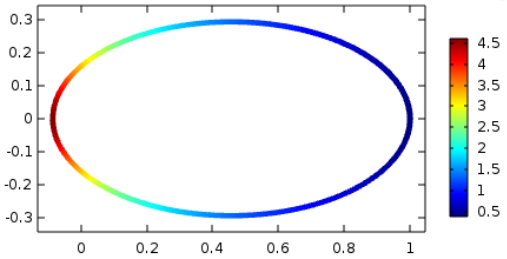
Coloring and Style – **Line style** – **Type** =Tube

– **Line style** – **Tube radius expression** =0.005

– **Line style** – **Radius scale factor** =True,1

Coloring and Style – **Point style** – **Type** = None

LMB > Plot



Видима ділянка траєкторії відповідає інтервалу часу від нуля до значення, зазначеного в полі **Time** вузла **Particle Trajectories (pt)**. Колірна палітра справа відображає величину (модуль) швидкості, з якою тіло рухається по кривій. Це визначається налаштуваннями в ПВ вузла **Particle Trajectories 1 - Color Expression 1** в полі **Expression**. В нашому прикладі це вираз pt.V, який позначає величину швидкості.

Побудуйте анімацію руху планети вздовж траєкторії.

Results –> Export - Animation –> Player

Export – Animation 1:

Scene – **Subject** = Particle Trajectories (pt)

Animation **Aditing** – **Time selection** = Manual

– **Times(s)(1-2001)** = range(1,10,260)

Остання опція задає діапазон номерів моментів часу (а не самі моменти).

Frames – **Frame selection** = All

Playing – **Repeat** = True

LMB > Generate Frame.

Почнеться процес генерування кадрів. По його завершенні натисніть кнопку Play ▶ на панелі керування анімацією. Вона розташована вгорі лівого кута графічного вікна. Цим ви запустите анімацію.

Зауваження. Спробуйте розв'язати задачу при більшому часовому кроці. Наприклад, у вузлі **Step 1: Time Dependent** в полі **Times** задайте діапазон range(0,0.1,20). Ви отримаєте значно гірший рисунок траєкторії.

4.4.8. Математичний більярд в еліпсі.

Є плоский горизонтальний більярдний стіл довільної форми без луз. По ньому без тертя рухається маленька кулька, пружно відбиваючись від бортів. Потрібно визначити траєкторію цієї кулі/точки. Механічна система – рухома точкова куля в області Q – називається математичним більярдом. Траєкторія точки в математичному більярді визначається її початковим положенням і вектором швидкості. В п. 2.3.8 ми розв'язували цю задачу для прямокутної області. Інтерфейс Mathematical Particle Tracing дозволяє моделювати рух точки в математичному більярді довільної форми.

Розглянемо рух точки в еліптичному математичному більярді.

Розв'язання. Створіть безрозмірну динамічну 2D модель задачі з Mathematical Particle Tracing інтерфейсом.

File -> New -> Model Wizard -> 2D -> Mathematics - Mathematical Particle Tracing (pt) -> Time Dependent

Untitled.mph (root): Unit System – <> = None

Розрахунковою областю задачі вважатимемо еліпс.

Geometry -> **Primitives** – Ellipse

Component 1 - Geometry 1 - Ellipse1:

Size and Shape – **a-semiaxes, b-semiaxes,** = 2,1

Position – **Base, x, y** = Center,0,0

LMB > Build All Objects

Перевірте, що в ПВ вузла **Mathematical Particle Tracing (pt)** обрана Newtonian редакція інтерфейсу.

У вузлі **Particle Properties 1** задайте одиничне значення маси.

Mathematical Particle Tracing (pt) – Particle Properties 1:

Particle Mass – $\mathbf{m_p}$ = 1

Вузол типу Wall застосовується для визначення поведінки часток в момент контакту з межею розрахункової області (відбиття, приставання, застигання і т.д.). В нашому випадку потрібно обрати відбиття (bounce).

Mathematical Particle Tracing (pt) – Wall 1:

Wall condition – **Wall condition:** = Bounce

В гілку **Mathematical Particle Tracing (pt)** додайте вузол Release from Grid, в ПВ якого введіть початкове положення і початкову швидкість точки.

Mathematical Particle Tracing (pt) : RMB > Release from Grid

Mathematical Particle Tracing (pt) – Release from Grid 1:

Initial Coordanates – $\mathbf{q_{x,0}}$, $\mathbf{q_{y,0}}$ = 0.2, 0.3

Initial velocity – $\mathbf{v_o}$ **<x>**= 1

– $\mathbf{v_o}$ **<y>**= sqrt(2)

Додайте вузол Force і в його ПВ задайте нульову силу.

Mathematical Particle Tracing (pt) : RMB > Force

Mathematical Particle Tracing (pt) – Force 1:

Domain Selection – **Selection** = Manual, 1

Force – **Force: F <x>** = 0

Force – **Force: F <y>** = 0

Задайте діапазон зміни параметра часу t.

Step 1: Time Dependent:

Study Settings – **Times** = range(0,0.05,10)

Виконайте обчислення.

Home -> Compute

В гілці **Results – Data Sets** буде створено два джерела даних **Study 1/Solution 1** і **Particle 1**, а також двовимірна графічна група **Particle Trajectories (pt)** з вузлом **Particle Trajectories 1**. Відкоригуйте їх налаштування.

Particle Trajectories (pt) - Particle Trajectories 1 - Color Expression 1:

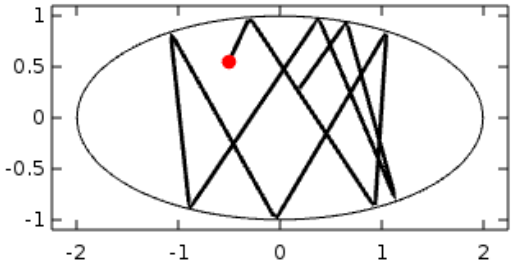
RMB > Delete

Particle Trajectories (pt) - Particle Trajectories 1:

Coloring and Style – **Line style** – **Type** =Tube
– **Line style** – **Tube radius expression** =0.02
– **Line style** – **Radius scale factor** =True,1
– **Color** =Black

Coloring and Style – **Point style** – **Type** = Point
– **Point style** – **Point radius expression** = 0.1
– **Point style** – **Radius scale factor** = True,1
– **Point style** – **Color** = Red

LMB > Plot



Створена крива являє відрізок траєкторії, яка починається в момент часу t=0 в точці $(0.2, 0.3)$, і закінчується в момент t, заданий у полі **Time** батьківської графічної групи (t=10).

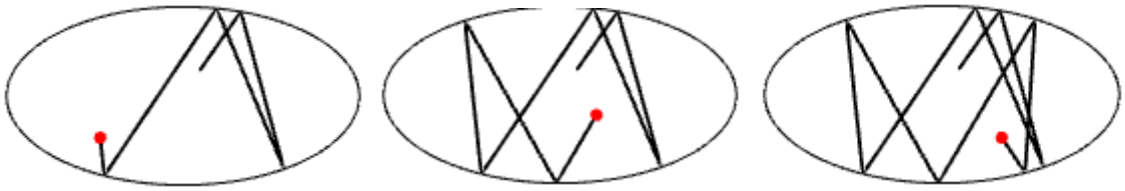
Побудуйте анімацію.

Results –> Export - Animation –> Player

Animation 1:

Frames – **Frame selection** = All

Згенеруйте кадри анімації, натиснувши на кнопку Generate Frame, і запустіть її кнопкою Play. На наступному рисунку показані кадри, побудовані в моменти часу t=4.2,6.7,8.7.



4.4.9. Поперечні коливання струни з двома вантажами.

Маємо 2 вантажу маси *M*, які закріплені на горизонтальній струні довжиною $L = 3a$ в точках *x*=*a* і *x*=2*a*. Відрізки струни між вантажами однакові, невагомі і підкоряються закону Гука. Натяг струни дорівнює *T*. Позначимо вертикальне відхилення вантажів від положення рівноваги на осі струни через $y_1(t)$ і $y_2(t)$. Обмежимося розгляданням малих коливань. Тоді рівняння руху вантажів матимуть вид

$$M\frac{d^2y_1}{dt^2} = T\left(\frac{y_2 - y_1}{a}\right) - T\left(\frac{y_1 - y_0}{a}\right),$$

$$M\frac{d^2y_2}{dt^2} = T\left(\frac{y_3 - y_2}{a}\right) - T\left(\frac{y_2 - y_1}{a}\right)$$

де y_0, y_3- зміщення лівого і правого кінців закріпленої струни, тобто нулі. Проаналізуємо коливання механічної системи без початкової швидкості, поклавши *a*=1, *M*=1, *T*=1.Тоді отримаємо систему

$$\begin{cases} \dfrac{d^2y_1}{dt^2} = y_2 - 2y_1 \\ \dfrac{d^2y_2}{dt^2} = -2y_2 + y_1 \end{cases}. \qquad (1)$$

Користувачеві повинно бути зрозуміло, що для дослідження цієї задачі в COMSOL потрібно розв'язати систему ЗДР, тобто підходить `0D Global ODEs and DAEs` інтерфейс. Але ми бажаємо побудувати графік струни з точковими масами. Тоді нам потрібна просторова модель і для цього може підійти `Domain ODEs and DAEs` інтерфейс.

Розв'язання. Створіть одновимірну динамічну модель з Domain ODEs and DAEs інтерфейсом.

File->New->Model Wizard->1D-> Mathematics - ODEs and DAEs Interfaces - Domain ODEs and DAEs (dode) -> Time Dependent

Untitled.mph (root): `Unit System` – <> = None

Геометрія моделі складатиметься з відрізка, який буде зображати невагому струну, і двох точок, що будуть зображати зосереджені маси.

`Geometry`**->Pimitives->** Interval

Geometry – Interval 1 (i1):

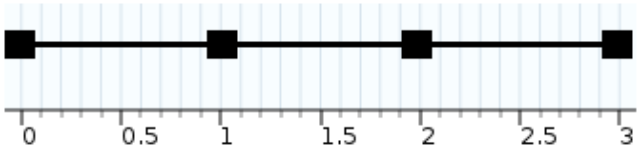
`Interval` – Right endpoint = 3

`Geometry`->**Primitives->**Point

`Point` – x: = 1,2 (*абсциси двох внутрішніх точок*)

LMB > Build All Objects

В графічному вікні буде побудовано зображення відрізка з точками розбиття.



На відрізку побудуйте СкЕ сітку.

Mesh 1:

Mesh Settings – Element size = Finer

LMB > Build All

Додайте в модель Global Equations вузол і в його ПВ заповніть таблицю рівнянь.

Physics -> **Global ->** Global Equations

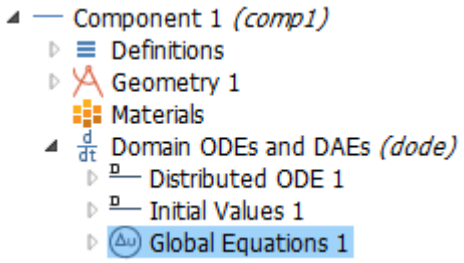
Domain ODEs and DAEs(dode) – Global Equations 1:

Global Equations – Name, f(u, ut, utt, t) = y1,y1tt-y2+2*y1

– Initial value(u_0), Initial value(u_t0) =1,0

Global Equations – Name, f(u, ut, utt, t) = y2,y2tt-y1+2*y2

– Initial value(u_0), Initial value(u_t0) =1,0



Global Equations

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t)	Initial value (u_0)	Initial value (u_t0)
y1	y1tt-y2+2*y1	1	0
y2	y2tt-y1+2*y2	1	0

Таким чином, тут визначено дві невідомі y1 і y2, які будуть розшукуватися як розв'зок системи (1), і зображати зміщення точкових мас. Зверніть увагу на те, що початкові значення цих змінних (і їх похідні по часу) визначено в таблиці. Маючи y1 і y2, можна записати рівняння струни на різних частинах геометрії. На них струна буде зображатися прямолінійними відрізками, положення кінців яких відомо. Щоб записати таке рівняння, спочатку на кожному з інтервалів створіть змінну f, задану власним виразом.

Definitions -> **Variables** – Local Variables

Definitions - Variables 1:

Geometric Entity Selection – Geometric entity level = Domain

– Selection - <> = 1 (*лівий відрізок*)

Variables (таблиця) – Name = f

– Expression = y1*x

Створіть ще один вузол змінних.

Definitions -> **Variables** – Local Variables

Definitions - Variables 2:

Geometric Entity Selection – Geometric entity level = Domain

– Selection - <> = 2 (*середній відрізок*)

Variables (таблиця) – Name = f

– Expression = y1+(y2-y1)*(x-1)

Створіть третій вузол змінних.

Definitions -> **Variables** – Local Variables

Definitions - Variables 3:

Geometric Entity Selection – Geometric entity level = Domain

– Selection - <> = 3 (*правий відрізок*)

Variables (таблиця) – **Name** = f

– **Expression** = y2*(3-x)

Ми створили три різні вузли Variables, в кожному з яких задали змінну з однаковим ім'ям f. Для кожного з вузлів визначили різні ділянки дії (різні відрізки) і для f задали різні формули. Коли в моделі ви будете звертатися до змінної f, система буде визначати ділянку і автоматично обирати потрібний вираз для її обчислення. Таким чином, змінна f визначена на трьох різних інтервалах і, в результаті, задається формулою

$$f = \begin{cases} y_1 \cdot x & , 0 \le x \le 1 \\ y_1 + (y_2 - y_1)(x-1) & , 1 \le x \le 2 \\ y_2 \cdot (3-x) & , 2 \le x \le 3 \end{cases} .$$

Налаштуйте параметри розподіленої задачі.

Domain ODEs and DAEs (dode) - Distributed ODE 1:

Source Term – **f** = u-f

Damping or Mass coefficient – $\mathbf{d_a}$ = 0

Mass coefficient – $\mathbf{e_a}$ = 0

Отже шукана змінна u буде визначатися рівнянням $u - f(x) = 0$ і обчислюватися за різними виразами на різних ділянках геометрії.

Налаштуйте часовий інтервал.

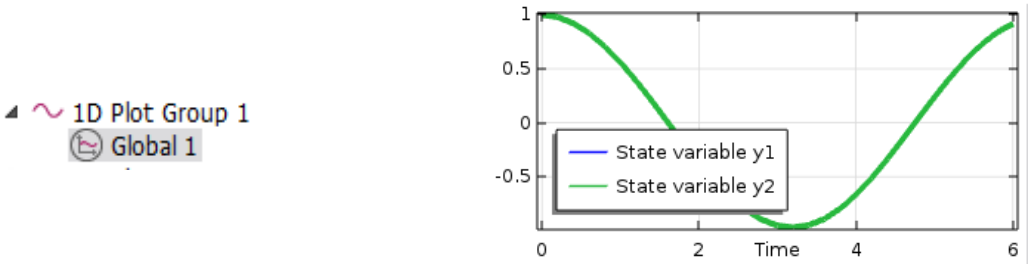
Study 1 - Step 1: Time Dependent:

Study Settings – **Times** = range(0,0.2,6)

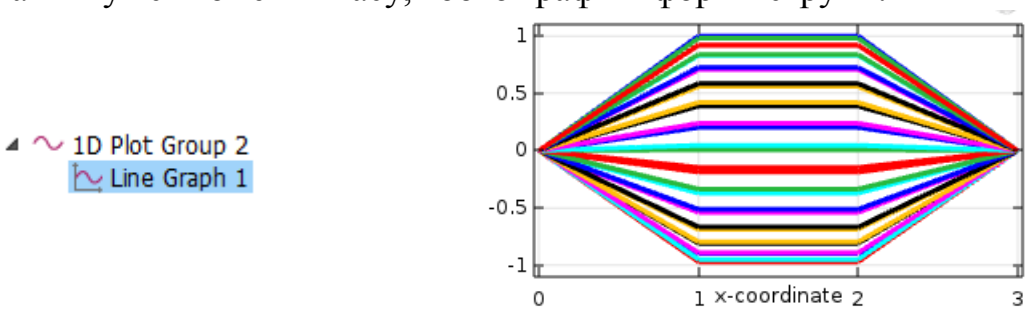
Виконайте обчислення.

Home->Compute

Буде створено дві графічні групи. В групі **1D Plot Group 1** побудований графіки функцій $y_1(t)$ і $y_2(t)$ в залежності від часу (криві співпадають).



В групі **1D Plot Group 2** побудовані графіки функції u в залежності від координати x у всі моменти часу, тобто графіки форми струни.



Бажано на графіку зобразити маси. Для цього в групу **1D Plot Group 2** додайте ще два лінійних графіки, які будуть зображати вирази $y_1(t)$ і $y_2(t)$ в залежності від фіксованої просторової координати.

1D Plot Group 2:

Data – **Time selection** = Manual

Data – **Times (1-41)** = 1

Axis – **Manual axis limits** = True

– **y minimum, y maximum** = -1.1,1.1

1D Plot Group 2 -> **Add Plot** – Line Graph

1D Plot Group 2 - Line Graph 2:

Selection – **Selection -** < > = 2 (*індикатор середнього відрізка*)

y-Axis Data – **Expression** = y1

x-Axis Data – **Parameter** = Expression

– **Expression** = 1 (*абсциса лівої маси*)

Coloring and Style – **Width** = 7 (*розмір точки*)

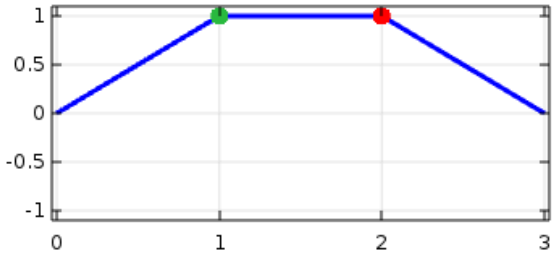
RMB > Duplicate

1D Plot Group 2 - Line Graph 3:

y-Axis Data – **Expression** = y2

x-Axis Data – **Expression** = 2 (*абсциса правої маси*)

Налаштуйте стиль графіка на свій розсуд, і виконайте команду LMB > Plot. Буде побудований профіль струни з точковими масами в початковий момент часу.



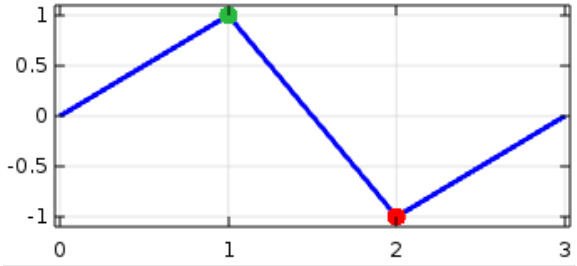
Анімацію руху струни з масами побудуйте самостійно.

Отримане синхронне коливання обох мас зображає першу моду коливань системи. Для побудови другої моди, яка зображає рух мас в «протифазі», поміняйте в ПВ вузла **Domain ODEs and DAEs(dode) – Global Equations 1** в таблиці рівнняь початкове значення для шуканої величини y2 .

Domain ODEs and DAEs(dode) – Global Equations 1:

Global Equations – **Initial value(u_o),** = -1 (*в* 2-*му рядку таблиці*)

Переобчисліть модель і у вузлі **1D Plot Group 2** отримайте наступний графік.



На цьому рисунку зображено початковий стан системи. Обираючи в полі **Data – Times (1-41)** вузла **1D Plot Group 2** інші номери моментів часу, ви будете отримувати інші профілі струни.

Для створення довільних коливань (суперпозиції першої і другої моди) достатньо задати довільні початкові зміщення. Спробуйте, наприклад, умови $y_1(0)=1,\ y_2(0)=0$.

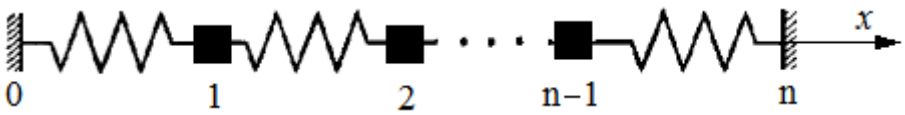
Зауваження 1. Для побудови точкових мас здається можна було б використати вузли типу `Point Graph`, але вони не підходять для побудови анімації, бо в динамічних задачах будують траєкторії руху точок (тобто будуватимуться вертикальні відрізки вздовж яких рухаються маси).

Зауваження 2. В нашому прикладі `Domain ODEs and DAEs (dode)` інтерфейс відігравав допоміжну роль. Одновимірна геометрія створювалась у вигляді інтервалу в цьому інтерфейсі, а польова змінна `u`, створена разом з інтерфейсом, репрезентувала зміщення точок струни.

Вправа. Розв'яжіть ту ж задачу, створивши динамічну 1D модель з `Global ODEs and DAEs` інтерфейсом.

4.4.10 Одновимірна модель кристала.

Наведемо найпростішу одновимірну модель кристала – ланцюжок однакових тягарців маси *m*, розташованих на однаковій відстані *a* один від одного, і пов'язаних один з одним однаковими пружинками жорсткістю *k*. При відхиленні від положення рівноваги пружини намагаються повернути тягарці в стан рівноваги.



Позначимо через x_i відхилення уздовж осі X грузика *i* від положення рівноваги. У випадку малих (лінійних) коливань, відповідно до закону Гука, рівняння руху *i*-го грузика матиме вигляд

$$m\frac{d^2x_i}{dt^2}=k\left(x_{i+1}-x_i\right)-k\left(x_i-x_{i-1}\right)\quad (i=1,\ 2,\ \ldots,n-1), \tag{1}$$

де в силу умови закріплення кінців, слід вважати $x_0(t)=0,\ x_n(t)=0$.

Фермі з колегами С. Уламом і Дж. Пастой припустили, що сила $F\left(x_{i+1},x_i\right)$, з якої взаємодіють `(i+1)`-й та `i`-й атоми має невелику нелінійну добавку і, наприклад, може мати вигляд $F\left(x_{i+1},x_i\right)=k\cdot\Delta l+\alpha\cdot\Delta l^{\,2}$, де $\Delta l=x_{i+1}-x_i$ і α - мало. Це призводить до наступної системи ЗДР

$$m\frac{d^2x_i}{dt^2}=k\cdot\left(x_{i+1}-2x_i+x_{i-1}\right)+\alpha\cdot\left(\left(x_{i+1}-x_i\right)^2-\left(x_i-x_{i-1}\right)^2\right)\quad (i=1,2,\ldots,n-1), \tag{2}$$

де $x_0(t)=0,\ x_n(t)=0$. При розв'язанні системи (2) Фермі з колегами стежили не за окремими частинками, а за поведінкою мод коливань всієї механичної системи, які при $\alpha=0$ збігаються з синусоїдами.

Спочатку розглянемо задачу лінійних коливань системи n=32 тягарців, у якої в початковий момент часу збуджено тільки дві перші моди. Потім – лінійні коливання з багатьма модами (початковий профіль у формі ділянки параболи). В третьому прикладі розв'яжемо задачу Фермі – Пасти – Улама.

Приклад 1. Дослідити лінійні подовжні коливання механічної системи n однакових тягарців маси $m=1$, розташованих вздовж прямої лінії на однаковій відстані $a=1$ один від одного, і з'єднаних однаковими пружинками жорсткістю $k=1$. Крайні тягарці фіксовані. Початкова швидкість всіх грузиків дорівнює нулю. Початкові зміщення $x_i(0)$ тягарців відносно положення рівноваги $i\cdot a$ задається сумою двох синусоїд $x_i(0)=0.8\cdot(\sin(\pi i/n)+\sin(2\pi i/n))$.

При обраних параметрах з (1) отримуємо наступну задачу Коші

$$\frac{d^2x_i}{dt^2}=(x_{i+1}-x_i)+(x_{i-1}-x_i)\quad (i=1,\ 2,\ \dots,n-1),\quad x_0(t)=0,\ x_n(t)=0,\qquad (3)$$

$$x_i(0)=0.8\cdot(\sin(\pi i/n)+\sin(2\pi i/n)),\ \ \frac{dx_i}{dt}(0)=0.$$

Її можна розв'язувати з застосуванням Newtonian редакції Mathematical Particle Tracing інтерфейсу. В його рівняннях використовується не зсув частинок x_j відносно положення рівноваги, а координати (абсциси) q_j частинок, які пов'язані зі зсувом співвідношенням $q_j=j\cdot a+x_j$, або $x_j=q_j-j\cdot a\quad (j=0,1,2\dots,n)$. Тоді права частина рівняння (3) може бути записана у вигляді

$$\begin{gathered}(x_{i+1}-x_i)+(x_{i-1}-x_i)=\\ =((q_{i+1}-(i+1)a)-(q_i-ia))+((q_{i-1}-(i-1)a)-(q_i-ia))=\\ =(q_{i+1}-ia-a-q_i+ia)+(q_{i-1}-ia+a-q_i+ia)=\\ =(q_{i+1}-a-q_i)+(q_{i-1}+a-q_i)=(q_{i+1}-q_i)+(q_{i-1}-q_i)\end{gathered}$$

Враховуючи, що $\frac{d}{dt}x_j(t)=\frac{d}{dt}q_j(t),\ \ \frac{d^2}{dt^2}x_j(t)=\frac{d^2}{dt^2}q_j(t)$, задача (3) набуває вигляду

$$\frac{d^2q_i}{dt^2}=(q_{i+1}-q_i)+(q_{i-1}-q_i)\quad (i=1,\ 2,\ \dots,n-1),\quad q_0(t)=0,q_n(t)=n\cdot a,\qquad (4)$$

$$q_i(0)=i\cdot a+0.8\cdot(\sin(\pi i/n)+\sin(2\pi i/n)),\ \ \frac{dq_i}{dt}(0)=0.$$

Звичайно, що в Mathematical Particle Tracing інтерфейсі при постановці задачі замість q_i використовуватиметься позначення qx.

Розв'язання. Створіть безрозмірну динамічну 2D модель задачі з Mathematical Particle Tracing інтерфейсом.

File -> New -> Model Wizard -> 2D -> Mathematics -
Mathematical Particle Tracing (pt) -> Time Dependent

Untitled.mph (root): `Unit System` – <> = None

В гілці **Global Definitions** створіть вузол **Parameters** і в таблиці задайте параметри `a` і `n`.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `a,1`
– **Name, Expression** = `n,32`

Розрахунковою областю вважатимемо прямокутник.

`Geometry` -> **`Primitives`** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

`Size` – **Width, Height** = `n*a,6*a`

`Position` – **Base, x, y** = `Corner,0,-3*a`

LMB > Build All Objects

Перевірте, що в ПВ вузла **Mathematical Particle Tracing (pt)** обрана `Newtonian` редакція інтерфейсу (значення за замовчуванням).

У вузлі **Particle Properties 1** задайте одиничне значення маси (всіх мас).

Mathematical Particle Tracing (pt) – Particle Properties 1:

`Particle Mass` – $\mathbf{m_p}$ = `1`

В гілку **Mathematical Particle Tracing (pt)** додайте вузол `Release from Grid`, в ПВ якого введіть початкові положення мас (включно з точками кріплення).

Mathematical Particle Tracing (pt) : RMB > Release from Grid

Mathematical Particle Tracing (pt) – Release from Grid 1:

`Initial Coordanates` – **Grid type**= `Specified combinations`

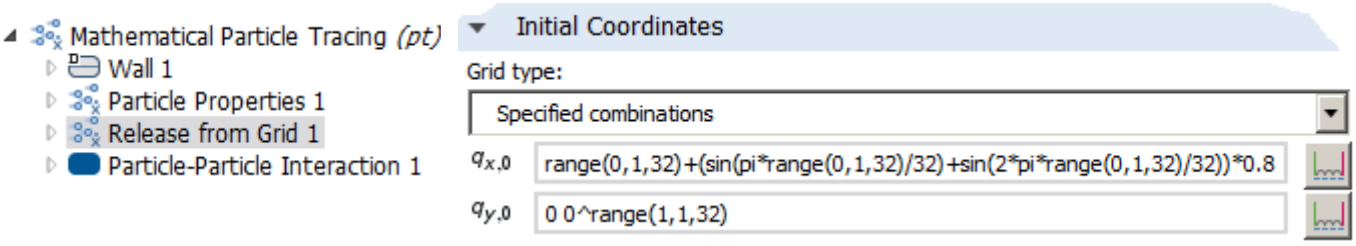
Опція `Specified combinations` визначає, що в наступних двох полях $\mathbf{q_{x,0}}$ і $\mathbf{q_{y,0}}$ будуть задаватися початкові абсциси і ординати вантажів, а не всілякі комбінації даних з цих полів.

`Initial Coordanates` – $\mathbf{q_{x,0}}$ = `range(0,1,32)+`
`(sin(pi*range(0,1,32)/32)+`
`sin(2*pi*range(0,1,32)/32))*0.8`

`Initial Coordanates` – $\mathbf{q_{y,0}}$ = `0 0^range(1,1,32)`

Оскільки в `Mathematical Particle Tracing` інтерфейсі `qx` представляє положення точок, а не їх зміщення, то у виразі $\mathbf{q_{x,o}}$ ми використали доданок `range(0,1,32)`, який визначає положення рівноваги важків. А початковий зсув точок механічної системи визначено сумою двох синусоїд. У версії COMSOL 5.1 у вузлі `Release from Grid` є недолік – не можна застосовувати параметри. Тому ми вимушено використали скаляр `32` замість параметра `n`. В подальших версіях програми ця недоробка усунута.

Зверніть також увагу на діапазон $\mathbf{q_{y,o}}$, в якому перший нуль (з тридцяти трьох) введено окремо. Це викликано тим, що система обчислює `0^0=1` (в полі $\mathbf{q_{y,o}}$ можна було б ввести `0^range(1,1,33)`).



Початкова швидкість точок покладається нульовою.

Initial velocity – $\mathbf{v_o}$ **<x>**= 0

– $\mathbf{v_o}$ **<y>**= 0

В Newtonian редакції Mathematical Particle Tracing інтерфейсу можуть існувати два види сил, які діють на частинки – це зовнішні силові поля (вони створюються у підвузлі Force) і сили взаємодії між частинками (вони створюються у вузлі Particle-Particle Interaction). В нашому прикладі зовнішнє силове поле відсутнє, але діють пружні сили між вантажами.

В гілку **Mathematical Particle Tracing (pt)** додайте вузол Particle-Particle Interaction і в його ПВ введіть праву частину системи (4).

Mathematical Particle Tracing (pt) : RMB > Particle – Particle Interaction

Mathematical Particle Tracing (pt) – Particle - Particle Interaction 1:

Domain Selection – **Selection** = Manual,1

Force – **Interaction force** = User defined

Force – **Force: F <x>** = if(dest(pt.pidx)==1
|| dest(pt.pidx)==n+1,0,(qx-dest(qx))*
if(pt.pidx==dest(pt.pidx)-1 ||
pt.pidx==dest(pt.pidx)+1,1,0))

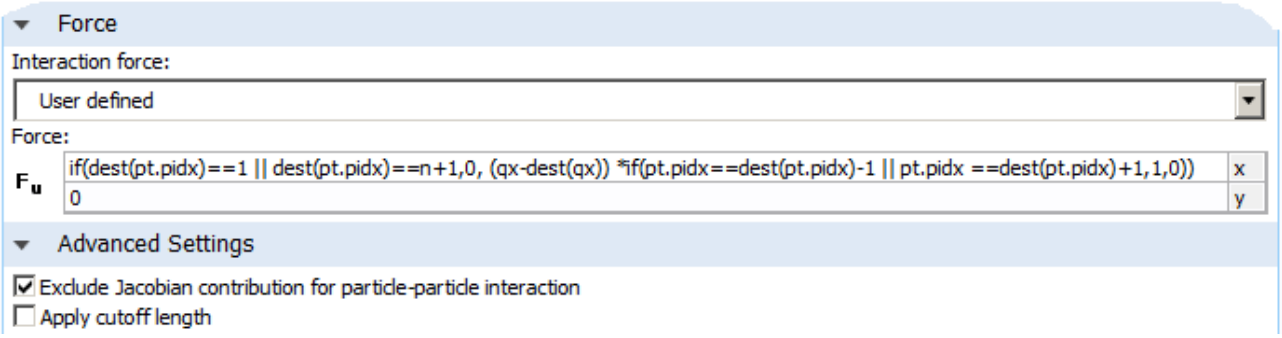
Пояснимо останній вираз. Координати і індекси точок механічної системи знаходяться в змінних qx, qy, pt.pidx. При записі виразу зовнішньої сили, яка діє на будь-яку частинку, координати і індекси поточної частинки позначають dest(qx), dest(qy), dest(pt.pidx), а координати і індекси будь-якої іншої – qx,qy, pt.pidx. При чому індекси pt.pidx починають відлік від 1.

Сумарна сила (пружна сила плюс реакція закріплення), яка діє на кінцеві (затиснені) вузли з індексами 1 і 33, дорівнює нулю. Тому щоб формула не поширювалася на крайні частинки, а загальна діюча на них сила дорівнювала нулю, було використано зовнішній умовний оператор if(dest(pt.pidx)==1 || dest(pt.pidx)==n+1,0,...), а в трикрапках записано вираз сили, яка діє на внутрішні тягарці.

Припустимо, що розглядається внутрішня частинка з позначкою dest(pt.pidx). Вираз qx-dest(qx) повинен обчислюватися лише для пружин, сусідніх до цієї частинки. Множник з внутрішнім умовним оператором if(pt.pidx==dest(pt.pidx)-1 || pt.pidx==dest(pt.pidx)+1,1,0) дорівнює 0 для всіх важків системи, окрім тих, індекси pt.pidx яких відрізняються на одиницю від індексу dest(pt.pidx) оброблюваного тягарця. Тому він визначає, що пружні сили діють лише між сусідніми важками.

Вертикальні координати тягарців в початковий момент часу (як і у всі інші) дорівнюють нулю.

Force – **Force: F <y>** = 0



Задайте діапазон зміни часу `t`.

Step 1: Time Dependent:

Study Settings – **Times** = `range(0,1,2*n)`

Зауваження. В поточному прикладі `2*n` – це період коливань розглядуваної механічної системи.

Виконайте обчислення.

Home -> Compute

В гілці **Results – Data Sets** буде створено два джерела даних **Study 1/Solution 1** і **Particle 1**. Також буде створена двовимірна графічна група **Particle Trajectories (pt)** з вузлом **Particle Trajectories 1**. Відкоригуйте їх налаштування.

Particle Trajectories (pt):

Data – **Time** = 64

Plot Settings – **Plot data set edges** = `False`

Particle Trajectories 1 - Color Expression 1: RMB > Delete

LMB > Plot (*наступний рисунок ліворуч*)

На рисунку (наступний ліворуч) зміщення точок відкладено в горизонтальному напрямку. Це правильно, але не наглядно. Для наочності зміщення вантажів краще відкласти у вертикальному напрямку. Для цього додайте до графіка вузол деформацій, за допомогою якого зміщення точок в горизонтальному напрямку зробіть нульовими, а вертикальні, для наочності, збільшіть в 5 разів.

Particle Trajectories (pt) - Particle Trajectories 1: RMB > Deformation

Particle Trajectories 1 - Deformation 1:

Expression – **x component** = `-qx+(pt.pidx-1)`

Тут величина `pt.pidx-1` представляє абсциси вантажів в стані рівноваги (при a =1).

Expression – **y component** = `(qx-(pt.pidx-1))*5`

В полі **y component** відкладено горизонтальні зміщення тягарців. Для цього введено не їх координати `qx`, а горизонтальний зсув відносно положення рівноваги, який, вочевидь, дорівнює `qx-(pt.pidx-1)`. Щоб абсциси тягарців співпадали з їх рівноважними положеннями,

масштаб по горизонталі повинен дорівнювати одиниці. Для збільшення вертикального масштабу в полі **y component** ми поставили множник 5.

Scale – **scale factor** = `True, 1`

Для порівняння можна показати положення точок механічної системи в стані рівноваги. Для цього додайте до графічної групи `Scatter Surface` вузол.

`Particle Trajectories(pt)` -> Add Plot - More Plots -> Scatter Surface

Particle Trajectories (pt) - Scatter Surface 1 :

Data– **Data set** = `Particle 1`

Expression – **x component, y component** = `x,y`

Evaluation Points–**x grid points** – **Entry method**=`Coordinates`

–**x grid points** – **Coordinates** = `range(0,1,n)`

–**y grid points** – **Entry method** = `Coordinates`

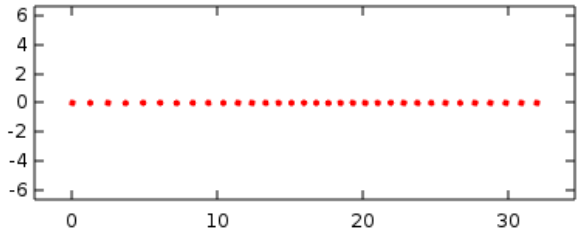
–**y grid points** – **Coordinates** = `0^range(1,1,n+1)`

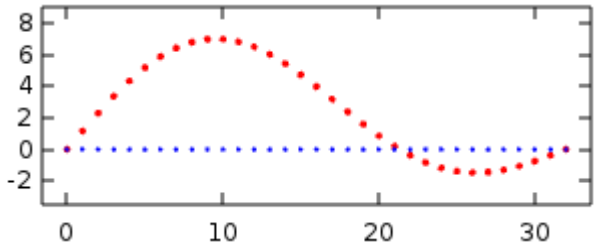
Radius – **Expression** =`0.15`

– **Radius scale factor** =`1`

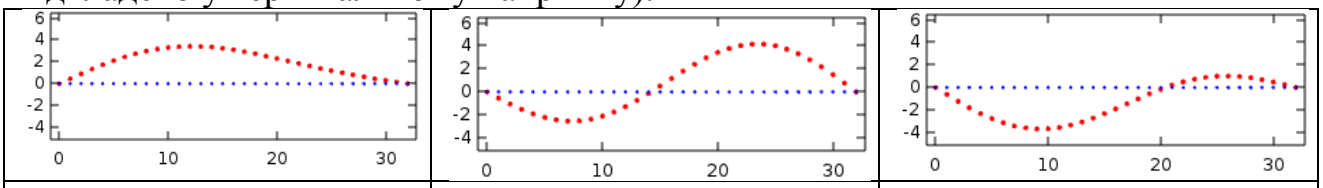
Coloring and Style – **Coloring, Color** = `Uniform, Blue`

LMB > Plot (*наступний рисунок праворуч*)





Анімацію руху побудуйте самостійно. На наступному рисунку наведено графіки форми механічної системи в деякі моменти часу (зміщення тягарців відкладено у вертикальному напрямку).



t=7	t=13	t=21

Збережіть файл, наприклад, з ім'ям `CrystalModel_1D.mph`, але не закривайте його.

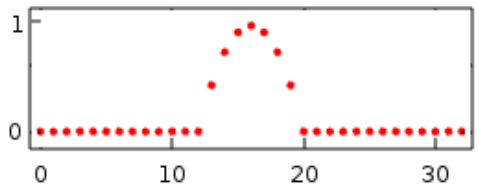
Вправа. Задайте початковий профіль розглядуваної механічної системи у формі першої моди $x_i(0) = \sin(\pi i / n)$. Дослідіть поведінку системи важків у часі приблизно до моментів `t=1000`. Звичайно, що відповідь потрібно буде трохи почекати.

Вказівка. При дослідженні моделі на великому інтервалі часу потрібно зменшити абсолютну і відносну похибки обчислень, наприклад, до значень 10^{-8}, інакше накопичувана похибка суттєво спотворить розв'язок.

Приклад 2. Дослідити попередню задачу при початковому зміщенні тягарців, яке визначається функцією

$$x_i(0)=\begin{cases}(i-12)\cdot(20-i)/16, & 12<i<20\\ i, & i\le 12 \text{ або } i\ge 20\end{cases}.$$

Параболічний пагорб всередині.



Розв'язання. Відкрийте файл `CrystalModel_1D.mph` попереднього прикладу і, для внесення змін, збережіть з іншим ім'ям.

Для розв'язання задачі потрібно змінити початкові координати точок механічної системи у вузлі **Release from Grid 1**. В цьому вузлі COMSOL Multiphysics 5.1 є недоробки, тому найпростішим способом введення початкового положення тягарців буде безпосереднє введення значень абсцис (всі ординати дорівнюють нулю). Відкрийте будь-який інший математичний пакет (наприклад, Matlab), створіть в ньому функцію

$$f(i)=\begin{cases}i+(i-12)\cdot(20-i)/16, & 12<i<20\\ i, & i\le 12 \text{ або } i\ge 20\end{cases}$$

для обчислення початкових абсцис точок механічної системи, і застосуйте її до всіх значень $i=0,1,2,\ldots,32$. Отримані числа введіть в полі **Initial Coordanates** – $\mathbf{q_{x,o}}$ у вигляді діапазону, наприклад, через кому і у фігурних дужках.

Mathematical Particle Tracing (pt) – Release from Grid 1:

Initial Coordanates – $\mathbf{q_{x,0}}$ = `{0,1,2,3,4,5,6,7,8,9,10,11,12, 13+7/16,14+3/4,15+15/16,17,17+15/16,18+3/4,19+7/16, 20,21,22,23,24,25,26,27,28,29,30,31,32}`

Зменшіть похибки обчислення (відносну і абсолютну), одночасно збільшивши розглядуваний часовий інтервал.

Study 1 – Step 1: Time Dependent:

Study Settings – **Times** = `range(0,1,10*n))`

– **Relative tolerance** = `True, 1E-8`

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

Absolute Tolerance – **Tolerance** = `1E-8`

Переобчисліть модель.

`Home` -> Compute

Графік у вузлі **Particle Trajectories (pt)** буде перебудований. Для наочності збільшіть вертикальний масштаб.

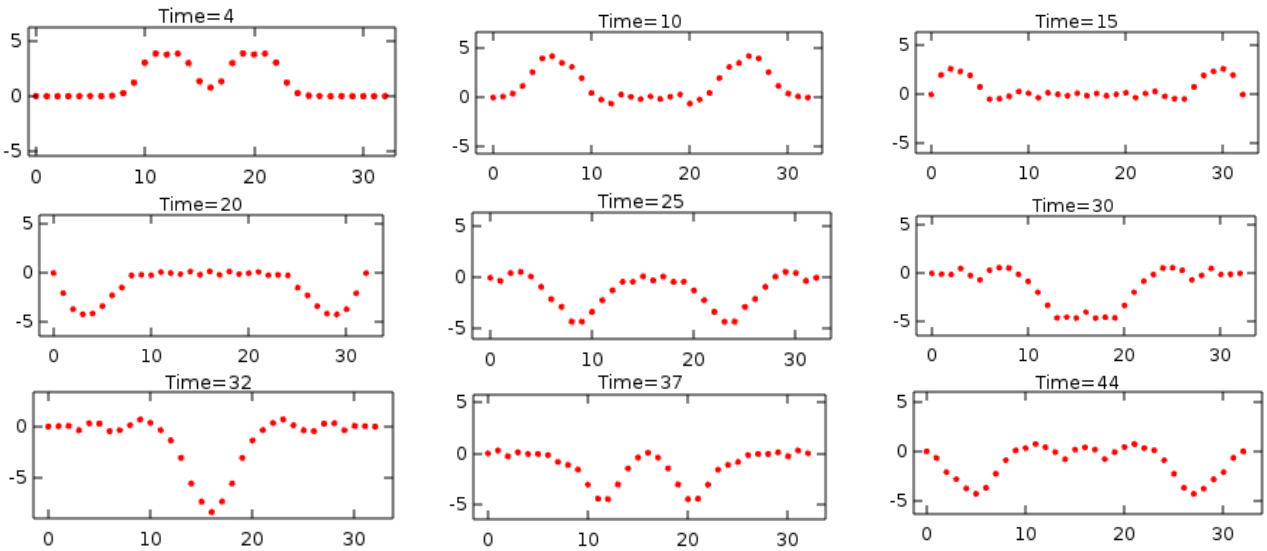
Particle Trajectories (pt) - Scatter Surface 1 : RMB > Delete

Particle Trajectories (pt) - Particle Trajectories 1 - Deformation 1:

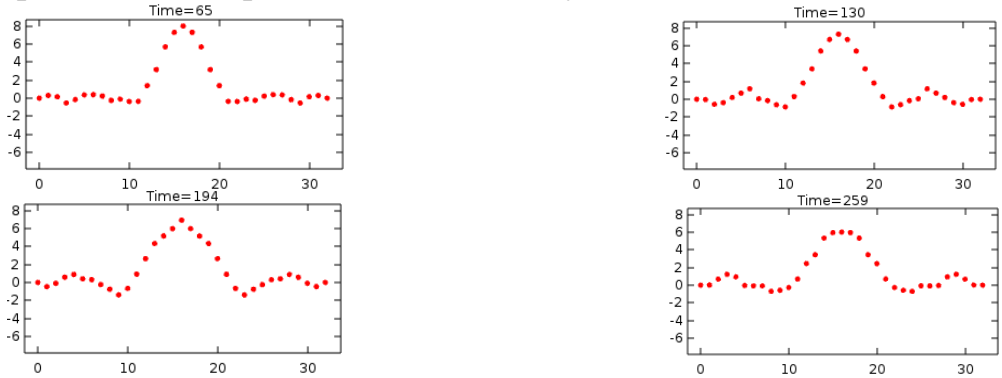
Expression – **y component** = `(qx-(pt.pidx-1))*8`

Оновіть вузол анімації і запустіть її.

На наступних рисунках наведено форми механічної системи в різні моменти часу (зміщення важків відкладено у вертикальному напрямку).



Профіль системи при `t=0`, показаний на початку прикладу, ділиться навпіл. Одна частина рухається вліво, інша – вправо. При досягненні кінців хвильові пагорбки відбиваються зі зміною знака амплітуди і рухаються до середини і далі. В момент часу `t=2n` профіль механічної системи приблизно повторює початковий. На наступному рисунку наведено графіки системи на початку другого, третього, четвертого і п'ятого циклу коливань



Оскільки втрати енергії в системі немає (в рівняннях відсутні члени з першою похідною по часу), то коливання повинні якось продовжуватись нескінченно. Читач може самостійно дослідити поведінку механічної системи в подальші моменти часу, збільшивши розглядуваний часовий інтервал.

Приклад 3. Ланцюжок Фермі – Пасти - Улама.

Дослідити систему ЗДР (2) при $n=32$, $m=1$, $a=1$, $k=1$, $\alpha=1$, нульовій початковій швидкості $x'(0)=0$ і початковому зміщенні важків у формі $x_i(0)=\sin(\pi i/n)$ першої моди лінійної системи. Дослідження виконати у часі приблизно до моментів `t=5000`.

Оскільки задача розв'язуватиметься в `Mathematical Particle Tracing` інтерфейсі, то аналогічно до прикладу 1 потрібно переписати рівняння, використовуючи не зсуви x_j частинок, а їх координати q_j.

В прикладі 1 ми бачили, що $x_{i+1}-x_i=q_{i+1}-q_i-a$, $x_{i-1}-x_i=q_{i-1}-q_i+a$. Тоді, аналогічно до прикладу (1), задача (2) набуде наступного вигляду (в лінійних членах доданки $\pm a$ скорочуються)

$$\frac{d^2 q_i}{d t^2} = \left((q_{i+1} - q_i) + (q_{i+1} - q_i - a)^2\right) - \left((q_i - q_{i-1}) + (q_i - q_{i-1} - a)^2\right) \qquad (5)$$

$$(i = 1,\ 2,\ \ldots,\ n-1)$$

$$q_0(t) = 0,\ q_n(t) = n \cdot a,\ q_i(0) = i \cdot a + \sin(\pi i / n),\ \frac{d q_i}{d t}(0) = 0.$$

Розв'язання. Відкрийте файл `CrystalModel_1D.mph` з прикладу 1 і для внесення в нього змін, збережіть з іншим ім'ям.

При розв'язанні задачі потрібно змінити початкові координати точок механічної системи у вузлі **Release from Grid 1**, а також у вузлі **Particle - Particle Interaction 1** змінити вираз діючої між тягарцями сили.

Mathematical Particle Tracing (pt) – Release from Grid 1:

Initial Coordanates – $\mathbf{q_{x,0}}$ = `range(0,1,32)+ sin(pi*range(0,1,32)/32)`

Mathematical Particle Tracing (pt) – Particle - Particle Interaction 1:

Force – **Force: F <x>** =

```
if(dest(pt.pidx)==1 ||
    dest(pt.pidx)==n+1, 0,
  if(pt.pidx==dest(pt.pidx)-1,
      qx-dest(qx)-(dest(qx)-qx-a)^2,0)+
  if(pt.pidx==dest(pt.pidx)+1,
      qx-dest(qx)+(qx-dest(qx)-a)^2,0))
```

Останній вираз реалізує праву частину формули (5). Сили, діючі на кінцеві частинки дорівнюють нулю. Сила, діюча на внутрішню частинку, наприклад з індексом i, з боку частинки $i-1$, обчислюється за формулою $q_{i-1} - q_i - (q_i - q_{i-1} - a)^2$, а з боку частинки $i+1$ – за формулою $q_{i+1} - q_i + (q_{i+1} - q_i - a)^2$. Якщо в цих формулах q_i прийняти за `dest(qx)`, а q_{i-1}, q_{i+1} – за `qx`, то отримаємо введений вираз.

Задайте діапазон зміни часу `t`.

Study 1 – Step 1: Time Dependent:

Study Settings – **Times** = `range(0,1,170*n)`

При дослідженні задач на великому інтервалі часу потрібно обрати достатньо малі значення відносної і абсолютної похибок. Відносна похибка задається у цьому ж вузлі.

Study Settings – **Relative tolerance** = `True, 1E-8`

Абсолютна похибка задається у вузлі **Time-Dependent Solver 1**.

Study 1 – Solver Configurations – Solution 1 - Time-Dependent Solver 1:

Absolute Tolerance – **Tolerance** = `1E-8`

Виконайте обчислення. Вони триватимуть достатньо довго (кілька хвилин).

`Home` -> Compute

Графік у вузлі **Particle Trajectories (pt)** буде перебудований. Збільшіть в ньому вертикальний масштабний множник з `5` до `10`.

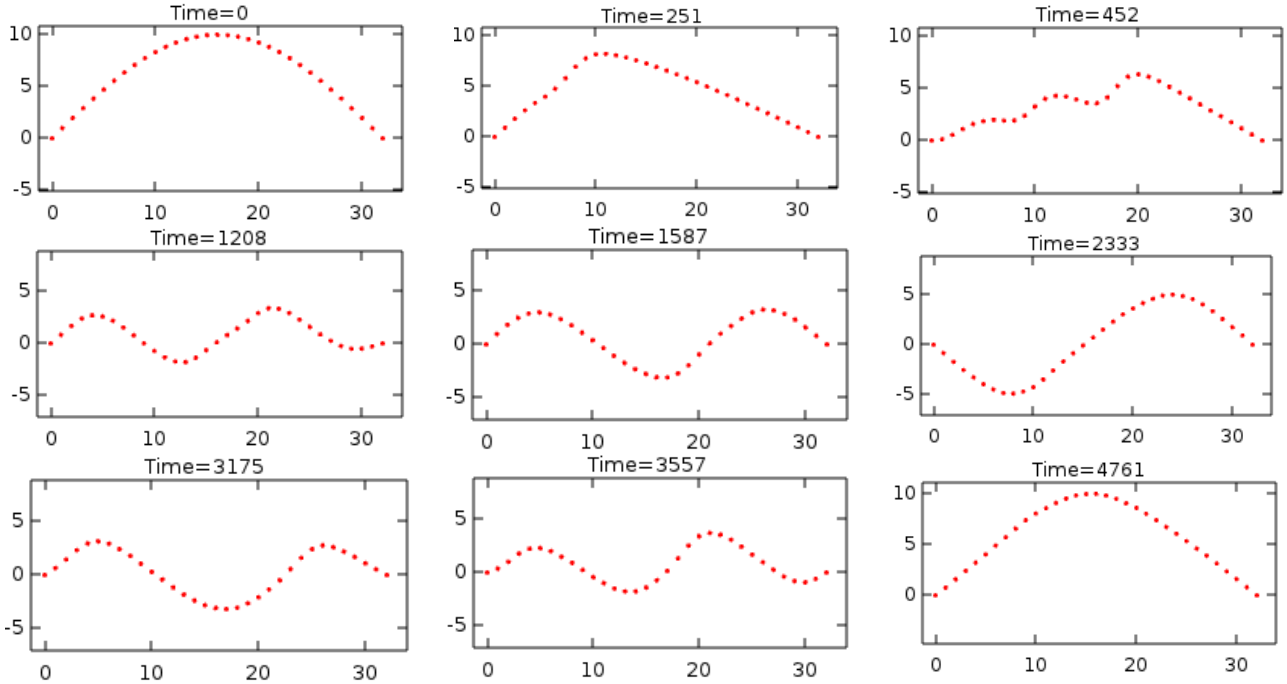
Particle Trajectories (pt) - Particle Trajectories 1 - Deformation 1:

Expression – **y component** = `(qx-(pt.pidx-1))*10`

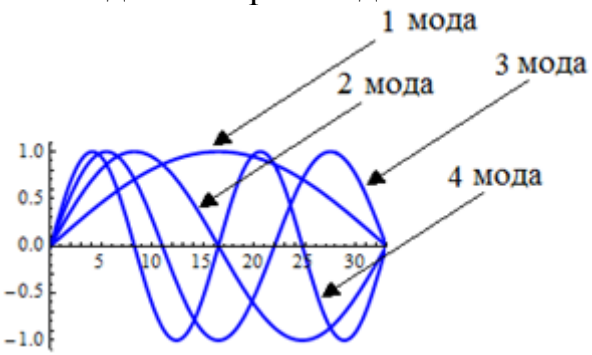
Particle Trajectories (pt) - Scatter Surface 1 : RMB > Delete

Оновіть вузол анімації. Цей процес буде тривати довше обчислень, бо генеруватиметься кілька тисяч кадрів. Запустіть анімацію і проаналізуйте її.

На наступному рисунку наведені форми механічної системи в деякі характерні моменти часу. Поступово спостерігається зсув максимуму від середини відрізка [0,n]. Перша мода ще впізнавана при $t \approx 14 \cdot n$ (див. графік при t=452).



Нагадаємо читачеві як виглядають перші моди коливань лінійної системи.



Аналізуючи графіки, помічаємо, що при $t \approx 38 \cdot n$ (див. попередню таблицю графіків при $t = 1208$) форма коливань механічної системи схожа на 4-у моду лінійної системи. При $t \approx 50 \cdot n$ (див. графік при $t = 1587$) форма системи схожа на третю моду. При $t \approx 73 \cdot n$ (див. графік при $t = 2333$) ми спостерігаємо переважно другу моду. Графіки розв'язків, побудовані при $t \approx 99 \cdot n$ (див. графік при $t = 3175$), знову нагадують 3-ю моду. При $t \approx 111 \cdot n$ (див. графік при $t = 3557$) переважно спостерігається четверта мода. При $t \approx 148 \cdot n$ (див. графік при $t = 4761$) ми повертаємося до першої моді.

Підіб'ємо підсумок, який вперше спостерігали Фермі і його співробітники. В початкові моменти часу форма коливань близька до форми першої моди, але потім починається перекачування енергії в інші моди, і форма розв'язку більше нагадує суму кількох синусоїд з різним періодом (сума мод). При $t \approx 50 \cdot n$ збуджена в основному 3-я мода, при $t \approx 73 \cdot n$ переважає друга мода. Потім при

$t \approx 99 \cdot n$ форма знову близька до третьої моді, і при $t \approx 148 \cdot n$ коливання знову повертаються до першої моді.

Чисельні експерименти співробітників Фермі показали, що така поведінка не випадковість. Збільшення числа важків, зміна значення параметра α, зміна форми нелінійності (наприклад, $\alpha \Delta l^3$ замість $\alpha \Delta l^2$) не змінює поведінку системи. Моди не зливаються, а відбувається виділення деяких з них, які домінують по черзі. Коли повертається початкова мода, все повторюється. Час повернення T_R (в нашому прикладі $t \approx 148 \cdot n$) залежить від n, від виду нелінійності, але «солірування» нижчих мод і повернення при $t = T_R$ спостерігалося співробітниками Фермі у всіх обчисленнях. Відносно недавно [7] показано, що в ланцюжку з'єднаних важок при деяких рівнях енергії система все ж таки може переходити до теплової рівноваги.

Зауваження. `Mathematical Particle Tracing` моделі не призначені для розв'язання схожих задач, тому обчислення прикладів цього параграфа для великих значень часу тривають відносно довго. Наприклад, схожа задача дослідження поведінки ланцюжка Фермі – Пасти – Улама в пакеті Mathematica триває кілька секунд [6], а не хвилин, як в поточному прикладі. Та метою цього параграфа було не стільки знайомство читача з дослідженням поведінки ланцюжка тягарців, скільки подальше знайомство з інструментарієм застосованого інтерфейсу.

Література.

1. Н.И.Левитский "Синтез механизмов по Чебышеву". Изд. акад. наук, М-Л, 1946г., 168 с.
2. А.В. Манжиров, А.Д. Полянин. Методы решения интегральных уравнений: Справочник. М.: «Факториал», 1999. – 272 с.
3. П.Г. Доля. Об одном способе представления кусочных полиномов в системах символьной математики. Вестник Харк. нац. ун-та., – 2008. – № 833. Сер. «Математическое моделирование. Информационные технологии. Автоматизированные системы управления», вып. 10 .
4. В. Вольтерра Теория функционалов, интегральных и интегро-дифференциальных уравнений. М.: Наука, Главная редакция физико-математической литературы, 1982. - 304 с.
5. С.П. Тимошенко. Колебания в инженерном деле. М.: Гос. издательство физико-математической литературы, 1959. - 439 с.
6. http://geometry.karazin.ua/~Dolya/documents: Доля П. Г. Mathematica для математиков. – Х.: ХНУ ім. В.Н. Каразіна, 2015. – 400 с.
7. Onorato M., Vozella L., Proment D., Lvov Y. V. Route to Thermalization in the α–Fermi–Pasta–Ulam System. Proc. Nat. Acad. Sci. U.S.A. 2015. Vol. 112. P. 1–6.

5. Крайові задачі для диференціальних рівнянь.

Система COMSOL Multiphysics призначена для розв’язання статичних і динамічних систем **д**иференціальних **р**івнянь в **ч**астинних **п**охідних (ДРЧП) другого порядку в одно–, дво– або тривимірному просторі. Вона використовує три основні форми рівнянь. Коефіцієнтна форма має вигляд

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \left(-c \nabla u - \alpha u + \gamma \right) + \beta \nabla u + \mathrm{a}\, u = f \,. \qquad (1)$$

Рівняння в цьому випадку задаються з використанням коефіцієнтів при похідних шуканих функцій.

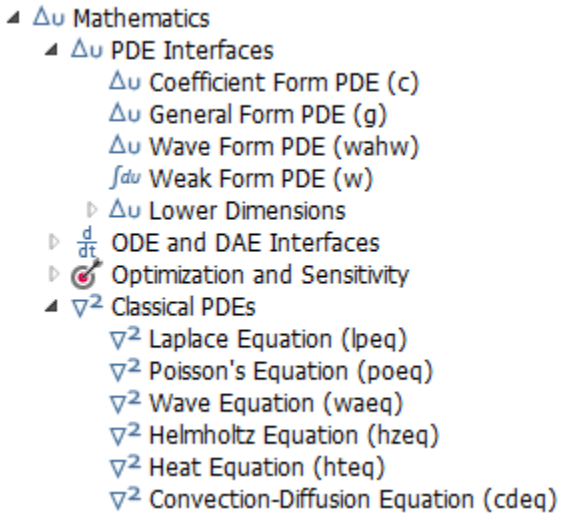
Загальна (генеральна) форма має вигляд

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \Gamma = \mathrm{F}, \qquad (2)$$

де Γ і F можуть бути скалярами або векторами, які залежать від шуканих величин і їх похідних.

Слабка форма використовує не саме ДРЧП, а деяке еквівалентне йому інтегральне співвідношення.

В цій главі піде мова про крайові задачі для звичайних диференціальних рівнянь. Зручніше за все ставити крайові задачі, використовуючи коефіцієнтну форму, яка в дереві інтерфейсів панелі `Select Physics` обирається в гілці `Mathematics - PDE Interfaces - Coefficient Form PDE(c)`. Для моделювання одновимірних крайових задач, окрім коефіцієнтної форми, можна використовувати її окремі види – класичні ДРЧП. Їх ви знайдете в дереві інтерфейсів панелі `Select Physics`, якщо розкриєте гілку `Mathematics - Classical PDEs`.



Для деяких з цих інтерфейсів доступний тільки стаціонарний тип дослідження, для інших - тільки динамічний, для третіх - можливі обидва варіанти. Але щоб одновимірна модель зводилася до звичайного диференціального рівняння або їх системи, вона не повинна залежати від часу, тобто мусить бути стаціонарною.

Чимало одновимірних задач мають аналітичний розв’язок. Ті, для яких це не так, розв'язуються чисельно з мінімальним використанням комп'ютерних ресурсів і в інших математичних пакетах, таких як Matlab, Mathematica або

Maple. Тому дослідження стаціонарних одновимірних задач, як самостійних моделей, не часто використовується в COMSOL Multiphysics. Проте, система здатна розв’язувати ці задачі досить ефективно. Для повноти картини і для тренування читача ми продовжимо розгляд учбових моделей, який розпочато в попередній главі, і опишемо деякі інструменти COMSOL на прикладах розв’язання одновимірних крайових задач для ЗДР.

Зазвичай для існування і єдиності розв’язку ЗДР додаткових умов має бути стільки, який порядок має рівняння. Однак існують крайові задачі, у яких розв’язків немає, наприклад,

$$y'' + y = 0,\ y(0) = 0,\ y(\pi) = 1.$$

Дійсно, загальний розв’язок диференціального рівняння $y'' + y = 0$ має вигляд $y = C_1 \sin x + C_2 \cos x$, де C_1, C_2 довільні сталі. З умови $y(0) = 0$ відразу випливає, що $C_2 = 0$. Але тоді другій граничній умові неможливо задовольнити ні при якому значенні C_1.

З іншого боку, крайова задача $y'' + y = 0$, $y(0) = 0, y(\pi) = 0$ має нескінченну кількість розв’язків виду $y = C \sin x$.

Існує ряд теорем, які дозволяють визначити, вирішувана чи ні крайова задача для ЗДР, а також встановити единість розв'язку [1]. У всіх прикладах цієї глави ми будемо припускати, що розв’язок відповідних задач існує.

5.1. Граничні умови для одновимірних задач.

У цьому параграфі ми обговоримо особливості граничних умов, які можуть використовуватися в одновимірних крайових задачах. Граничні умови в COMSOL Multiphysics для дво- і тривимірних задач - це окрема тема. Однак для одновимірних задач постановка граничних умов відносно проста. Можна задати значення шуканих функцій в граничних точках (гранична умова першого роду, умова Діріхле), або значення перших похідних (граничні умови другого роду, умови Неймана), або комбінацію шуканих функцій і їх похідних (граничні умови третього роду, умови Робіна, мішана крайова задача). Зустрічаються і нестандартні умови, наприклад, коли в граничних умовах з'являються другі похідні шуканих величин по просторовим координатам.

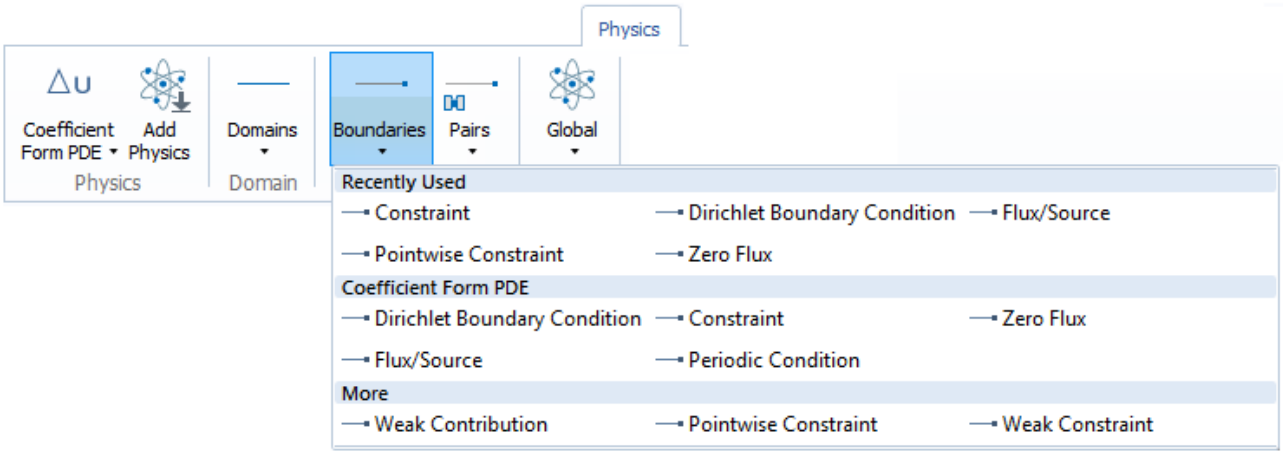
Іноді потрібно враховувати «внутрішні» границі. Наприклад, при з'єднанні двох матеріалів поверхню, що їх розділяє, можна розглядати як внутрішню межу, на якій задано деякі умови контакту. Так в задачі теплопровідності це може бути умова неперервності температури і теплового потоку.

Тут ми обговоримо використання наступних граничних умов:

- умова `Zero Flux` задає нульовий потік поля на межі тіла (умова ізоляції). Зазвичай він (потік) пов'язаний з градієнтом залежної величини. В одновимірних задачах ця умова фактично задає нульові значення похідних шуканої величини в граничних точках;

- умови `Dirichlet Boundary Condition` дозволяють задавати значення шуканих величин на межі розрахункової області (граничні умови першого роду);
- умови `Flux/Source` задають потік поля на межі розрахункової області через комбінації значень шуканих величин і їх перших похідних (граничні умови другого або третього роду);
- умова `Continuity` використовується на внутрішніх границях, визначаючи неперервність шуканих величин і потоків (речовини, тепла, заряду і т.д.);
- умова `Constraint` зазвичай використовує нестандартні зв'язки між шуканими величинами всередині розрахункової області; вона може застосовуватися і на межі тіла, заміняючи граничні умови;
- умова `Periodic Condition` визначає рівність значень шуканих величин на кінцях відрізка розрахункової області.

Граничні умови, які можна використати при розв’язанні крайових задач для одновимірних ЗДР, обираються з випадного списку-кнопки `Boundaries` панелі `Physics` , який зображено на наступному рисунку.



Умову `Continuity` тут не показано, тому що вона обирається з випадного списку команд `Pairs` (пари). Частіше за все пари позначають ділянки меж контактуючіх тіл (тобто внутрішні границі). Фактично пари представляють дві сторони розділової поверхні, які відносяться до різних матеріалів, і на кожній можна задати власну граничну умову. На парі можна також задати умову неперервності фізичного поля або складнішу умову.

Умова неперервності означає неперервність як шуканих величин, так і відповідних фізичних потоків, які зазвичай деяким емпіричних чином через матеріальний закон пов'язані з градієнтом залежної змінної. Наприклад, якщо розглядається загальна (генеральна) форма ДРЧП $e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \Gamma = F$, то величина Г виступає в ролі потоку поля. Тоді неперервність поля означатиме неперервність шуканої величини u і величини Г. Якщо розглядається рівняння теплопровідності $d_a \frac{\partial u}{\partial t} + \nabla \cdot (-c \nabla u) = f$, то величина $-c \nabla u$ представляє потік

тепла, який повинен бути неперервним на межі двох тіл (одночасно з неперервністю температури u). Рівняння Пуассона $\nabla\cdot(-c\nabla u)=f$ можна інтерпретувати як стаціонарне рівняння теплопровідності і в точках стику ділянок, на яких використовуються різні коефіцієнті c і f, умова неперервності означатиме неперервність шуканої величини u і величини $-c\nabla u$.

Зауважимо, що неперервність фізичних потоків на внутрішніх межах зазвичай виконується автоматично без використання пар. Пари корисні, коли умови взаємодії контактуючих тіл накладають між фізичними полями по різні боки внутрішньої межі більш складні взаємозв'язки ніж неперервність.

Для наочності ми наводимо таблицю подібності деяких стаціонарних фізичних полів [2].

Фізичне явище	**Шукана величина**	**Матеріальна константа**	**Потік поля**
Теплопровідність	Температура u	Коефіцієнт теплопровід-ності k	Потік тепла $\boldsymbol{Q}=-k\cdot\mathrm{grad}u$
Дифузія	Концентрація u	Коефіцієнт дифузії D	Потік речовини $\boldsymbol{Q}=-D\cdot\mathrm{grad}u$
Електростатика	Потенціал електричного поля u	Діелектрична стала ε	Вектор електричної індукції $\boldsymbol{D}=\varepsilon\boldsymbol{E}=$ $=-\varepsilon\cdot\mathrm{grad}u$
Магнитостатика	Потенціал магнітного поля u	Магнітна проникність μ	Вектор магнітної індукції $\boldsymbol{B}=-\mu\cdot\mathrm{grad}u$
Електричне поле постійного струму	Потенціал u	Коеф. електро-провідності σ	Густина струму $\boldsymbol{j}=-\sigma\cdot\mathrm{grad}u$

В стаціонарних задачах величина u задовольняє рівнянню Пуассона $\nabla\cdot(-c\nabla u)=f$, де c одна з вище вказаних матеріальних констант. Якщо на деякій поверхні матеріальна константа зазнає розриву, то умову неперервності поля можна представити у вигляді $u_1=u_2$, $c_1\dfrac{\partial u_1}{\partial n}=c_2\dfrac{\partial u_2}{\partial n}$, де індекси 1 і 2 відповідають граничним значенням шуканої величини u по різні боки розглядуваної поверхні, а $\dfrac{\partial}{\partial n}$ позначає похідну по нормалі до поверхні.

В COMSOL Multiphysics при створенні моделі з ДРЧП, автоматично створюється вузол `Zero Flux`, який задає нульові потоки в граничних точках. З точки зору метода скінченних елементів, це природні граничні умови, які не накладають додаткових обмежень на рівняння в слабкій формі. Нагадаємо, що за лаштунками в COMSOL Multiphysics всі ДРЧП переписуються в слабкій формі (інтегральній). Будь які граничні умови, окрім природних, вносять доданки в рівняння слабкої форми. Тому, щоб задати інші умови, для них в

процесі моделювання потрібно створити відповідні вузли граничних умов, які скасують/перевантажать дію вузлів `Zero Flux`, якщо будуть задані на однакових з ними ділянках межі.

5.1.1. Гранична умова Flux/Source.

В ПВ вузла Flux/Source наведено загальний вигляд умови $-\boldsymbol{n}\cdot(-c\nabla u)=g-qu$, де $\boldsymbol{n}$ - одиничний вектор зовнішньої нормалі до межі області. Позначимо одиничний вектор, спрямований уздовж осі X через $\boldsymbol{i}$. В одновимірному випадку для правого граничного вузла $\boldsymbol{n}=\boldsymbol{i}$, а для лівого $\boldsymbol{n}=-\boldsymbol{i}$. Крім того, в одновимірному випадку $\nabla u=\boldsymbol{i}\,\partial u/\partial x$. Тоді для правого граничного вузла матимемо $-\boldsymbol{n}\cdot(-c\nabla u)=-\boldsymbol{i}\cdot(-c\,\boldsymbol{i}\,u'_x)=c\,u'_x$ і гранична умова `Flux/Source` набуває вигляду $c\,u'_x=g-qu$. Аналогічно, для лівого граничного вузла умова матиме вид $-c\,u'_x=g-qu$.

Приклад 1. Розглянемо ЗДР $u''_{xx}=1$ на відрізку [1, 2]. Для його представлення використаємо стаціонарне одновимірне рівняння теплопровідності. На лівому кінці завжди будемо ставити умову Діріхле $u(1)=0$. На правому кінці відрізка граничну умову будемо міняти і порівнювати аналітичний розв'язок $u=(x-1)^2/2+C\cdot(x-1)$ крайової задачі з тим, який будуватиме система (константа C буде залежати від другої граничної умови).

За допомогою майстра створіть одновимірну модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Heat Equation(hteq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): **`Unit System`** – <> = `None`

Геометрія моделі складається з відрізка [1,2]. Створіть його.

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1):

`Interval` – **Left endpoint, Right endpoint** = `1,2`

Interval 1: LMB > Build All Objects

Побудуйте СкЕ сітку.

Mesh 1:

`Mesh Settings` –**Element size** = `Finer`

LMB> Build All

Перевірте, що в ПВ вузла **Heat Equation (hteq) - Heat Equation 1** в розділі **`Domain Selection`** в секції вибору стоїть одиниця (індикатор відрізка). В розділі **`Equation`** записано рівняння теплопровідності в формі

$$d_a\frac{\partial u}{\partial t}+\nabla\cdot(-c\nabla u)=f\,.$$

Параметри рівняння задайте в полях нижньої частини ПВ.

Heat Equation (hteq) - Heat Equation 1:

`Diffusion Coefficient` – **c** = `-1`

Source Term – **f** = 1

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Цим задається рівняння $u''_{xx} = 1$.

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

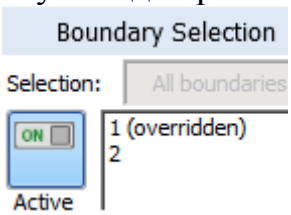
Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** <> = Manual,1 (*лівий кінець відрізка*)

Dirichlet Boundary Condition – **r** = 0

1. Задача: $u''_{xx} = 1$, $u(1) = 0$, $u'_x(2) = 0$.

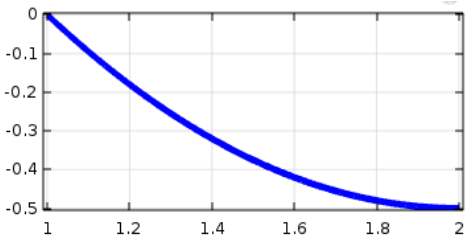
Другу граничну умову залиште незмінною. Вона за замовчуванням визначена в ПВ вузла **Zero Flux 1**, в секції вибору якого вказується індикатор вузла (в нашому випадку правого), в якому похідна розв'язку обертається на нуль.



В результаті будуть використовуватися граничні умови $u(1) = 0$, $u'_x(2) = 0$.

Виконайте обчислення і отримайте графік розв'язку.

Home -> Compute



Для порівняння самостійно додайте в гілку **1D Plot Group 1** графік аналітичного розв'язку $u(x) = \frac{1}{2}(3 - 4x + x^2)$ (графіки повинні співпасти).

2. Задача: $u''_{xx} = 1$, $u(1) = 0$, $u'_x(2) = 1$.

Задайте на правому кінці відрізка граничну умову типу Flux/Source. При $c = -1$ вона має вигляд $-u'_x = g - qu$.

Physics -> **Boundary** - **Boundaries** – Flux/Source

Heat Equation (hteq) - Flux/Source 1:

Boundary Selection – **Selection,** < > = Manual,2 (*правий кінець відрізка*)

Boundary Flux/Source – **g** =-1

Boundary Absorption/Impedance Term – **q** =0

Цим задається умова $-u'_x = -1$. Умова **Zero Flux 1** на правому кінці скасується автоматично.

Клацніть правою кнопкою миші по вузлу самого верхнього рівня (**root**) і виконайте команду Add Study.

Untitled.mph (root) : RMB > Add Study

У відкрившемуся віконці Add Study подвійним клацанням виберіть Stationary тип дослідження і закрийте вікно. В дереві моделі з'явиться вузол **Study 2**.

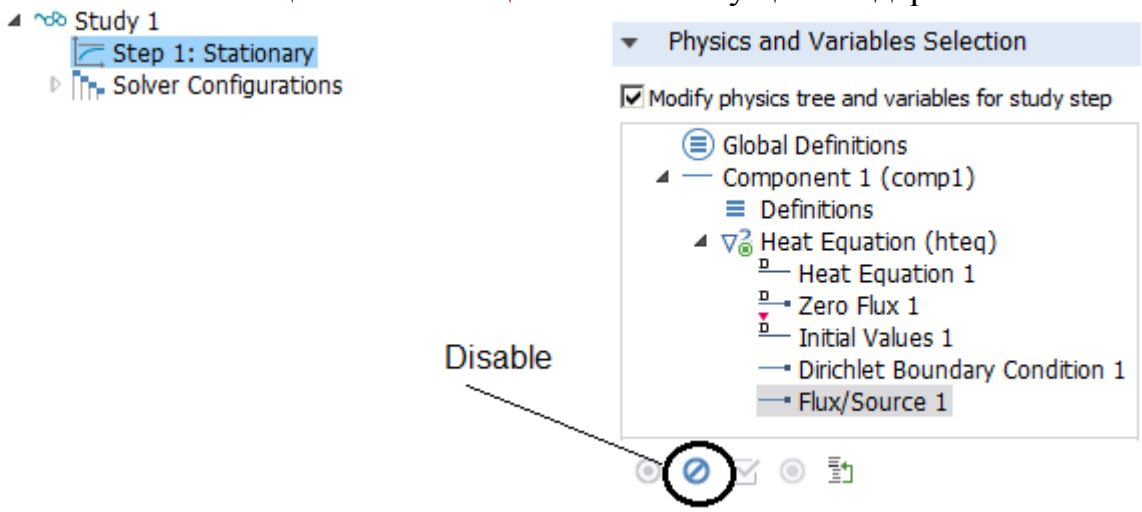
Щоб перше обчислення не змінилося, виключіть вузол **Flux/Source 1** з першого дослідження **Study 1**.

Study 1 – Step 1 : Stationary:

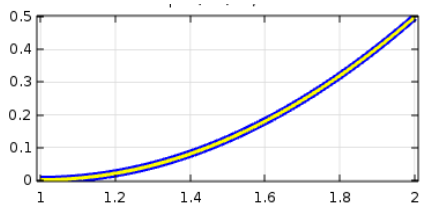
Physics and Variables Selection –

Modify physics tree and variables for study step = True

Після установки перемикача під ним відкриється дерево моделі, в якому деактивуйте вузол **Flux/Source 1**. Для цього в новому ДМ виділіть ім'я вузла **Flux/Source 1** і клацніть по кнопці Disable внизу цього «дерева».



Якщо зараз виділити вузол **Study 1** і виконати команду Compute, то ви знову отримаєте перший розв'язок. Якщо виділити вузол **Study 2** і виконати команду Compute, то буде побудовано другий розв'язок (і другий графік), який використовує нову граничну умову $u'_x(2)=1$.



Для порівняння самостійно додайте в графічну групу **1D Plot Group 2** вузол **Line Graph 2** з графіком аналітичного розв'язку $u(x)=\frac{1}{2}(1-2\,x+x^2)$ (графіки збігаються).

3. Задача $u''_{xx}=1, u(1)=0, u'_x(2)=-2+2\cdot u(2)$ має аналітичний розв'язок $u(x)=\frac{1}{2}((x-1)^2+4\,(x-1))$. Те, що $u''_{xx}=1$ і $u(1)=0$ видно відразу. Легко перевіряється і друга умова $u'_x(x)\big|_{x=2}=-2+2\cdot u(x)\big|_{x=2}$.

Для нового розв'язку можна створити ще один вузол Study, але ми цього робити не будемо. Внесемо зміни в граничну умову $u'_x=-g+qu$ на правому кінці відрізка.

Heat Equation (hteq) - Flux/Source 1:

Boundary Flux/Source – **g** =2

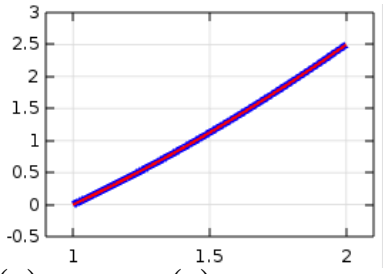
Boundary Absorption/Impedance Term – q =2

Цим задається умова $-u'_x = 2 - 2u$.

Виконайте обчислення вузла **Study 2**

Study 2: RMB > Compute

і отримайте графік розв'язку в графічній групі **1D Plot Group 2**. В цій групі побудуйте також графік аналітичного розв'язку $u(x) = \frac{1}{2}((x-1)^2 + 4\,(x-1))$, замінивши формулу у вузлі **Line Graph 2** (якщо ви його вже створили). Криві повинні співпасти.



4. Задача: $u''_{xx} = 1, u(1) = 0, u'_x(2) = -2 + u(2)$. Розв'язку немає!

Внесіть зміни в ПВ вузла **Flux/Source 1:**

Boundary Absorption/Impedance Term – q =1

і для вузла **Study 2** спробуйте виконати обчислення.

Study 2: RMB > Compute

Система виведе повідомлення про помилку `Failed to find solution`.

При `q=1` наша крайова задача розв'язку не має! Дійсно, з представлення розв'язку $u = (x-1)^2/2 + C\cdot(x-1)$, яке враховує ліву граничну умову, випливає, що $u(2) = 1/2 + C$ і $u'_x(2) = 1 + C$. Але тоді співвідношення $1 + C = -2 + 1/2 + C$, яке отримується з другої граничної умови, не може виконатися ні при якому значенні C.

5. Задача $u''_{xx} = 1, u\,(1) = 0, u'_x\big|_{x=2} = -(x + x\cdot u)\big|_{x=2}$.

В якості коефіцієнтів `g` і `q` слід застосувати вирази, в які входить незалежна змінна `x`. Але в граничній умові $u'_x = -g(x) + q(x)u$ замість аргументу `x` буде використовуватися його значення в граничній точці, зокрема, в нашому прикладі це буде число `x=2`. В результаті умова $u'_x = -x - x\cdot u$ на правому кінці відрізка буде проінтерпретована як $u'_x(2) = -2 - 2\cdot u(2)$.

Внесіть зміни в праву граничну умову (в точці $x = 2$).

Heat Equation (hteq) - Flux/Source 1:

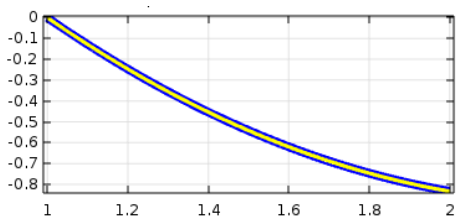
Boundary Flux/Source – g =x

Boundary Absorption/Impedance Term – q =-x

Study 2: LMB > Compute

Цим задається умова у вигляді $-u'_x(x)\big|_{x=2} = x - (-x)\cdot u(x)\big|_{x=2}$ (нагадаємо, що $c = -1$).

Для порівняння самостійно змініть в групі **1D Plot Group 2** графік аналітичного розв'язку, задавши у вузлі **Line Graph 2** формулу $u(x) = (x-1)^2/2 - 4(x-1)/3$.



6. В умові $u'_x = -g + qu$ коефіцієнти g і q можуть залежати від шуканої функції нелінійним чином. Зокрема, задача $u''_{xx} = 1, u(1) = 0, u'_x\big|_{x=2} = -\left(u^2 + u\right)\big|_{x=2}$ має аналітичні розв'язки $u(x) = \frac{1}{2}(x-1)^2 + \left(\pm\frac{1}{\sqrt{2}} - \frac{3}{2}\right)(x-1)$. Нелінійність граничної умови призводить до того, що існує два розв'язки крайової задачі.

Внесіть зміни в граничну умову на правому кінці відрізка.

Heat Equation (hteq) - Flux/Source 1:

Boundary Flux/Source – **g** = u^2

Boundary Absorption/Impedance Term – **q** =-1

Study 2: LMB > Compute

У вузлі **1D Plot Group 2** з'явиться графік розв'язку нової задачі. В графічній групі побудуйте криві обох аналітичних розв'язків.

1D Plot Group 2 - Line Graph 2:

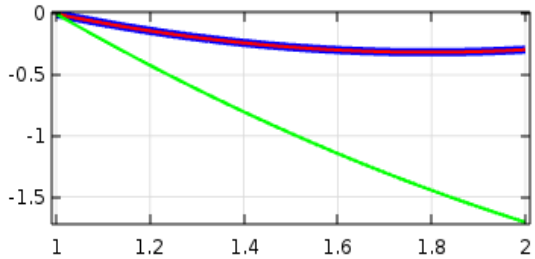
y-Axis Data – **Expression**= (x-1)^2/2+(1/sqrt(2)-3/2)*(x-1)

1D Plot Group 2 - Line Graph 2: RMB > Duplicate

1D Plot Group 2 - Line Graph 3:

y-Axis Data – **Expression**= (x-1)^2/2+(-1/sqrt(2)-3/2)*(x-1)

LMB > Plot



Графік показує, що чисельний алгоритм вибрав перший розв'язок (з плюсом).

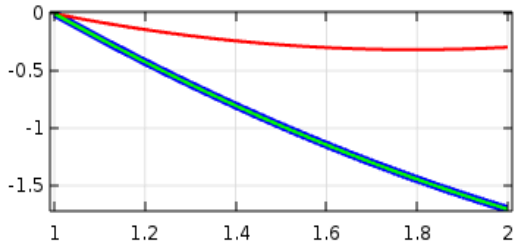
Змініть значення у вузлі **Initial Values 1**. Воно в стаціонарній задачі розглядається як початкове наближення. В останньому обчисленні ми його не чіпали, і воно покладалося рівним нулю.

Heat Equation (hteq) - Initial Values 1:

Initial Values – **Initial value for u** = -2

Study 2: LMB > Compute

В гілці **1D Plot Group 2** графік буде перебудований.



Тепер чисельний розв’язок збігся з другим аналітичним розв’язком.

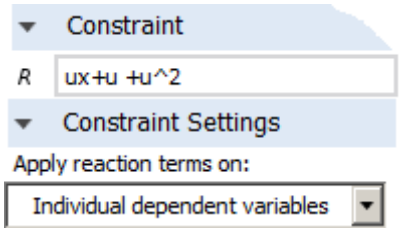
Спробуйте задати початкове наближення, яке дорівнює `-1`. Буде взагалі побудовано невірний розв’язок. Наявність нелінійності в граничній умові вимагає акуратного вибору початкових наближень і додаткового дослідження на можливість неєдиності розв’язку! Намагайтеся уникати нелінійних граничних умов.

Збережіть файл, але не закривайте його.

5.1.2. Гранична умова Constraint.

Гранична умова `Constraint` (зв'язок, закріплення) використовується тоді, коли стандартні граничні умови непридатні, наприклад, коли в умову входить друга похідна шуканої величини.

Гранична умова `Constraint` (зв'язок) задається у вигляді $0 = R(x, u, \ldots)$. Якщо в ПВ такого вузла встановлена опція **`Individual dependent variables`**, то поле **R** являє вираз, обертання на нуль якого викликано відповідними значеннями u в граничній точці. Всі `Flux/Source` обмеження, наведені вище, можуть бути представлені у вигляді обмежень `Constraint`. Наприклад, в пункті 6 останнього прикладу умова на правому кінці $-u'_x(2) = u^2(2) + u(2)$ може бути представлена у вигляді `ux+u+u^2=0` і використана у вузлі типу `Constraint`.



Перевіримо це.

7. Задача $u''_{xx} = 1, u(1) = 0, u'_x\big|_{x=2} = -\left(u^2 + u\right)\big|_{x=2}$.

Відкрийте файл останнього прикладу і виконайте інструкції:

`Physics` **-> Boundary – Boundaries –** Constraint

Heat Equation (hteq) – Constraint 1:

`Boundary Selection` – **Selection,** <> = `Manual,2` (*правий кінець відрізка*)

– **R** = `ux+u+u^2`

`Constraint Settings` – **Apply reaction terms on** =

`Individual dependent variables`

Клацніть правою кнопкою миші по вузлу самого верхнього рівня (**root**) і виконайте команду Add Study.

Untitled.mph (root) : RMB > Add Study

Якщо ви зберігали файл, то назва вузла верхнього рівня буде іншою. У відкрившемуся віконці `Add Study` подвійним клацанням виберіть `Stationary` тип дослідження і закрийте вікно. В дереві моделі з'явиться вузол **Study 3**.

Щоб перше дослідження не змінилося в ПВ вузла **Study 1 – Step 1: Stationary** в дереві моделі деактивуйте вузол **Constraint 1** (вузол **Flux/Source 1** деактивований раніше).

Study 1 – Step 1: Stationary:

`Physics and Variables Selection` –

Heat Equation (hteq) – Constraint 1: RMB > Disable

Щоб другий розв'язок не змінився, виключіть вузол **Constraint 1** з другого дослідження (**Study 2**).

Study 2 – Step 1: Stationary:

`Physics and Variables Selection` –

Modify physics tree and variables for study step = `True`

`Physics and Variables Selection` – **Heat Equation (hteq) –**

– Constraint 1: RMB > Disable

Налаштуйте третє дослідження (**Study 3**), виключивши з його ДМ вузол **Flux/Source 1**.

Study 3 – Step 1 : Stationary:

`Physics and Variables Selection` –

Modify physics tree and variables for study step = `True`

`Physics and Variables Selection` –

Heat Equation (hteq) – Flux/Source 1: RMB > Disable

Поверніть нульові початкові наближення і виконайте обчислення.

Heat Equation (hteq) - Initial Values 1:

`Initial Values` – **Initial value for u** = `0`

Study 3: LMB > Compute

В графічній групі **1D Plot Group 3** ви повинні отримати криву, яка повторює графік першого розв'язку п.6 поточного прикладу.

8. Задача $u''_{xx} = 1, u(1) = 0, u''_{xx} + 8u\big|_{x=2} = 0$. Друга гранична умова має нетиповий вигляд і її можна задати у вузлі типу `Constraint`.

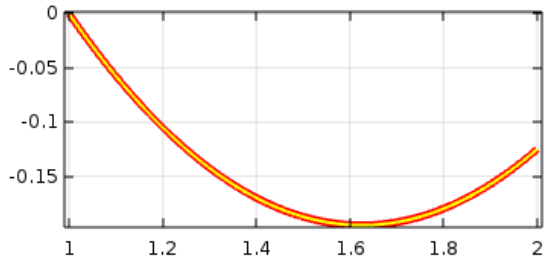
Відкрийте файл останнього прикладу і змініть налаштування вузла **Constraint 1** так, щоб в ньому була задана зазначена умова. Потім виконайте обчислення.

Heat Equation (hteq) – Constraint 1:

`Boundary Constraint` – **R** = `uxx+8*u`

Study 3: LMB > Compute

У графічну групу **1D Plot Group 3** самостійно додайте графік аналітичного розв'язку $u = \frac{1}{8}(9 - 13\,x + 4\,x^2)$. Криві повинні співпасти.



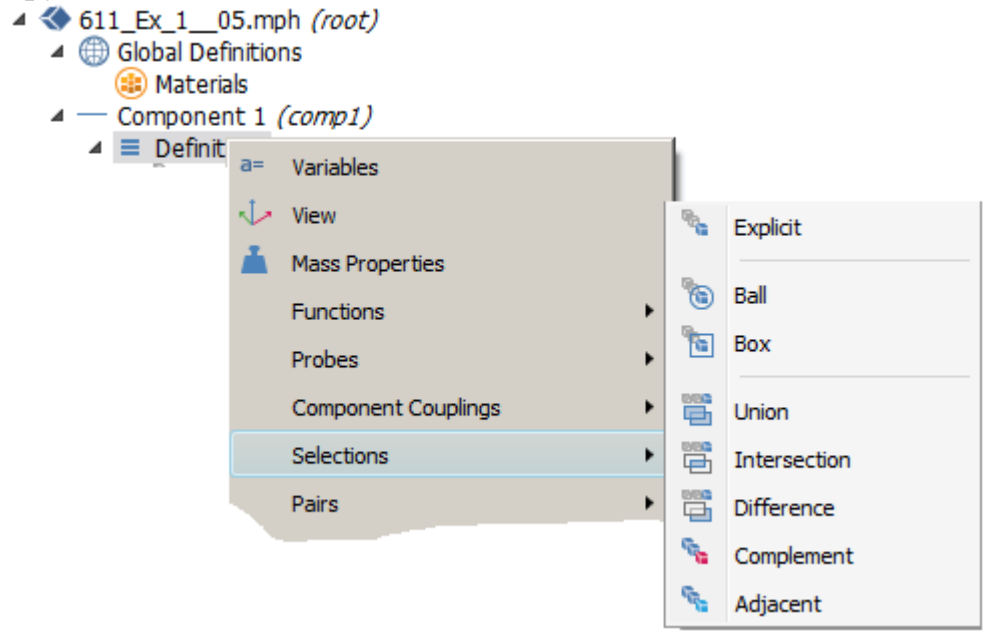
Збережіть файл зі старим ім'ям, а також для подальшої роботи з новим, наприклад, DemoPoisson1.mph , і не закривайте його.

Зауваження. На панелі Physics є випадна кнопка-список **Global** з командою Global Constraint, за допомогою якої накладаються обмеження на змінні моделі, які не залежать від просторових координат.

5.1.3. Гранична умова Periodic Condition.

Гранична умова Periodic Condition використовується тоді, коли шукану величину треба зробити рівною на лівій і правій межах розрахункової області, тобто $u(x_{left}) = u(x_{right})$. Налаштування вузла Periodic Condition дозволяють також реалізовувати умову антиперіодичності, тобто $u(x_{left}) = -u(x_{right})$. При створенні умови періодичності в гілці Definitions компонента автоматично створюється Selection вузол явного виділення (Explicit).

Selection (виділення) вузли використовуються для іменування групи геометричних об'єктів, яким дається спільне ім'я. Їх використовують разом при заданні на них однакової умови (наприклад, граничної). Вони створюються в гілці Definitions з випадного меню командою Selections > спосіб вибору. Якщо виділення (Selection) буде виконуватися за допомогою інструментів вибору окремих геометричних сутностей, то створюється вузол типу Explicit. Виділені геометричні сутності можуть повністю або частково знаходиться всередині кулі (вузол вибору типу Ball), повністю або частково знаходитися всередині паралелепіпеда (вузол вибору типу Box), бути логічним об'єднанням, перетином, різницею або доповненням вже створених вибірок (вузли вибору типу Union, Intersection, Difference, або Complement), бути суміжними до заданих геометричних об'єктів або виділень (вузол вибору типу Adjacent). Тут вузли вибору нам не знадобляться.



При заданні умови періодичності потрібно відрізняти границю-джерело і границю-призначення, які зазвичай визначаються у створюваному автоматично «лише для читання» вузлі вибору `Explicit`. Але користувач може самостійно вказувати межі призначення, додаючи до вузла `Periodic Condition` підвузол `Destination Selection`.

Приклад 1. Розв'язати крайову задачу $u''_{xx} = 1, u(1) = 1, u(2) = u(1)$.

Відкрийте файл `DemoPoisson1.mph`, змініть в ньому значення `u` на лівому кінці і додайте вузол періодичної граничної умови.

Heat Equation (hteq) – Dirichlet Boundary Condition 1:

`Dirichlet Boundary Condition` – **r** = `1`

`Physics` -> **`Boundary - Boundaries`** – Periodic Condition

Heat Equation (hteq) – Periodic Condition 1:

`Boundary Selection` – **Selection,** <> = `Manual,1,2` (*кінцеві точки*)

В полі **Selection** вказуються обидві кінцеві точки, які будуть ототожнюватися. Зверніть увагу на те, що після створення вузла **Periodic Condition 1** автоматично створюється вузол **Definitions – Explicit 1**.

Деактивуйте в ДМ вузол **Constraint 1**.

Heat Equation (hteq) – Constraint 1: RMB > Disable

Якщо бажаєте, можете виключити вузол **Periodic Condition 1** з послідовності обчислень вузлів **Study 1 – Step 1 : Stationary** і **Study 2 – Step 1 : Stationary**.

Виконайте обчислення.

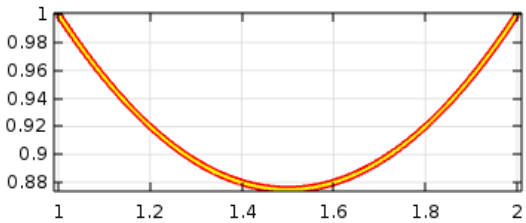
Study 3: LMB > Compute

В графічній групі **1D Plot Group 3** задайте новий вираз аналітичного розв'язку $u = 2 - \frac{3}{2}x + \frac{1}{2}x^2$.

1D Plot Group 3 - Line Graph 2:

`y-Axis Data` – **Expression**= `2-3*x/2+x^2/2`

LMB > Plot



Збережіть модель в файл, але не закривайте його.

Приклад 2. Розв'язати задачу: $u''_{xx} = 1$, $u(2) = u(1)$ з додатковою умовою, що середнє значення величини u по відрізку [1,2] дорівнює 1. Для її розв'язання в поточну модель потрібно внести невеликі зміни.

Створіть в моделі оператор обчислення середнього значення по відрізку.

`Definitions` **`-> Coupling - Component Couplings -`** Average

Component 1 - Definitions - Average 1 (aveop1):

`Source Selection` – **Geometric entity level** = `Domain`

– **Selection -** < > = `Manual,1` (*відрізок* [1,2])

Ім'я оператора `aveop1` в полі **Operator name** залиште без змін. Нагадаємо, що оператор `aveop1(expr)` обчислює середнє значення виразу `expr` (в нашому прикладі по відрізку [1,2]).

Залиште в ДМ вузол **Periodic Condition 1**.

Щоб середнє величини *u* набуло заданого значення, потрібно правильно підібрати граничне значення. Позначимо його ідентифікатором `pf` (воно зараз невідоме) і використаємо у вузлі граничної умови Діріхле.

Dirichlet Boundary Condition 1:

`Boundary Selection` – **Selection,** <> = `Manual,1` (*лівий кінець відрізка*)

`Dirichlet Boundary Condition` – **r** = `pf`

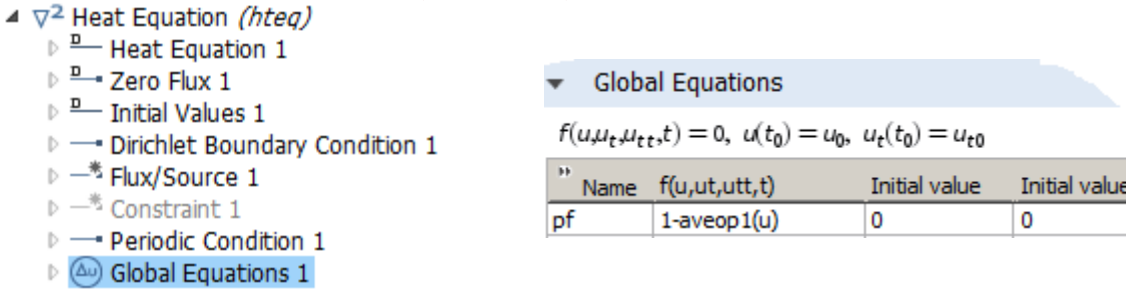
Для визначення невідомої `pf` додайте в модель вузол рівнянь Global Equations.

`Physics` **`-> Global`** – Global Equations

Heat Equation (hteq) - Global Equations 1:

`Global Equations` – **Name** = `pf`

`Global Equations` – **f(u,ut,utt,t)** = `1-aveop1(u)`



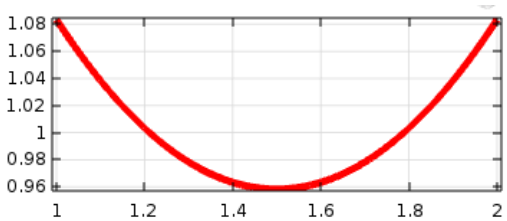
Розуміти запис в таблиці потрібно наступним чином. Задача буде розв'язуватися відносно невідомої `pf` одночасно з рівнянням у вузлі **Heat Equation 1** доки вираз в комірці **f(u,ut,utt,t)** не буде дорівнювати нулю, тобто доки середнє значення `aveop1(u)` не стане заданою величиною.

Таким чином, в нашій задачі створено дві шукані величини: `u` та `pf`. При чому `u` – це польова змінна (функція від *x*), а `pf` – невідомий скаляр.

Виконайте обчислення.

Study 3: LMB > Compute

Після їх завершення в графічній зоні гілки **1D Plot Group 3** буде побудований графік величини `u`.



Для перевірки добудуйте самостійно в цій групі графік аналітичного розв'язку $U(x)=\frac{1}{2}(\frac{25}{6}-3x+x^2)$.

5.1.4. Умова Continuity.

При розв'язанні деяких задач диференціальне рівняння на різних ділянках потрібно розглядати з різними значеннями коефіцієнтів. Це еквівалентно тому,

що на різних частинах розрахункової області використовуються різні матеріали з різними фізичними властивостями. В таких задачах тіло потрібно розглядати не як суцільне, а як збірку, складену з кількох деталей, матеріал кожної з яких має свої фізичні властивості. Але тоді при постановці задачі потрібно визначати умови зв'язку фізичних полів в зістикованих тілах.

Коли конструкція розглядається як збірка кількох тіл, в дереві моделі COMSOL в гілці `Geometry` слід вказати `Assembly` тип з'єднання елементів геометрії. Зазвичай за замовчуванням використовується об'єднання (`union`) геометричних фігур. В результаті чого створюється суцільне тіло. Альтернативою є збірка (`assembly`). В цьому випадку система інтерпретує геометрію як колекцію фігур (замість їх об'єднання). Це приводить до того, що користувач сам повинен визначити способи зв'язку фізичних полів в цих тілах за допомогою додаткових інструментів, одним з яких є «пари» (pairs). Частіше за все пари позначають ділянки меж контактуючих тіл. На них можна застосовувати різноманітні «граничні» умови, зокрема, умову неперервності шуканого поля, яка в одновимірному випадку для рівняння Пуассона $\nabla\cdot(-c\nabla u)=f$ має вигляд $u_1=u_2$, $c_1u'_{1x}=c_2u'_{2x}$, де індекси $1, 2$ відповідають граничним значенням шуканої величини u ліворуч і праворуч від розділювальної точки.

Для задання умови неперервності фізичного поля використовується вузол `Continuity`, який потрібно створювати в гілках кожного з рівнянь (фізик), діючих на сусідніх ділянках розрахункової області.

Приклад. Розглянемо задачу

$$u''_{xx}=8x,\ x\in[0,1],\ u(0)=0, \tag{1}$$

$$(4v'_x)'_x=2-x,\ x\in[1,2],\ v(2)=0 \tag{2}$$

з умовами стикування

$$u(1)=v(1),\ u'_x(1)=4v'_x(1). \tag{3}$$

У цієї задачі розв'язком є кускова функція

$$w(x)=\begin{cases}\frac{1}{3}\left(-\frac{29}{5}x+4x^3\right) & ,\ 0\le x\le 1\\ \frac{1}{24}\left(-\frac{114}{5}+\frac{17}{5}x+6x^2-x^3\right), & 1\le x\le 2\end{cases} \tag{4}$$

яку можна буде порівняти з розв'язком, отриманим в COMSOL.

Обидва рівняння (1) і (2) можна розглядати як одновимірні рівняння Пуассона $(-c\,w'_x)'_x=f(x)$. Для першого рівняння матимемо $c=c_1=-1$, $f(x)=f_1(x)=8x$. Для другого $c=c_2=-4$, $f(x)=f_2(x)=2-x$. Оскільки другу умову (3) можна записати у вигляді $-c_1u'_x=-c_2v'_x$, то (3) і є умовою неперервності, яку реалізує вузол `Continuity`.

Р о з в ' я з о к 1. Для розв'язання задачі (1) – (3) за допомогою майстра створіть одновимірну модель з двома рівняннями Пуассона.

File –> New –> Model Wizard –> 1D –> Mathematics -
Classical PDEs – Poisson's Equation (poeq) –>
Classical PDEs – Poisson's Equation (poeq2) + Variables = u –> Stationary

При створенні моделі ми відразу додали дві фізики/рівняння, і в обох задали однакове ім'я u шуканої змінної. Коли на різних ділянках геометрії потрібно розв’язувати однотипні рівняння, то можна використовувати однакові імена шуканих величин. Це не приводитиме до конфлікту, оскільки рівняння діятимуть на різних відрізках.

Untitled.mph (root): `Unit System` – <> = `None`

В цій задачі корисно встановити прапорець, який визначатиме відображення індикаторів елементів геометрії.

Component 1 – Definitions - View 1:

`View` – Show geometry labels = `True`

Створіть геометрію моделі, яка складається з двох пристикованих відрізків, і з'єднайте їх в збірку (assembly).

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1) :

`Interval` – Left endpoint, Right endpoint = `0,1`

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 2 (i2) :

`Interval` – Left endpoint, Right endpoint = `1,2`

LMB > Build All Objects

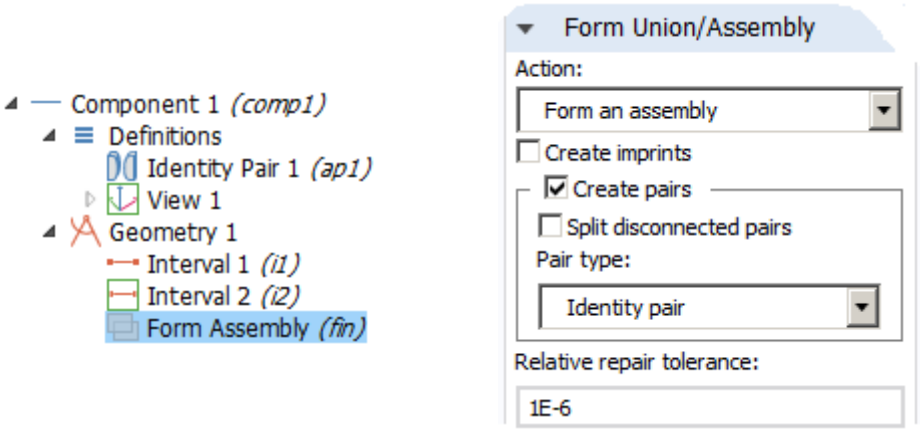
Geometry 1 - Form Union:

`Form Union/Assembly` – Action = `Form an assembly`

Остання опція змінить назву вузла на **Form Assembly (fin)**.

`Form Union/Assembly` – Create pairs = `True`

Ця інструкція вказує системі створювати пари (за замовчуванням типу `Identity Pair`), які позначають контактуючі ділянки меж зістикованих тіл.



Зауваження. Вузол `Identity Pair` створюється в дереві моделі в гілці **Definitions** автоматично після визначення розрахункової геометрії кожної фізики, тобто після вибору робочих відрізків в гілках «фізик». Спочатку вузлів пар не видно.

Зауваження. Вважається, що відрізки примикають один до одного, якщо відстань між ними (їх кінцями) менше значення, заданого в полі **Relative repair tolerance**.

Генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

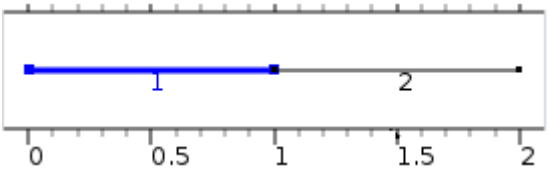
Mesh Settings – **Element Size** =Finer

LMB> Build All

Визначте геометрію, на якій діятиме перше рівняння Пуассона, а також коефіцієнти цього рівняння.

Component 1 - Poisson's Equation (poeq):

Domain Selection – **Selection**=Manual,1 (*відрізок* [0,1])



Component 1 - Poisson's Equation (poeq) - Poisson's Equation 1:

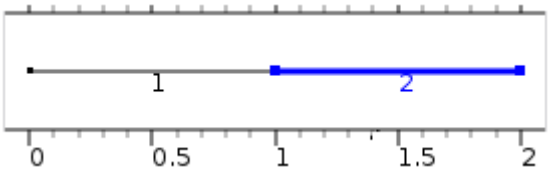
Diffusion Coefficient – **c** = -1

Source Term – **f** = 8*x

Визначте геометрію для другого рівняння Пуассона і задайте його коефіцієнти.

Component 1 - Poisson's Equation 2 (poeq2):

Domain Selection – **Selection**=Manual,2 (*відрізок* [1,2])



Component 1 - Poisson's Equation 2 (poeq2) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -4

Source Term – **f** = 2-x

Задайте нульові граничні умови на кінцях розрахункової області. Оскільки точка x=0 входить в розрахункову область першого рівняння, то граничну умову $u(0)=0$ потрібно задавати в гілці **Poisson's Equation (poeq)**.

Component 1 - Poisson's Equation (poeq):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1 (*лівий кінець x*=0)

Dirichlet Boundary Condition – **r** = 0

Точка x=2 входить в розрахункову область другого рівняння. Тому граничну умову $u(2)=0$ потрібно задавати в гілці **Poisson's Equation 2 (poeq2)**.

Component 1 - Poisson's Equation 2 (poeq2):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation 2 (poeq2) - Dirichlet Boundary Condition 1:

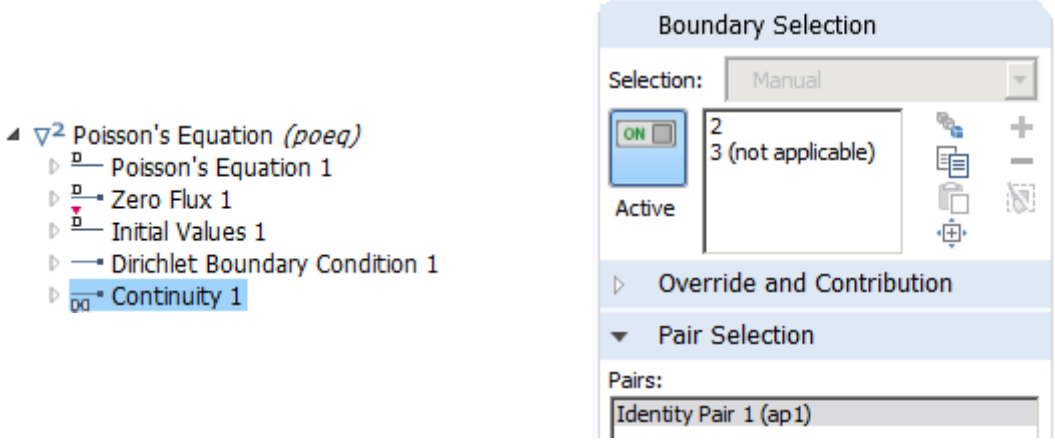
Boundary Selection – **Selection** = Manual,4 (*правий кінець x*=2)

Dirichlet Boundary Condition – **r** = 0

Додайте умову неперервності у вузлі стикування відрізків [0,1] і [1,2]. Її потрібно задавати в обох фізиках.

Poisson's Equation (poeq): RMB > Pairs – Continuity

В ПВ створеного вузла **Continuity 1** в розділі **`Pair Selection`** в списку **Pairs** клацніть по рядку `Identity Pair 1 (ap1)`.



В розділі **`Pair Selection`** обираються пари, які визначають місця накладання умови неперервності. Як наслідок в списку **`Selection`** вказані індикатори точок «робочої» пари. Нагадаємо, що кінці відрізка [0,1] мають індикатори 1 і 2, а кінці відрізка [1,2] – індикатори 3 і 4. Тобто індикатори 2 і 3 позначають «різні сторони» спільної точки цих відрізків.

В гілці фізики **Poisson's Equation 2 (poeq2)** створіть вузол `Continuity` аналогічно попередньому.

Poisson's Equation 2 (poeq2): RMB > Pairs – Continuity

Poisson's Equation 2 (poeq2) – Continuity 1:

`Pair Selection` – **Pairs** = `Identity Pair 1 (ap1)`

Виконайте обчислення.

`Home` **`-> Study`** - Compute

Створюється дві гілки з однаковими графіками (кожна «фізика» побудувала власний графік розв'язку `u`). Видаліть другу гілку.

Results - 1D Plot Group 2: RMB > Delete

Для порівняння в першій групі створіть графік «аналітичного» розв'язку $w(x)$. Для цього спочатку в моделі створіть кускову функцію, яка обчислюється за формулою (4).

Component 1(comp1):

`Definitions` -> **`Functions`** **->** Piecewise

Function name = `pw1`

`Definition` – **Intervals (таблиця)** -

– **Start, End, Function** = `0,1, (-29*x/5+4*x^3)/3`

– **Start, End, Function** = `1,2, (-114/5+17*x/5+6*x^2-x^3)/24`

Щоб функція стала доступною, потрібно оновити розв'язок.

Study 1: RMB > Update Solution

В гілку **1D Plot Group 1** додайте графік цієї функції.

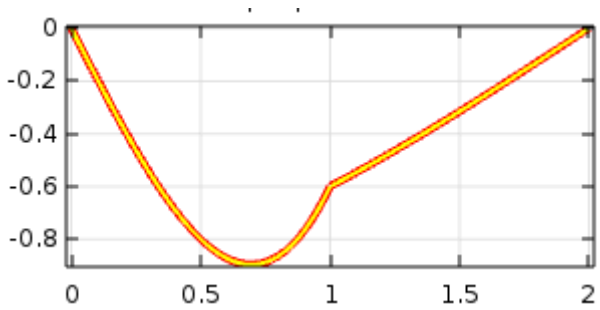
Results - 1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 2 :

`y-Axis Data` – **Expression** = `pw1(x)`

Стиль кривої налаштуйте так, щоб вона відрізнялася від графіка розв'язка *u*.

LMB > Plot



Як бачите, графіки співпадають.

Розв'язок 2 (без використання пар). Насправді в розв'язку 1 ми значно «перемудрували». Якщо у вузлі Geometry елементи геометрії поєднані за допомогою вузла Union, і на сусідніх ділянках використовується однакове ім'я шуканої величини, то автоматично припускається, що на межі стику матеріалів фізичне поле неперервне. Перевіримо це, розв'язавши ту ж саму задачу. Послідовність дій схожа на розв'язок 1, але з неї виключено створення і використання пар.

File -> New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs – Poisson's Equation (poeq) ->
Classical PDEs – Poisson's Equation (poeq2) + Variables = u -> Stationary

Untitled.mph (root): `Unit System` – <> = `None`

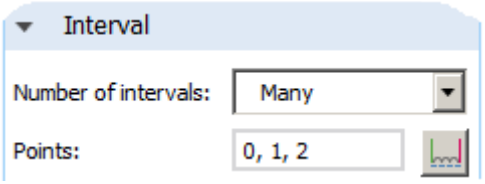
Два з'єднаних відрізка (тобто збірка з них не вийде) можна створити в одному вузлі `Interval`.

`Geometry` -> **`Primitives`** -> Interval

Geometry 1 - Interval 1 (i1) :

Interval – Number of intervals = `Many`

Interval – Points = `0,1,2`



LMB > Build All Objects

Генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

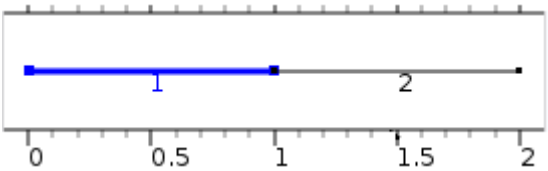
`Mesh Settings` – **Element Size** =`Finer`

LMB> Build All

Визначте геометрії, на яких діятимуть рівняння Пуассона, а також задайте їх коефіцієнти.

Component 1 - Poisson's Equation (poeq):

`Domain Selection` – **`Selection`**=`Manual,1` (*відрізок* [0,1])



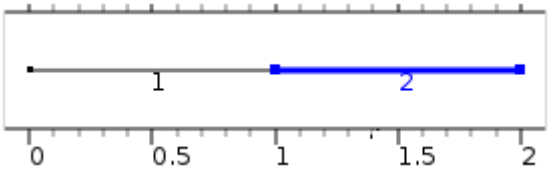
Component 1 - Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = 8*x

Component 1 - Poisson's Equation 2 (poeq2):

Domain Selection – **Selection**=Manual,2 (*відрізок* [1,2])



Component 1 - Poisson's Equation 2 (poeq2) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -4

Source Term – **f** = 2-x

Задайте нульові граничні умови на кінцях розрахункової області.

Component 1 - Poisson's Equation (poeq):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1 (*лівий кінець x*=0)

Dirichlet Boundary Condition – **r** = 0

Component 1 - Poisson's Equation 2 (poeq2):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation 2 (poeq2) - Dirichlet Boundary Condition 1:

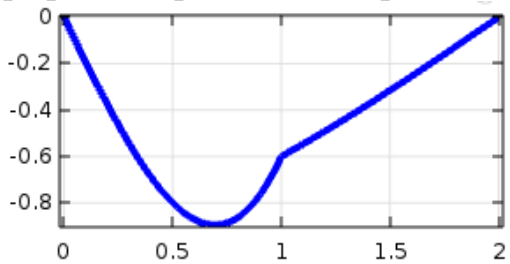
Boundary Selection – **Selection** = Manual,3 (*правий кінець x*=2)

Dirichlet Boundary Condition – **r** = 0

Виконайте обчислення.

Home **-> Study** - Compute

Ви отримаєте такий же графік, як і раніше (насправді два однакових графіка).



5.1.5. Внутрішні граничні умови.

Коли в моделі досліджується фізичний процес, що відбувається в тілі, складеному з двох матеріалів, поверхню, що їх розділяє, можна розглядати як внутрішню межу. Тоді на ній потрібно задавати деякі «граничні» умови. Так в задачі теплопровідності це може бути умова неперервності температури і теплового потоку. Інколи на такій поверхні відомі шукані величини або їх

потоки. Зустрічаються випадки коли на різних сторонах такої межі ці величини повинні бути різними. В цьому параграфі ми розглянемо способи, які застосовуються в COMSOL Multiphysics для накладання умов на «внутрішніх» границях.

Приклад 1. Розглянемо задачу

$$u''_{xx} = -1, \quad x \in [0,1], u(0) = 0, \tag{1}$$

$$4v''_{xx} = 16, \quad x \in [1,2], v(2) = 0 \tag{2}$$

з різними умовами стикування в точці $x = 1$:

1. Умова не задається, тобто автоматично виконуються умови $u'_x(1) = 0$, $v'_x(1) = 0$. На різних ділянках використовуються різні імена шуканих величин.
2. Умова не задається, але використовується спільне ім'я `u` шуканої величини; це призводитиме до автоматичного використання умови неперервності у вигляді $u(1) = v(1)$, $u'_x(1) = 4v'_x(1)$.
3. Використовуються умови `Constraint` $u(1) = v(1)$, $u'_x(1) = v'_x(1)$.
4. В першій фізиці використовуватиметься умова Діріхле $u(1) = v(1)$, а в другій – умова Flux/Source $u'_x(1) = v'_x(1)$.

При створенні моделі у всіх випадках використовуватимуться дві фізики в одному компоненті. Вони діятимуть на сусідніх відрізках, поєднаних операцією «Form Union».

Р о з в ' я з о к . Для розв'язання задачі (1) – (2) за допомогою майстра створіть одновимірну модель з двома рівняннями Пуассона.

File -> New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs – Poisson's Equation (poeq) ->
Classical PDEs – Poisson's Equation (poeq) + Variables = v -> Stationary

Зверніть увагу, що для розв'язку другого рівняння Пуассона ми обрали ім'я `v`.

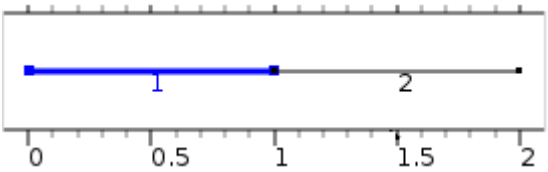
Untitled.mph (root): `Unit System` – <> = `None`
`Geometry` -> **`Primitives`** -> Interval

Geometry 1 - Interval 1 (i1) :
Interval – Number of intervals = `Many`
Interval – Points = `0,1,2`
LMB > Build All Objects

Component 1 (comp1) - Mesh 1 :
`Mesh Settings` – Element Size =`Finer`
LMB> Build All

Визначте геометрії, на яких діятимуть рівняння Пуассона (1) і (2), а також задайте їх коефіцієнти.

Poisson's Equation (poeq):
`Domain Selection` – `Selection`=`Manual,1` (*відрізок* [0,1])



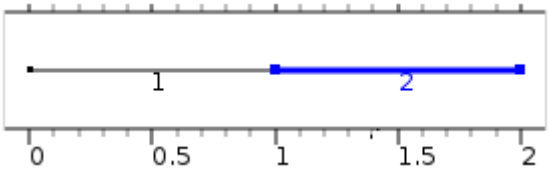
Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = 1

Source Term – **f** = 1

Poisson's Equation 2 (poeq2):

Domain Selection – **Selection**=Manual,2 (*відрізок* [1,2])



Poisson's Equation 2 (poeq2) - Poisson's Equation 1:

Diffusion Coefficient – **c** = 4

Source Term – **f** = -16

Задайте нульові граничні умови на кінцях відрізка [0,2].

Оскільки лівий кінець належить розрахунковій області першого рівняння, то в точці x=0 граничну умову Діріхле потрібно задавати в гілці **Poisson's Equation (poeq)**.

Poisson's Equation (poeq):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1 (*лівий кінець* x=0)

Dirichlet Boundary Condition – **r** = 0

Правий кінець належить розрахунковій області другого рівняння. Тому в точці x=2 граничну умову Діріхле слід задавати в гілці **Poisson's Equation 2 (poeq2)**.

Poisson's Equation 2 (poeq2):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation 2 (poeq2) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,3 (*правий кінець* x=2)

Dirichlet Boundary Condition – **r** = 0

1. Зверніть увагу, що в створеній моделі імена шуканих величин різні: в першому рівняння Пуассона – u, а в другому – v. Розв'яжемо задачу, не задаючи ніяких умов на внутрішній межі (точка $x=1$). Тоді використовуватимуться умови за замовчуванням $u'_x(1)=0$, $v'_x(1)=0$.

Home -> **Study** – Compute

Створюються дві графічні групи (кожна «фізика» креслить графік власного розв'язку). Поєднаємо криві функцій u і v, а також для порівняння побудуємо графік «аналітичного» розв'язку $w_1 = \begin{cases} \frac{1}{2}(2x - x^2) \\ 2(-2x + x^2) \end{cases}$.

Спочатку створіть кускову функцію, яка обчислюється за останньою формулою.

Component 1(comp1):

Definitions -> **Functions** -> Piecewise

Function name = w1

Definition – **Intervals (таблиця)** -

– **Start, End, Function** = `0,1, -(x^2-2*x)/2`

– **Start, End, Function** = `1,2, (x^2-2*x)*2`

Щоб функція стала доступною, потрібно оновити розв'язок.

Study 1: RMB > Update Solution

В гілку **1D Plot Group 1** додайте графік розв'язку `v`, побудований над відрізком [1,2], і графік функції w_1, побудований над відрізком [0,2].

1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 2 :

Selection – **Selection** = `Manual,2` (*відрізок* [1,2])

y-Axis Data – **Expression** = `v`

1D Plot Group 1 - Line Graph 2: RMB > Duplicate

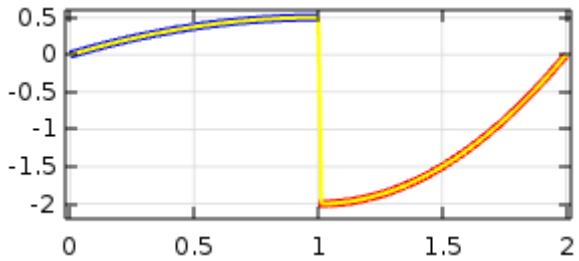
1D Plot Group 1 - Line Graph 3 :

Selection – **Selection** = `All domains`

y-Axis Data – **Expression** = `w1(x)`

Стилі кривих налаштуйте на свій розсуд.

LMB > Plot



Видаліть другу непотрібну графічну групу.

Results - 1D Plot Group 2: RMB > Delete

Збережіть файл, але не закривайте його.

2. У вузлі **Poisson's Equation 2 (poeq2)** змініть ім'я розв'язку.

Poisson's Equation 2 (poeq2):

Dependent Variables – **Dependent variable** = `u`

Одночасно у вузлі **1D Plot Group 1 - Line Graph 2** поміняйте ім'я величини, графік якої будується.

1D Plot Group 1 - Line Graph 2 :

y-Axis Data – **Expression** = `u`

Спільне ім'я шуканої функції в обох «фізиках» призводить до автоматичного використання умови неперервності величини `u` і потоку поля, тобто до умов $u(1-0)=u(1+0)$, $u'_x(1-0)=4u'_x(1+0)$, де 1 ± 0 позначає праву і ліву границю в точці `x=1`. Умова за замовчуванням $u'_x(1)=0$, $v'_x(1)=0$, яка використовувалася системою раніше, не застосовуватиметься.

Аналітичний розв'язок цієї задачі має вигляд

$$w_2=\begin{cases}-\dfrac{1}{2}x\,(2+x)\\ 1-\dfrac{9}{2}x+2\,x^2\end{cases}$$

Створіть цю функцію.

Component 1(comp1):

`Definitions` -> **`Functions`** -> Piecewise

Function name = `w2`

`Definition` – **Intervals (таблиця)** -

– **Start, End, Function** = `0,1, -(x+2)*x/2`

– **Start, End, Function** = `1,2, 2*x^2-9*x/2+1`

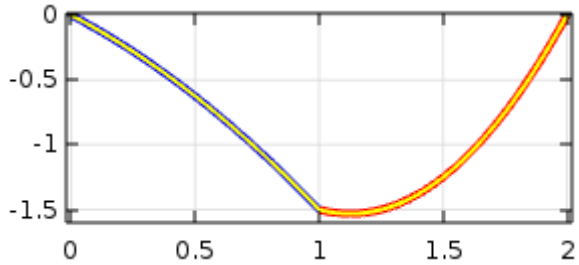
У вузлі **1D Plot Group 1 - Line Graph 3** поміняйте ім'я функції.

1D Plot Group 1 - Line Graph 3 :

`y-Axis Data` – **Expression** = `w2(x)`

Заново обчисліть модель і отримайте графік нового розв'язку.

`Home` -> **`Study`** – Compute



Зауваження. При однаковому імені розв'язку ліворуч і праворуч від точки стику умови за замовчуванням типу `Zero Flux` ігноруватимуться, і застосовуватиметься умова неперервності шуканої величини і її потоку. Будь-які інші «внутрішні» умови, наприклад, Діріхле або Flux/Source, будуть враховуватися і умова неперервності фізичного поля тоді не застосовуватиметься.

Збережіть файл з іншим ім'ям, але не закривайте його.

3. Для внутрішньої межі (в точці `x=1`) в обох «фізиках» створіть вузли обмежень `Constraint` і налаштуйте їх так, щоб вони реалізували умови стикування $u(1)=v(1)$, $u'_x(1)=v'_x(1)$.

Спочатку для розв'язку другого рівняння Пуассона поверніть ім'я `v`.

Poisson's Equation 2 (poeq2):

`Dependent Variables` – **Dependent variable** = `v`

В першій фізиці створіть вузол обмежень `Constraint`.

Poisson's Equation (poeq):

`Physics` -> **`Boundaries`** – Constraint

Poisson's Equation (poeq) – Constraint 1:

`Boundary Selection` – **Selection,** <> = `Manual,2` (*точка x*=1)

– **R** = `v-u`

`Constraint Settings` – **Apply reaction terms on** =

`Individual dependent variables`

В другій фізиці у вузлі `Constraint` задайте інше обмеження $u'_x(1)=v'_x(1)$.

Poisson's Equation 2 (poeq2):

`Physics` -> **`Boundaries`** – Constraint

Poisson's Equation 2 (poeq2) – Constraint 1:

Boundary Selection – **Selection,** <> = Manual,2 (*точка* x=1)
– **R** = vx-ux
Constraint Settings – **Apply reaction terms on** =
Individual dependent variables

Зауваження. За замовчуванням використовуються двобічні умови Constraint. Двобічність означає, що застосувавши умову в одній фізиці, ви додаєте додаткове обмеження в іншу. Тут ми відмінили двобічність, обравши опцію Individual dependent variables, бо «по різні боки» в точці x=1 застосовуватимуться різні умови.

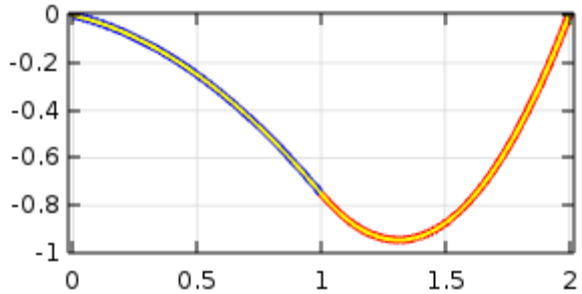
Побудуйте кускову функцію, яка представлятиме «аналітичний» розв'язок цієї крайової задачі

$$w_3 = \begin{cases} -\frac{1}{4}x\,(1+2\,x) \\ \frac{5}{2}-\frac{21}{4}x+2\,x^2 \end{cases}$$

Component 1(comp1):
Definitions -> **Functions** -> Piecewise
Function name = w3
Definition – **Intervals (таблиця)** -
– **Start, End, Function** = 0,1, -(2*x+1)*x/4
– **Start, End, Function** = 1,2, 2*x^2-21*x/4+5/2

Скорегуйте налаштування графіків.
1D Plot Group 1 - Line Graph 2 :
y-Axis Data – **Expression** = v
1D Plot Group 1 - Line Graph 3 :
y-Axis Data – **Expression** = w3(x)

Заново обчисліть модель і отримайте новий графік розв'язків.
Home -> **Study** – Compute



З графіка зрозуміло, що виконані обидві умови: неперервність шуканих величин і їх похідних (маємо спільну дотичну в точці стику).

Збережіть файл ще раз з іншим ім'ям, і не закривайте його.

4. Для внутрішньої межі (в точці x=1) в першій фізиці створіть вузол Dirichlet Boundary Condition, а в другій – вузол Flux/Source, і налаштуйте їх так, щоб вони реалізували умови стикування $u(1)=v(1)$, $u'_x(1)=v'_x(1)$.

Відкрийте поточний файл і деактивуйте в ньому вузли Constraint.

Poisson's Equation (poeq) – Constraint 1: RMB > Disable
Poisson's Equation 2 (poeq2) – Constraint 1: RMB > Disable
Замість деактивованих вузлів створіть вузли нових граничних умов. В фізиці **Poisson's Equation (poeq)** створіть вузол граничної умови Діріхле для виразу $u = v$, а в фізиці **Poisson's Equation 2 (poeq2)** створіть вузол типу `Flux/Source` для задання похідної $v'_x = u'_x$.

Poisson's Equation (poeq):

`Physics` -> **`Boundary - Boundaries`** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 2:

`Boundary Selection` – **Selection** = `Manual,2` (*точка* x=1)

`Dirichlet Boundary Condition` – **r** = `v`

Poisson's Equation 2 (poeq2):

`Physics` -> **`Boundary - Boundaries`** – Flux/Source

Poisson's Equation 2 (poeq2) - Flux/Source 1:

`Boundary Selection` – **Selection,** < > = `Manual,2` (*точка* x=1)

`Boundary Flux/Source` – **g** = `-4*ux`

Пояснимо, чому праворуч потрібний коефіцієнт `-4`. Загальний вид умови вузла `Flux/Source` має вигляд $-\boldsymbol{n}\cdot(-c\nabla v) = g - qv$. В точці x=1 «з точки зору другої фізики» маємо $\boldsymbol{n} = -\boldsymbol{i}$ і $\nabla v = \boldsymbol{i}\,\partial v/\partial x$. При q=0 умова набуває вигляду $-c\,v'_x = g$, де значення $c = 4$ задано у вузлі фізики. Щоб ця умова набула вигляду $v'_x = u'_x$ потрібно покласти $g = -4\cdot u'_x$.

Заново обчисліть модель

`Home` -> **`Study`** – Compute

і отримайте той же графік розв'язків, який наведено при застосуванні вузлів `Constraint`.

■

Як зрозуміло з попереднього прикладу, умову `Constraint` можна застосовувати для «поєднання» розв'язків сусідніх «фізик» за допомогою нетипових умов. Пояснимо це докладніше.

Приклад 2. Розглянемо задачу

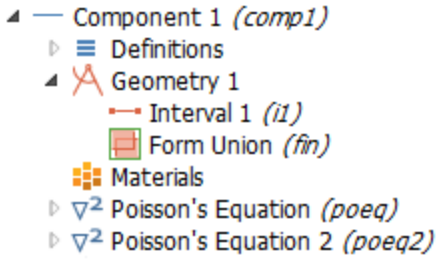
$$u''_{xx} = 1, \quad x \in [0,1], u(0) = -1, \tag{3}$$

$$4v''_{xx} = x, \quad x \in [1,2], v(2) = 3. \tag{4}$$

з різними умовами стикування в точці $x = 1$:

1. $u = v - 1,\ v'_x = u'_x$;
2. $u = v,\ v'_x = u'_x$;
3. $u = v,\ v'_x = u'_x + 3$;
4. $u = 3v,\ v'_x = u'_x$;
5. $u = 3v,\ v'_x = u'_x - 3$;
6. $u = v,\ v'_x = -0.5\,u'_x$;
7. $u = v - 2,\ v'_x = -0.5\,u$;

При створенні моделей для цих задач генеруватимуться дві фізики в спільному компоненті. Рівняння/фізики діятимуть на сусідніх відрізках. Відрізки поєднуватимуться операцією «Form Union».



Імена шуканих величин будуть різними: u і v. Всі умови реалізуватимуться у вузлах типу `Constraint`.

Р о з в ' я з о к . За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs – Poisson's Equation (poeq) ->
Classical PDEs – Poisson's Equation (poeq) + Variables = v -> Stationary

Зверніть увагу, що для розв'язку другого рівняння Пуассона ми обрали ім'я v.

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Геометрія моделі складатиметься з двох прилеглих відрізків [0,1] і [1,2].

`Geometry` -> **`Primitives`** -> Interval

Geometry 1 - Interval 1 (i1) :

Interval – Number of intervals = `Many`

Interval – Points = `0,1,2`

LMB > Build All Objects

Генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

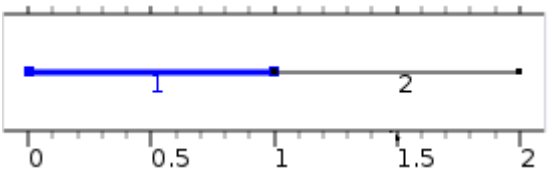
`Mesh Settings` – Element Size =`Finer`

LMB> Build All

Визначте геометрії, на яких діятимуть рівняння Пуассона (3) і (4), а також задайте їх коефіцієнти.

Poisson's Equation (poeq):

`Domain Selection` – `Selection`=`Manual,1` (*відрізок* [0,1])



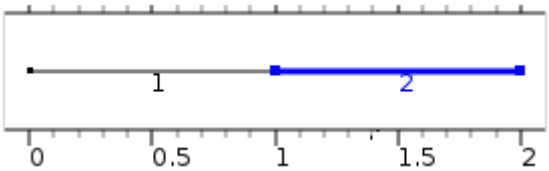
Poisson's Equation (poeq) - Poisson's Equation 1:

`Diffusion Coefficient` – c = `-1`

`Source Term` – f = `1`

Poisson's Equation 2 (poeq2):

`Domain Selection` – `Selection`=`Manual,2` (*відрізок* [1,2])



Poisson's Equation 2 (poeq2) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -4

Source Term – **f** = x

Задайте граничні умови Діріхле на кінцях відрізка [0,2].

Poisson's Equation (poeq):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1 (*точка x*=0)

Dirichlet Boundary Condition – **r** = -1

Poisson's Equation 2 (poeq2):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation 2 (poeq2) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,3 (*точка x*=2)

Dirichlet Boundary Condition – **r** = 3

Для внутрішньої межі (в точці x=1) в обох «фізиках» створіть по одному вузлу обмежень Constraint і налаштуйте їх так, щоб вони реалізовували умови стикування $u = v - 1$, $v'_x = u'_x$.

В першій фізиці створіть вузол обмежень Constraint.

Poisson's Equation (poeq):

Physics -> **Boundary - Boundaries** – Constraint

Poisson's Equation (poeq) – Constraint 1:

Boundary Selection – **Selection,** <> = Manual,2 (*точка x*=1)

– **R** = 1+u-v

Constraint Settings – **Apply reaction terms on** =

Individual dependent variables

Нагадаємо, що вираз, записаний в полі **R**, обертатиметься на нуль.

В другій фізиці теж створіть вузол Constraint і задайте в ньому інше обмеження $v'_x(1) = u'_x(1)$.

Poisson's Equation 2 (poeq2):

Physics -> **Boundary - Boundaries** – Constraint

Poisson's Equation 2 (poeq2) – Constraint 1:

Boundary Selection – **Selection,** <> = Manual,2 (*точка x*=1)

– **R** = vx-ux

Constraint Settings – **Apply reaction terms on** =

Individual dependent variables

Виконайте обчислення.

Home -> **Study** – Compute

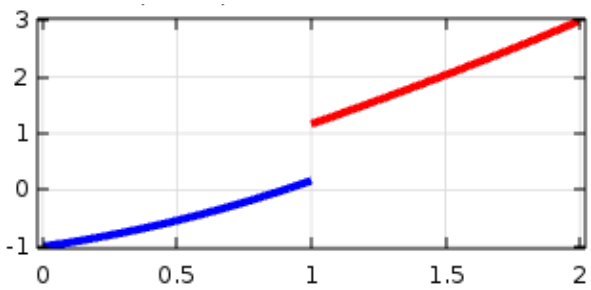
Створюються дві графічні групи (кожна «фізика» побудувала графік власного розв'язку на власному відрізку). Поєднайте криві розв'язків *u* і *v* в одній

графічній групі. Для цього скопіюйте вузол **1D Plot Group 2 – Line Graph 1** в першу групу. Стиль кривих налаштуйте на свій розсуд.

1D Plot Group 2 – Line Graph 1: RMB > Copy

1D Plot Group 1: RMB > Paste Line Graph

LMB > Plot



Аналітичний розв'язок цієї задачі має вид:

$$U = \frac{1}{6}(-6 + 4\,x + 3x^2)$$

$$V = \frac{1}{24}(-10 + 37\,x + x^3)$$

Для перевірки системи ви можете самостійно добудувати графіки функцій U і V в гілці **1D Plot Group 1**.

Корисно подивитися, як виглядатимуть графіки других похідних чисельного розв'язку. Зробимо це в графічній групі **1D Plot Group 2**.

1D Plot Group 2:

Axis –**Manual axis limits** = `True`

Axis –**y minimum, y maximum** = `0,1.1` (*два поля*)

1D Plot Group 2 – Line Graph 1:

Selection – < > = `1` (*індикатор відрізка* [0,1])

y-Axis Data – **Expression** = `uxx`

Coloring and style – **Color, Width** = `Blue, 3`

Legends – **Show legends** = `True`

– **Legends, <>** = `Manual,uxx`

1D Plot Group 2 – Line Graph 1: RMB > Duplicate

1D Plot Group 2 – Line Graph 2:

Selection – < > = `2` (*індикатор відрізка* [1,2])

y-Axis Data – **Expression** = `vxx`

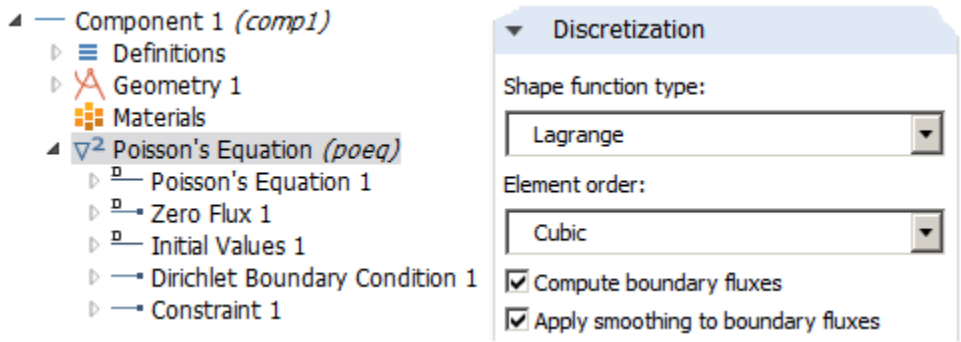
Legends – **Legends, <>** = `Manual,vxx`

LMB > Plot (*наступний графік ліворуч*)

Графік виглядає ламаним. Згладити криву другої похідної можна, якщо підвищити порядок поліноміальних функцій форми, якими на кожному скінченному елементі апроксимуються шукані величини в МСкЕ. В ПВ вузлів фізик є розділ **Discretization** (він не завжди відображатися), в якому визначається тип функцій форми і їх порядок. Зазвичай за замовчуванням встановлено другий (`Quadratic`) порядок. В обох фізиках встановіть третій (`Cubic`) порядок цих функцій.

Poisson's Equation (poeq):

Discretization – **Element order** = Cubic



Poisson's Equation 2 (poeq2):

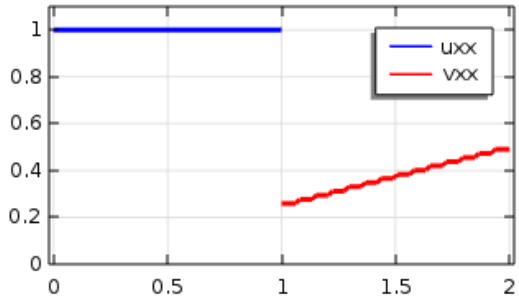
Discretization – **Element order** = Cubic

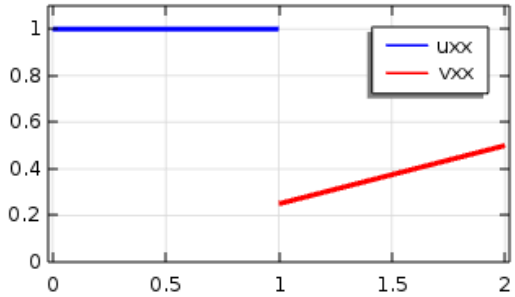
Переобчисліть модель. Графіки в гілці **1D Plot Group 2** покращають.

Home -> **Study** – Compute

1D Plot Group 2: LMB > Plot

(*наступний рисунок праворуч*)

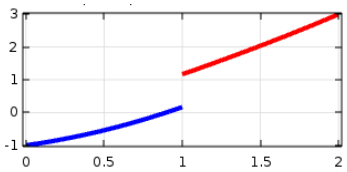
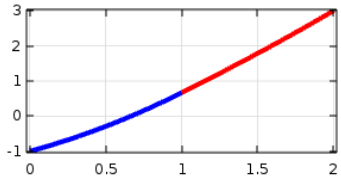
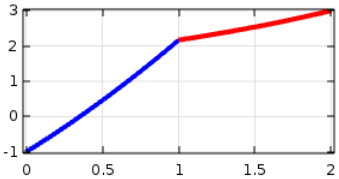


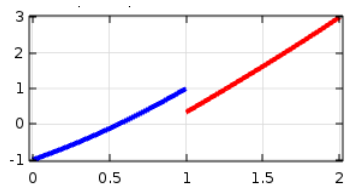
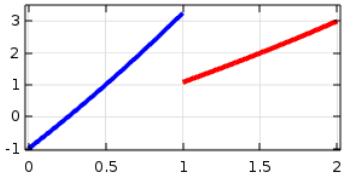
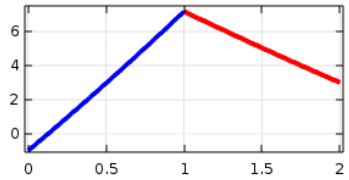
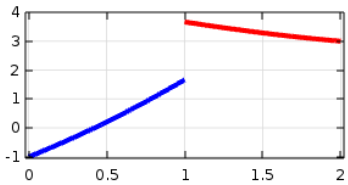


Збережіть файл, але не закривайте його.

Для перевірки інших умов стикування, наведених на початку цього прикладу, ви можете просто змінити у вузлах **Poisson's Equation (poeq) – Constraint 1** і **Poisson's Equation 2 (poeq2) – Constraint 1** вирази в полях **Boundary Selection** – **R**. Зробіть це самостійно. Ми наведемо лише результати, які зберемо в наступну таблицю.

Таблиця 1. *Розв'язки задачі* (3), (4) *з різними умовами стикування в* т. *x*=1.

	Умова стикування	**poeq – R** **poeq2 - R**	Графік	Аналітичний розв'язок
1.	$u = v - 1,$ $v'_x = u'_x$	1+u−v vx−ux		$U = \frac{1}{6}(-6 + 4x + 3x^2)$ $V = \frac{1}{24}(-10 + 37x + x^3)$
2.	$u = v,$ $v'_x = u'_x$	u−v vx−ux		$U = \frac{1}{6}(-6 + 7x + 3x^2)$ $V = \frac{1}{24}(-34 + 49x + x^3)$
3.	$u = v,$ $v'_x = u'_x - 3$	u−v vx−ux+3		$U = -1 + \frac{8x}{3} + \frac{x^2}{2}$ $V = \frac{1}{24}(38 + 13x + x^3$

4.	$u = 3v$, $v'_x = u'_x$	u-3*v vx-ux		$U = \frac{1}{2}(-2 + 3x + x^2)$ $V = \frac{1}{24}(-50 + 57x + x^3$
5.	$u = 3v$, $v'_x = u'_x - 3$	u-3*v vx-ux+3		$U = \frac{1}{4}(-4 + 15x + 2x^2)$ $V = \frac{1}{24}(-14 + 39x + x^3)$
6.	$u = v$, $v'_x = -0.5u'_x$	u-v vx+0.5*ux		$U = \frac{1}{6}(-6 + 46x + 3x^2)$ $V = \frac{1}{24}(278 - 107x + x^3)$
7.	$u = v - 2$, $v'_x = -0.5u$	u-v+2 vx+u/2		$U = \frac{1}{6}(-6 + 13x + 3x^2)$ $V = \frac{1}{24}(110 - 23x + x^3)$

Отже умова типу `Constraint` (зв'язок) дозволяє поєднувати сусідні рівняння/фізики на межі їх стикування. Зверніть увагу на те, що у виразах «на вузлі x=1» доступні обидва розв'язки (лівий і правий, тобто величини u і v).

Зауваження 1. Всі наведені в попередній таблиці умови можна реалізувати за допомогою вузлів типу `Dirichlet Boundary Condition` і `Flux/Source`. Наприклад, покажемо, як можна розв'язати попередню задачу (3), (4) з сьомою умовою стикування $u = v - 2$, $v'_x = -0.5u$. Відкрийте файл поточної моделі і деактивуйте в ньому вузли `Constraint`.

Poisson's Equation (poeq) – Constraint 1: RMB > Disable

Poisson's Equation 2 (poeq2) – Constraint 1: RMB > Disable

Замість деактивованих вузлів створіть вузли інших граничних умов. В фізиці **Poisson's Equation (poeq)** створіть вузол умови Діріхле для виразу $u = v - 2$, а в фізиці **Poisson's Equation 2 (poeq2)** створіть вузол Flux/Source для задання похідної $v'_x = -0.5u$.

Poisson's Equation (poeq):

`Physics` -> **`Boundary - Boundaries`** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 2:

`Boundary Selection` – **Selection** = `Manual,2` (*точка* x=1)

`Dirichlet Boundary Condition` – **r** = `v-2`

Poisson's Equation 2 (poeq2):

`Physics` -> **`Boundary - Boundaries`** – Flux/Source

Poisson's Equation 2 (poeq2) - Flux/Source 1:

`Boundary Selection` – **Selection,** < > = `Manual,2` (*точка* x=1)

`Boundary Flux/Source` – **g** = `-2*u`

Умова вузла `Flux/Source` $-\boldsymbol{n}\cdot(-c\nabla v)=g-qv$ в точці x=1 «з точки зору другої фізики» при q=0 має вид $-cv'_x=g$, де значення $c=-4$ задано у вузлі фізики. Тому, щоб вона набула вигляду $v'_x=-u/2$ потрібно покласти $g=-2\cdot u$.

Заново обчисліть модель

`Home` **`-> Study`** – Compute

і отримайте той же графік розв'язків, який наведено в попередній таблиці при застосуванні вузлів `Constraint` для сьомої умови стикування.

■

Часто при формуванні моделі в двох «сусідніх» рівняннях використовується однакове ім'я шуканої величини. Зазвичай це буває, коли в гілці «фізики» на деякій ділянці розрахункової геометрії «діє» інше рівняння. Тоді наведені вище приклади не спрацьовуватимуть, бо при задаванні граничних умов стикування потрібно розрізняти імена шуканих величин на протилежних боках внутрішньої межі, а вони однакові. Та ж проблема виникає у випадку «фізики у фізиці», коли потрібно явно задавати різні значення шуканої величини по різні боки від розділювальної межі. Припустимо, що в задачі (3), (4) в обох рівняннях використовується однакове ім'я шуканої величини, наприклад, u. Тоді в точці $x=1$ можна задати лише одне значення величини u або її потоку. Воно буде однаковими для обох боків від точки, бо застосовується до спільного імені u. В результаті задаватиметься однакова гранична умова для двох відрізків (лівого і правого). В COMSOL Multiphysics є різні способи подолати цю складність (або використати цю особливість). Одним з них є застосування збірок (assembly) і пар (pairs). Для пояснення сказаного дослідимо попередню задачу (3), (4), в якій друге рівняння (4) змоделюємо як «підфізику» основної фізики рівняння (3).

Приклад 3. Розглянемо задачу (3), (4)

$$u''_{xx}=1,\ \ x\in[0,1],\ u(0)=-1, \tag{3}$$

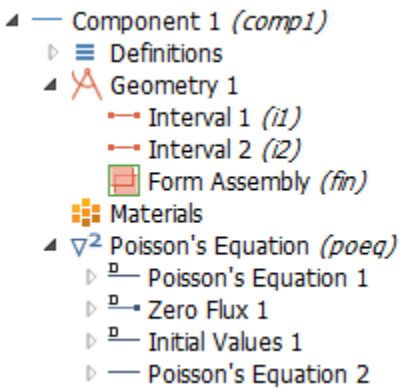
$$4u''_{xx}=x,\ \ x\in[1,2],\ u(2)=3 \tag{4}$$

з умовами стикування в точці $x=1$:

1. $u(1-0)=1,\ u(1+0)=0$.
2. $u(1-0)=2,\ u(1+0)=2$.
3. $u'_x(1-0)=4,\ u'_x(1+0)=1$.
4. $u(1-0)=u(1+0),\ u'_x(1-0)-4u'_x(1+0)=2$.
5. Неперервність поля: $u(1-0)=u(1+0),\ c_1u'_x(1-0)=c_2u'_x(1+0)$.

Тут $u(1\pm0)$ і $u'_x(1\pm0)$ позначають праву і ліву границю функцій в точці x=1.

При створенні моделей для цих задач генеруватиметься одна фізика на відрізках [0,1] і [1,2], поєднаних операцією «Form Assembly». На другому відрізку [1,2] задаватиметься рівняння (4), що автоматично звузить зону дії рівняння (3) до відрізка [0,1].



Ім'я шуканої величин в обох рівняннях буде однаковим. Створювана операцією «Form Assembly» пара, в залежності від «внутрішньої» граничної умови, використовуватиметься або ні.

Р о з в ' я з о к . За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics - Classical PDEs – Poisson's Equation (poeq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Встановіть прапорець, який визначатиме відображення індикаторів елементів геометрії.

Component 1 – Definitions - View 1:

`View` – Show geometry labels = `True`

Створіть геометрію моделі, яка складається з двох пристикованих відрізків, і з'єднайте їх в збірку (assembly).

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1) :

`Interval` – Left endpoint, Right endpoint = `0,1`

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 2 (i2) :

`Interval` – Left endpoint, Right endpoint = `1,2`

LMB > Build All Objects

Geometry 1 - Form Union:

`Form Union/Assembly` – Action = `Form an assembly`

`Form Union/Assembly` – Create pairs = `True`

Останній прапорець вказує системі створювати пари (за замовчуванням типу `Identity Pair`).

Зверніть увагу, що назва вузла **Form Union** в ДМ змінилася на **Form Assembly**. Крім того, точка x=1 відрізка [0,1] отримала індикатор з номером 2. Ця ж точка, але як частина відрізка [1,2], отримала індикатор з номером 3.

Генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – Element Size =`Extra fine`

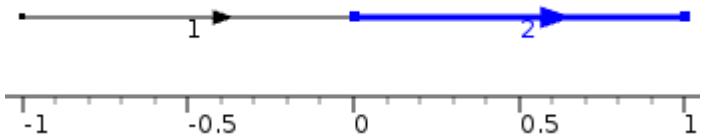
LMB> Build All

В нашій задачі є дві зони, в яких діятимуть різні рівняння. Для визначення рівняння (3) вузол **Poisson's Equation 1** вже створено. В гілці **Poisson's Equation (poeq)** для задання рівняння (4) створіть ще один такий же вузол.

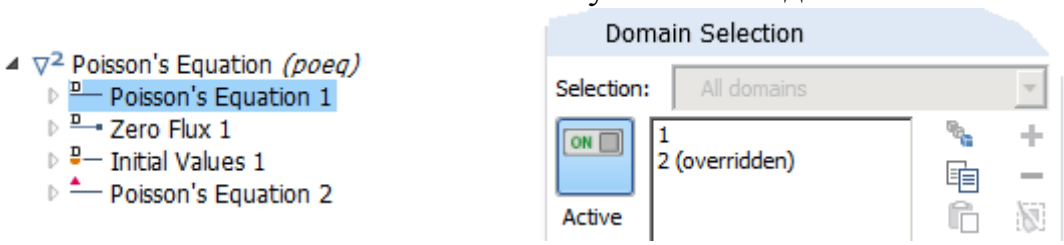
Physics-> **Domains ->** Classical PDEs - Poisson's Equation

Poisson's Equation (poeq) - Poisson's Equation 2:

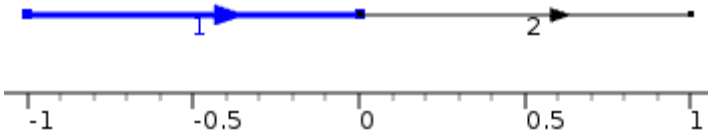
Domain Selection – **Selection,** <> = Manual, 2



Задання області дії другого рівняння автоматично звузило зону дії першого рівняння. Щоб це перевірити відкрийте ПВ вузла **Poisson's Equation 1**. Його розділ **Domain Selection** повинен мати наступний вигляд.



В секції **Domain Selection** видно, що відрізок з індикатором 2 не входить до зони дії першого рівняння, і в графічній зоні системи виділено відрізок 1.



Зверніть увагу на те, що у створеному вузлі **Poisson's Equation 2** немає можливості змінити ім'я шуканої величини, воно таке ж, як у вузла **Poisson's Equation 1**, і визначається в батьківській гілці **Poisson's Equation (poeq).**

Налаштуйте параметри вузлів **Poisson's Equation 1** і **Poisson's Equation 2**.

Poisson's Equation 1:

Diffusion Coefficient – **c** =1

Source Term – **f** = -1

Цим на відрізку [0,1] задається рівняння $-u''_{xx} = -1$.

Poisson's Equation 2:

Diffusion Coefficient – **c** = 4

Source Term – **f** = -x

Цим на відрізку [1,2] визначається рівняння $-4u''_{xx} = -x$.

Задайте граничні умови Діріхле на кінцях відрізка [0,2].

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1 (*точка x*=0)

Dirichlet Boundary Condition – **r** = -1

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection** = Manual,4 (*точка x*=2)

Dirichlet Boundary Condition – **r** = 3

Зауваження. Оскільки геометрія складається з двох окремих відрізків [0,1] і [1,2], то точка x=1, яка начебто належить обом відрізкам, «поділена» на дві частини, які мають індикатори 2 і 3. При чому, точка з індикатором 2 належить лівому відрізку, а точка з індикатором 3 – правому.

1. Внутрішні граничні умови $u(1-0)=1$, $u(1+0)=0$.

Задайте для лівої «фізики» в точці x=1 умову Діріхле $u(1)=1$, а для правої – умову $u(1)=0$.

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 3:

Boundary Selection – **Selection** = Manual,2 (*точка* x=1-0)

Dirichlet Boundary Condition – **r** = 1

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 4:

Boundary Selection – **Selection** = Manual,3 (*точка* x=1+0)

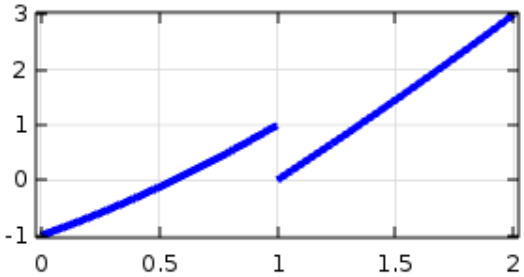
Використовуйте прокручування колеса миші для переміщення фокуса по черзі між індикаторами 2 і 3.

Dirichlet Boundary Condition – **r** = 0

Подивіться на номери індикаторів: в точці x=1 для граничної умови на відрізку [0,1] використовується індикатор 2; для умови в тій же точці на відрізку [1,2] – індикатор 3.

Виконайте обчислення.

Home -> **Study** – Compute



Звертаємо вашу увагу на те, що обидві «внутрішні» умови Діріхле створювалися командами з випадної кнопки-списку **Boundaries** – Dirichlet Boundary Condition. Якщо умову Діріхле задавати кнопкою-списком **Pairs** – Dirichlet Boundary Condition, то умова відноситиметься до «обох боків пари», тобто до точки цілком. Інакше кажучи, одна умова відноситиметься одночасно до обох точок з індикаторами 2 і 3. Пояснимо це.

2. Внутрішні граничні умови $u(1-0)=2, u(1+0)=2$.

Деактивуйте вузли останніх двох умов Діріхле і створить одну умову Діріхле на парі.

Dirichlet Boundary Condition 3: RMB > Disable

Dirichlet Boundary Condition 4: RMB > Disable

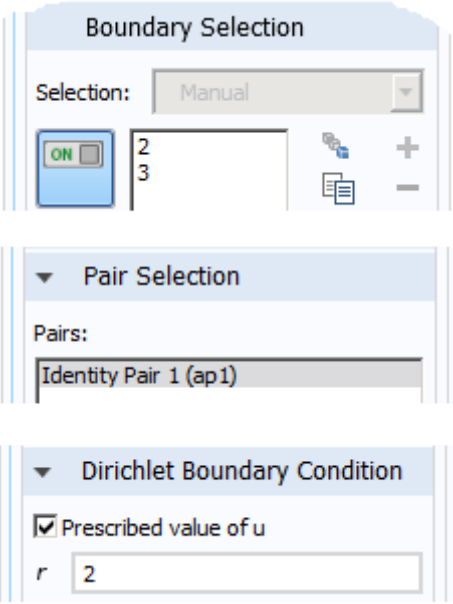
Physics -> **Boundary - Pairs** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 5:

В розділі **Pair Selection** в списку **Pairs** клацніть по рядку Identity Pair 1 (ap1).

Pair Selection – **Pairs** = Identity Pair 1(ap1)

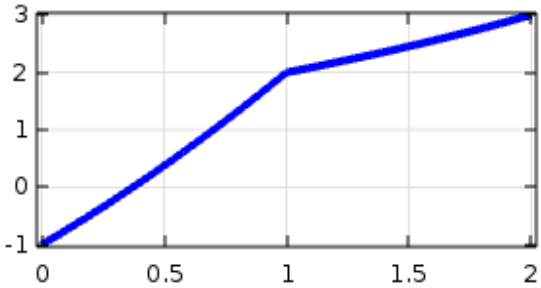
Dirichlet Boundary Condition – **r** = 2



В розділі **Pair Selection** обираються пари, які визначають місця накладання умови. Як наслідок в списку **Selection** вказані індикатори точок «робочої» пари. Нагадаємо, що кінці відрізка [0,1] мають індикатори 1 і 2, а кінці відрізка [1,2] – індикатори 3 і 4. Тобто індикатори 2 і 3 позначають «різні сторони» спільної точки цих відрізків.

Виконайте обчислення.

Home -> **Study** - Compute



Як бачите, $u(1)=2$ є граничним значенням u на обох відрізках [0,1] і [1,2].

Для перевірки самостійно добудуйте графіки аналітичного розв'язку

$$U(x)=\frac{1}{2}(-2+5\,x+x^2),\ \ x\in[0,1],$$

$$U(x)=\frac{1}{24}(30+17\,x+x^3),\ \ x\in[1,2].$$

Схожим чином можна використовувати вузли типу Flux/Source, задаючи в них різні значення потоків на сторонах пари.

3. Внутрішні граничні умови $u'_x(1-0)=4,\ u'_x(1+0)=1.$

Деактивуйте вузол останньої умови Діріхле і створить два вузли типу Flux/Source, які відноситимуться до різних «фізик».

Dirichlet Boundary Condition 5: RMB > Disable

Нагадаємо, що на правому кінці відрізка [0,1] умова `Flux/Source` має вигляд $c u'_x = g - q u$.

`Physics` -> **`Boundaries`** – Flux/Source

Flux/Source 1:

`Boundary Selection` – **Selection,** < > = `Manual,2` (т. x=1-0)

`Boundary Flux/Source` – **g** =4

Цим на відрізку [0,1] задається умова $u'_x(1) = 4$ (оскільки c=1 і q=0).

`Physics` -> **`Boundaries`** – Flux/Source

Flux/Source 2:

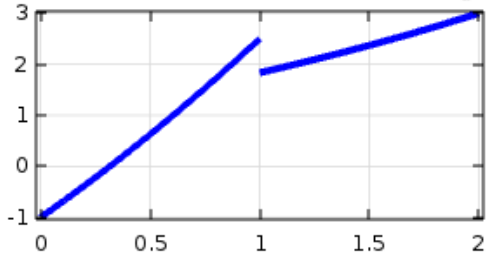
`Boundary Selection` – **Selection,** < > = `Manual,3` (т. x=1+0)

`Boundary Flux/Source` – **g** =-4

Цим на відрізку [1,2] задається умова $u'_x(1) = 1$ (нагадаємо, що при q=0 на лівому кінці умова має вигляд $-c u'_x = g$, оскільки c=4 ми поклали g=−4).

Виконайте обчислення.

`Home` -> **`Study`** - Compute



Для порівняння самостійно добудуйте графік аналітичного розв'язку

$$U(x) = \frac{1}{2}(-2 + 6\,x + x^2),\ \ x \in [0,1],$$

$$U(x) = \frac{1}{24}(22 + 21\,x + x^3),\ \ x \in [1,2].$$

Тепер подивимось, як працюватиме умова `Flux/Source`, створювана на парі.

4. Умова Flux/Source на парі. Деактивуйте два останніх вузла Flux/Source і замість них створіть один на парі.

Flux/Source 1: RMB > Disable

Flux/Source 2: RMB > Disable

`Physics` -> **`Boundary - Pairs`** – Flux/Source

Flux/Source 3:

`Pair Selection` – **Pairs** = `Identity Pair 1(ap1)`

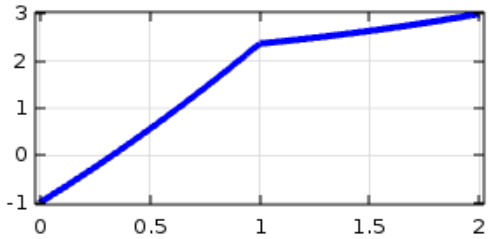
`Boundary Flux/Source` – **g** =1

Умови типу Flux/Source, які задані в спільній точці, сумуються, тобто сумуються ліві і праві частини умов $-\boldsymbol{n}\cdot(-c\nabla u) = g - q u$ цих вузлів. В нашому випадку умова Flux/Source на парі діятиме як дві окремі умови в точках з індикаторами 2 і 3. Якщо позначити індексами 1 і 2 розв'язки ліворуч і праворуч від точки стику, то при q=0 матимемо $[-\boldsymbol{n}_1\cdot(-c_1\nabla u_1)] + [-\boldsymbol{n}_2\cdot(-c_2\nabla u_2)] = g_1 + g_2$. Враховуючи напрямки нормалей і те, що $g_1 = g_2$, отримуємо $c_1 u'_{1x}(1) - c_2 u'_{2x}(1) = 2g_1$, тобто $u'_{1x}(1) - 4u'_{2x}(1) = 2$.

Також залишається умова $u_1(1)=u_2(1)$. Таким чином, умова Flux/Source на парі в т. x=1 матиме наступний вигляд $u(1-0)=u(1+0)$, $u'_x(1-0)-4u'_x(1+0)=2$.

Виконайте обчислення і отримайте графік розв'язку.

`Home` **-> Study** - Compute



Для перевірки самостійно добудуйте графік аналітичного розв'язку

$$U(x)=\frac{1}{2}(-2+\frac{86}{15}x+x^2),\ x\in[0,1],$$

$$U(x)=\frac{1}{120}(238+41\,x+5x^3),\ x\in[1,2].$$

5. Нагадаємо, що умова неперервності фізичного поля в точці стику має вигляд $u_1=u_2$, $c_1u'_{1x}=c_2u'_{2x}$, де індекси 1 і 2 відповідають граничним значенням шуканої величини u по різні боки внутрішньої межі. При однаковому імені шуканої величини ліворуч і праворуч від розділювальної точки для задання такої умови призначений вузол `Continuity`.

Деактивуйте останній вузол **Flux/Source 3** і замість нього створіть вузол `Continuity`.

Flux/Source 3: RMB > Disable

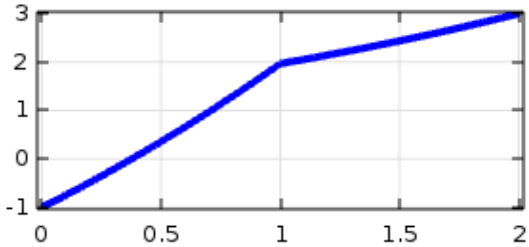
`Physics` -> **Boundary - Pairs** – Continuity

Poisson's Equation (poeq) – Continuity 1:

Pair Selection – **Pairs** = `Identity Pair 1 (ap1)`

Виконайте обчислення і отримайте графік розв'язку.

`Home` **-> Study** - Compute



Для перевірки самостійно добудуйте графіки аналітичного розв'язку

$$U(x)=\frac{1}{2}(-2+\frac{74}{15}x+x^2),\ x\in[0,1],$$

$$U(x)=\frac{1}{120}(142+89\,x+5x^3),\ x\in[1,2].$$

Знову збережіть файл, але не закривайте його.

Зауваження. Графіки розв'язків, отримані в пунктах 2 і 5, дуже схожі, але все ж таки відрізняються. Це видно і з аналітичних виразів розв'язків, наведених в цих пунктах. В п.2 умови в точці x=1 мали вигляд $u(1-0)=2$, $u(1+0)=2$. В п.5 вони були $u(1-0)=u(1+0)$, $c_1u'_x(1-0)=c_2u'_x(1+0)$. В останньому випадку $u(1-0)=u(1+0)=59/30\neq 2$.

■

Розглянемо використання обмеження `Constraint` на парі (з опцією **Individual dependent variables**=`True`). Умова `Constraint` на парі застосовується з боку входової (source) точки пари. Однаковість імен шуканих величин з обох боків внутрішньої межі призводить до додаткової умови рівності граничних значень розв'язку на цій межі.

Продовжимо розглядати задачі (3), (4), і проаналізуємо наступні обмеження `Constraint` на парі:

1. `R=2-u`.
2. `R=ux-1`.
3. `R=up(u)-down(u)+3/2`.

Приклад 4. Для перевірки роботи зазначених умов в файл попереднього прикладу будемо вносити зміни і дивитися, що виходить.

Розв'язок.

1. `R=2-u`. В останньому файлі деактивуйте вузол **Continuity 1** і замість нього створіть вузол `Constraint`.

Continuity 1: RMB > Disable

`Physics` -> **`Boundary - Pairs`** – Constraint

Poisson's Equation (poeq) – Constraint 1:

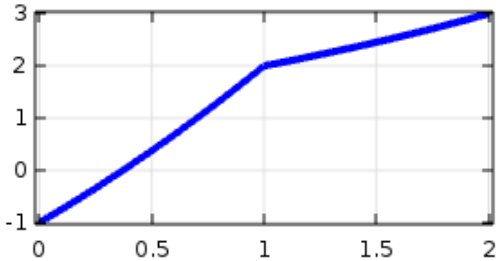
`Pair Selection` – **Pairs** = `Identity Pair 1 (ap1)`

`Constraint` – **R** = `2-u`

Нагадаємо, що входовою (source) точкою пари є точка з індикатором 2. Вираз, записаний в полі **R**, у входовій точці пари обертатиметься на нуль, тобто ми матимемо умову Діріхле $u(1-0)=2$ на «лівій» точці пари. Однаковість імен шуканих величин для «правої» точки пари призводитиме до умови $u(1+0)=u(1-0)$.

Виконайте обчислення.

`Home` -> **`Study`** - Compute



Фактично розглянуті умови використано в п.2 попереднього прикладу, тому і графіки повністю співпадають.

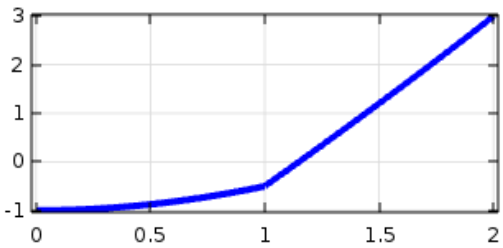
2. `R=1-ux`. Внесіть зміни у вираз, який використовується у вузлі **Constraint 1**.

Poisson's Equation (poeq) – Constraint 1:

`Constraint` – **R** = `1-ux`

`Constraint Settings` – **Apply reaction terms on** = `Individual dependent variables`

`Home` -> **`Study`** - Compute



Умова `R=1-ux=0` у вузлі **Constraint 1** реалізується як $u'_x(1-0)=1$ (на лівій точці пари) і $u(1+0)=u(1-0)$ – на правій (це випливає з однаковості імен розв'язків з обох боків пари). Аналітичний розв'язок для цієї задачі має вигляд

$$U(x) = \frac{1}{2}(-2 + x^2),\ x \in [0,1],$$

$$U(x) = \frac{1}{24}(-90 + 77\,x + x^3),\ x \in [1,2].$$

Для порівняння самостійно добудуйте ці криві в графічній групі.

Спробуйте змінити входову (source) точку пари. Для цього в ПВ вузла **Identity Pair 1 (ap1)** потрібно встановити прапорець **Manual control of selections** і в секціях вибору поміняти місцями індикатори точок пари.

Definitions - Identity Pair 1 (ap1):

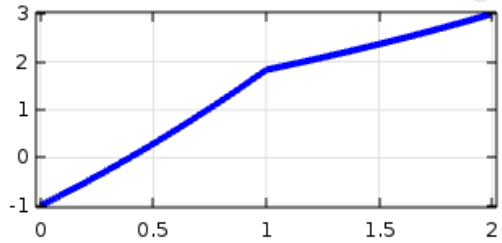
`Pair type` – **Manual control of selections** = `True`

`Source Boundaries` – **Selection,<>** = `Manual,3`

`Destination Boundaries` – **Selection,<>** = `Manual,2`

Використовуйте прокручування колеса миші для переміщення фокуса по черзі між індикаторами 2 і 3.

Знову виконайте обчислення і отримайте наступний графік.



Тепер умова `R=1-ux=0` у вузлі **Constraint 1** реалізується як $u'_x(1+0)=1$ (на правій точці пари) і $u(1-0)=u(1+0)$ – на лівій. Аналітичний розв'язок в цьому випадку має вигляд

$$U(x) = -1 + \frac{7}{3}x + \frac{1}{2}x^2,\ x \in [0,1],$$

$$U(x) = \frac{1}{24}(22 + 21\,x + x^3),\ x \in [1,2].$$

Самостійно добудуйте ці криві в графічній групі.

3. `R=up(u)-down(u)+3/2`.

Поверніть «напрямок дії пари».

Definitions - Identity Pair 1 (ap1):

`Pair type` – **Manual control of selections** = `False`

Внесіть зміни у вираз, який використовується у вузлі **Constraint 1**.

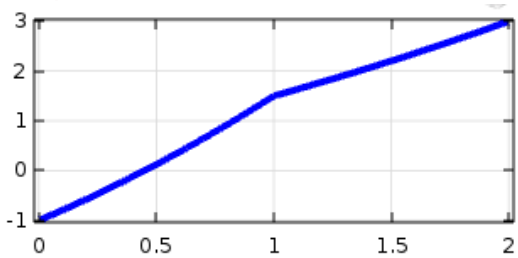
Poisson's Equation (poeq) – Constraint 1:

`Constraint` – **R** = `up(u)-down(u)+3/2`

Умова працює на парі з входовою точкою 2. Вона належить відрізку [0,1] і в ній вектор нормалі направлений зовні цього відрізка. Величина `u`, яка обчислюється в цій точці, визначена лише на цьому відрізку. Тому оператор `up(u)` повертатиме нуль, а $\mathrm{down}(u) \equiv u$. В результаті для лівого рівняння застосовуватиметься умова $u(1-0) = 3/2$. Для правого рівняння автоматично застосовуватиметься умова $u(1+0) = u(1-0)$.

Виконайте обчислення і отримайте наступний графік.

`Home` **-> Study** - Compute



Аналітичний розв'язок в цьому випадку має вигляд

$$U(x) = \frac{1}{2}(-2 + 4\,x + x^2),\ \ x \in [0,1],$$

$$U(x) = \frac{1}{24}(6 + 29\,x + x^3),\ \ x \in [1,2].$$

Самостійно добудуйте ці криві в графічній групі.

5.2. Приклади крайових задач на відрізку.

5.2.1. Крайова задача $u'' + (1+x^2)\,u = -1,\ u(-1) = 0,\ u(1) = 0$.

На відрізку [-1,1] розв'яжіть крайову задачу

$$u''_{xx} + (1+x^2)u = -1,\ -1 \le x \le 1,\ u(-1) = 0,\ u(1) = 0.$$

Р о з в ' я з о к . За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics - PDE Interfaces - Coefficient Form PDE (c) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): **`Unit System`** – <> = `None`

Геометрія моделі складається з відрізка [-1,1]. Створіть його.

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1):

`Interval` – **Left endpoint, Right endpoint** = `-1, 1`

LMB > Build All Objects

На створеному відрізку генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Задайте рівняння. В ПВ вузла **Coefficient Form PDE (c) - Coefficient Form PDE 1** в секції вибору **`Domain Selection`** повинна стояти одиниця (індикатор відрізка [-1,1]). Коефіцієнти рівняння задаються в полях нижньої частині цієї ПВ.

Coefficient Form PDE (c) - Coefficient Form PDE 1:

`Diffusion Coefficient` – **c** = `-1`

Absorption Coefficient – **a** = 1+x^2

Source Term – **f** = -1

Решту коефіцієнти покладіть нулями: e_a=0, d_a=0, α=0, β=0, γ=0.. Цим визначається рівняння $u''_{xx} + (1+x^2)u = -1$.

Задайте граничні умови.

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

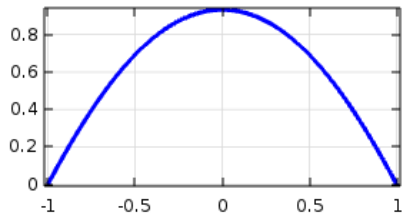
Boundary Selection – **Selection** = All boundaries

Dirichlet Boundary Condition – **r** = 0

Виконайте обчислення моделі.

Home **-> Study** - Compute

В графічній зоні вузла **Results - 1D Plot Group 1 – Line Graph 1** отримайте графік розв'язку.



Перевіримо, що праву частину рівняння можна задавати нульовою, але в фізичній моделі використовувати додатковий Source вузол, який буде зображати джерело одиничної потужності, що діє на всьому відрізку.

Внесіть зміни в коефіцієнти рівняння.

Coefficient Form PDE (c) - Coefficient Form PDE 1:

Source Term – **f** = 0

Додайте в «фізику» вузол типу Source.

Physics **-> Domain - Domains** – Coefficient Form PDE – Source

Coefficient Form PDE (c) - Coefficient Form PDE 1 – Source 1:

Domain Selection – **Selection** = Manual,1 (*відрізок* [-1,1])

Source Term – **f** = -1

Виконайте обчислення.

Home **-> Study** - Compute

Ви повинні отримати той же графік розв'язку, що і раніше.

5.2.2. Крайова задача з кусковим вільним членом.

Розв'яжіть крайову задачу

$$u'' = -f(x),\ 0 \le x \le 1,\ u(0) = 0,\ u(1) = 0, \qquad (1)$$

де кусково - стала функція $f(x)$ має вигляд

$$f(x) = \begin{cases} 0,\ x \le 0.25 \\ 1,\ 0.25 < x < 0.75 \\ 0,\ x \ge 0.75 \end{cases}.$$

Р о з в ’ я з о к 1 . За допомогою майстра створіть одновимірну модель задачі.

File->New -> Model Wizard -> 1D -> Mathematics -
PDE Interfaces - Coefficient Form PDE (c) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Створіть кускову функцію, яка буде представляти праву частину рівняння.

Component 1(comp1):

Definitions -> **Functions** -> Piecewise

Function name = pw1

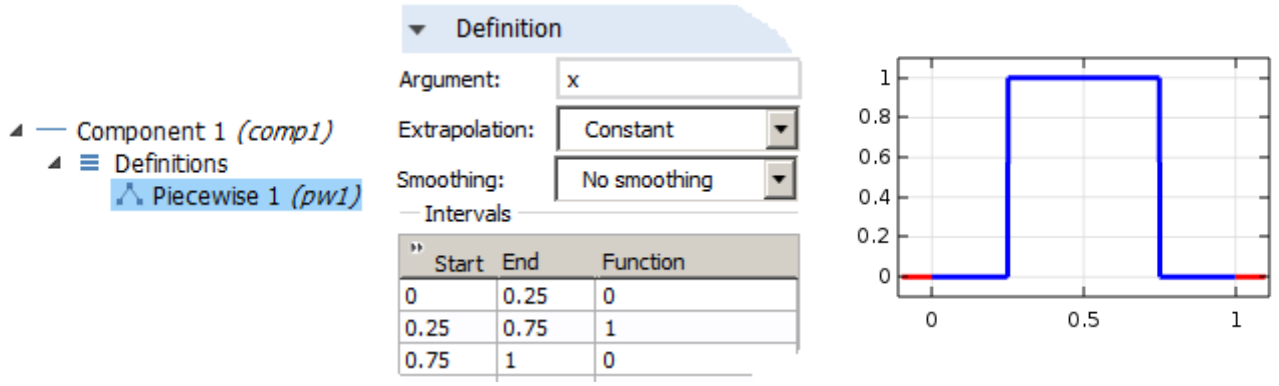
Definition – **Extrapolation** = Constant

Definition – **Intervals (таблиця)** -

– **Start, End, Function** = 0,0.25,0

– **Start, End, Function** = 0.25,0.75,1

– **Start, End, Function** = 0.75,1, 0



Вибір **Extrapolation**=Constant означає, що поза заданих ділянок функція продовжуватиметься своїми значеннями, які вона набуває на лівому і правому краях області визначення. Клацання по кнопці Plot у верхній частині вікна будує графік функції (для контролю).

Геометрія моделі складається з відрізка [0,1].

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1) : LMB > Build All Objects

На створеному відрізку генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Перевірте , що в ПВ вузла **Coefficient Form PDE (c) - Coefficient Form PDE 1** в секції вибору **Domain Selection** стоїть одиниця (індикатор відрізка). Коефіцієнти рівняння задайте в полях в нижній частині цієї ПВ.

Coefficient Form PDE (c) - Coefficient Form PDE 1:

Diffusion Coefficient – **c** = -1

Absorption Coefficient – **a** = 0

Source Term – **f** =-pw1 (x)

Решту коефіцієнти покладіть нулями: e_a=0, d_a=0, α=0, β=0, γ=0. Цим визначається рівняння $u''_{xx} = -\mathrm{pw1}(x)$.

Задайте граничні умови.

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

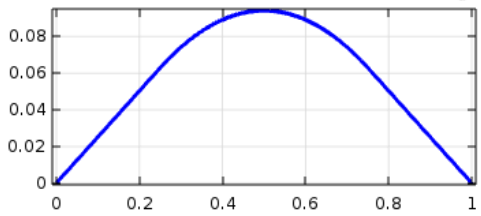
Boundary Selection – **Selection** = All boundaries

Dirichlet Boundary Condition – **r** = 0

Виконайте обчислення.

Home **-> Study** - Compute

В графічній зоні вузла **Results - 1D Plot Group 1 – Line Graph 1** отримайте графік розв'язку.



Розв'язок 2. У рівняннях типу (1) член f в правій частині зазвичай представляє зовнішній вплив (зовнішню силу, джерело тепла, джерело струму і т.д.). Його можна коригувати виразами, які додаються в рівняння за допомогою вузлів-джерел типу Source.

Продовжить роботу з попереднім файлом. Відкрийте ПВ вузла **Coefficient Form PDE (c) - Coefficient Form PDE 1** і в поле, що задає член ***f***, введіть 0 (замість -pw1(x)).

Coefficient Form PDE (c) - Coefficient Form PDE 1:

Source Term – **f** = 0

Активуйте вузол **Geometry 1** і створіть на відрізку [0,1] дві точки x=0.25 і x=0.75.

Geometry **-> Primitives->** Point

Geometry 1 – Point 1:

Point – **x** =0.25, 0.75

LMB > Build All Objects

Точки потрібні для того, щоб далі ви могли вказати інтервал [0.25,0.75], на якому буде діяти зовнішня сила.

В ПВ вузла **Dirichlet Boundary Condition 1** перевірте, що граничні умови не змінилися. Якщо в секції вибору ПВ цього вузла з'явилися нові точки, то видаліть їх номери зі списку.

Додайте в гілку **Coefficient Form PDE (c)** вузол типу Source, який представляє зовнішній вплив.

Physics **->Domain - Domains** – Coefficient Form PDE – Source

Coefficient Form PDE (c) - Coefficient Form PDE 1 – Source 1:

Domain Selection – **Selection** = Manual,2 (*відрізок* [0.25,0.75])

На відрізку [0.25,0.75] буде діяти зовнішня сила.

Source Term – **f** = -1

Виконайте обчислення моделі.

Home **-> Study** - Compute

В графічній зоні вузла **1D Plot Group 1 – Line Graph 1** ви повинні отримати ту ж криву, що і раніше.

Розв'язок 3 (для тренування на використання пар/pairs). При розв'язанні деяких задач тіло потрібно розглядати не як суцільне, а як збірку, складену з

декількох деталей, матеріал кожної з яких має власні фізичні властивості. В такому випадку в останньому вузлі гілки Geometry вказується Assembly тип з'єднання елементів. Тоді система інтерпретує геометрію як колекцію фігур (замість їх об'єднання), і користувачеві самому потрібно визначити способи зв’язку розрізнених тіл за допомогою інструментів, які називаються «парами» (pairs). Елементи пари позначають протилежні боки межі контактуючих тіл. За замовчуванням пари створюються автоматично при формуванні збірки.

Створіть одновимірну модель з двома «фізиками»: рівняннями Лапласа і Пуассона.

File->New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs – Laplace Equation (lpeq) ->
Classical PDEs – Poisson's Equation (poeq) + Variables = u -> Stationary

При створенні моделі ми відразу додали дві фізики/рівняння, і в обох задали однакове ім'я u шуканої змінної. Це не приводитиме до конфлікту, оскільки рівняння діятимуть на різних відрізках.

Untitled.mph (root): Unit System – <> = None

Встановіть прапорець відображення індикаторів елементів геометрії.

Component 1 – Definitions - View 1:

View – **Show geometry labels** = True

Створіть розрахункову область моделі, яка складається з трьох пристикованих відрізків, і з'єднайте їх в збірку (assembly).

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1) :

Interval – **Left endpoint, Right endpoint** = 0,0.25

Geometry->**Primitives->** Interval

Geometry 1 - Interval 2 (i2) :

Interval – **Left endpoint, Right endpoint** = 0.25,0.75

Geometry->**Primitives->** Interval

Geometry 1 - Interval 3 (i3) :

Interval – **Left endpoint, Right endpoint** = 0.75,1

LMB > Build All Objects

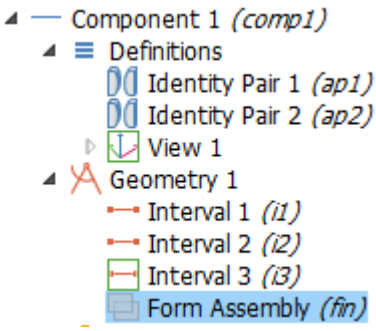
Geometry 1 - Form Union:

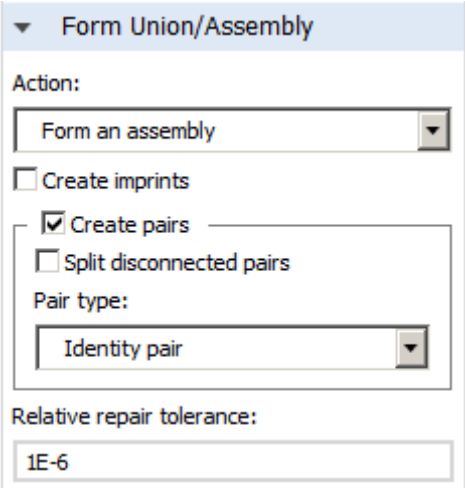
Form Union/Assembly – **Action** = Form an assembly

Остання інструкція змінить назву вузла на **Form Assembly (fin)**.

Form Union/Assembly – **Create pairs** = True

Ця інструкція вказує системі створювати пари (за замовчуванням типу Identity Pair).





Генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

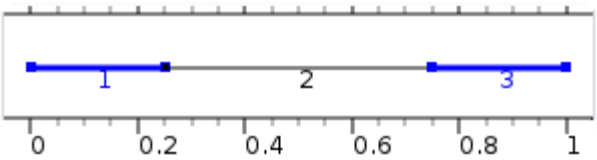
Mesh Settings – **Element Size** =Finer

LMB> Build All

Задайте геометрію, на якій діятиме рівняння Лапласа.

Component 1 - Laplace Equation (lpeq):

Domain Selection – **Selection**= Manual,1,3 (*відрізки* [0,0.25] і [0.75,1])

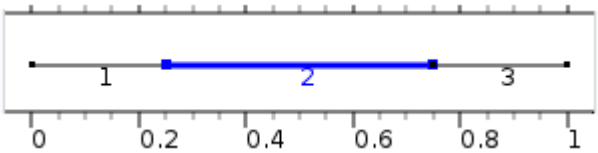


Коефіцієнти рівняння Лапласа задавати непотрібно.

Задайте геометрію, на якій діятиме рівняння Пуассона, і його параметри.

Poisson's Equation (poeq):

Domain Selection – **Selection**=Manual,2 (*відрізок* [0.25,0.75])



Poisson's Equation(poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = 1

Source Term – **f** = 1

Зауваження. Рівняння Пуассона не зміниться, якщо задати коефіцієнти c=-1, f=-1. Але в подальшому ми задаватимемо неперервність поля в точках x=0.25 і x=0.75, для якої коефіцієнти рівняння Пуассона повинні бути узгоджені з рівнянням Лапласа.

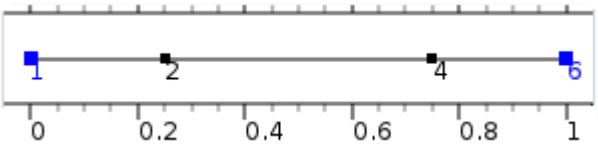
В гілці **Laplace Equation (lpeq)** створіть вузол нульових граничних умов.

Laplace Equation (lpeq):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Laplace Equation (lpeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1,6 (*лівий і правий кінці*)



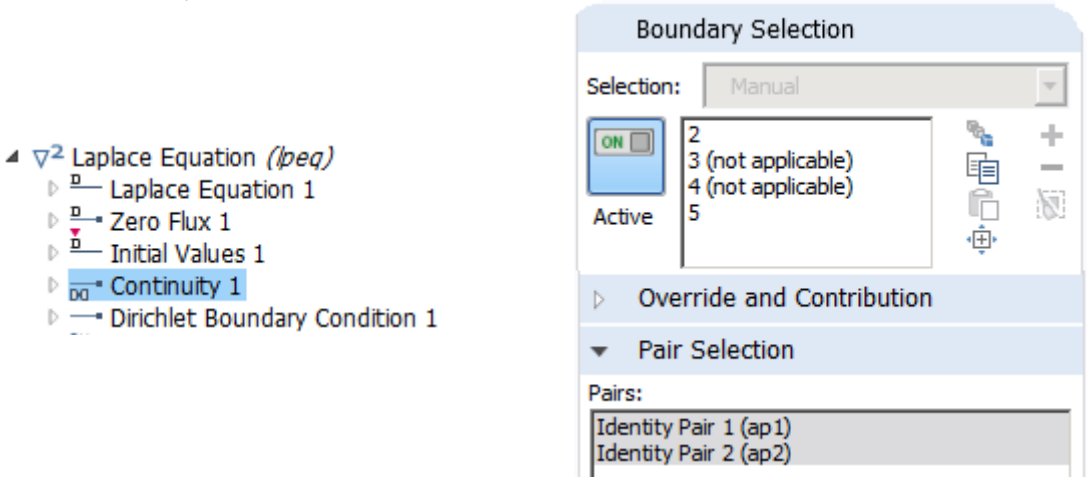
Dirichlet Boundary Condition – **r** = 0

Тепер додайте умови неперервності розв'язків у вузлах стикування розрахункових геометрій двох «фізик». Умова неперервності означає неперервність шуканої величини і потоку в зоні стикування. Математично це означає неперервність в точках стикування розв'язків рівняння Лапласа u_L і Пуассона u_P, ($u_L = u_p$ в точках з'єднання) і величин, пов'язаних з їх похідними.

В нашому прикладі $-1 \cdot \frac{\partial u_L}{\partial x} = -c \frac{\partial u_P}{\partial x}$ в точках x=0.25 і x=0.75. Умови неперервності потрібно створювати в обох фізиках.

Laplace Equation (lpeq): RMB > Pairs – Continuity

В ПВ створеного вузла **Continuity 1** в розділі **Pair Selection** в списку **Pairs** виділіть пари Identity Pair 1 (ap1) і Identity Pair 2 (ap2). Для цього утримуючи клавішу CTRL, клацніть мишею по цим елементам. Вони представляють вузли – кінці з'єднаних відрізків.



Тут в списку **Selection** стоять індикатори точок використовуваних пар зі сторін розглядуваної розрахункової області. В розділі **Pair Selection** вказуються пари, які визначають місця накладання умови неперервності. Обрати або зняти вибір можна комбінацією Ctrl-click.

Аналогічно попередньому в гілці фізики **Poisson's Equation (poeq)** створіть вузол Continuity.

Poisson's Equation (poeq): RMB > Pairs – Continuity

Poisson's Equation (poeq) – Continuity 1:

Boundary Selection – **Selection** = Manual,3,4 (*виділити обидві пари*)

Виконайте обчислення.

Home **-> Study** - Compute

Створюються дві гілки з однаковими графіками (кожна «фізика» побудувала власний графік розв'язку u). Вилучіть другу гілку.

Results - 1D Plot Group 2: RMB > Delete

Для порівняння в першій групі створіть графік «аналітичного» розв'язку

$$u(x)=\begin{cases} x/4 & , \quad x \le 0.25 \\ -x^2/2+x/2-1/32 & , 0.25 < x < 0.75 \\ (1-x)/4 & , \quad x \ge 0.75 \end{cases}.$$

Для цього спочатку в моделі створіть кускову функцію, яка обчислюється за останньою формулою.

Component 1(comp1):

`Definitions` -> **`Functions`** -> Piecewise

Function name = `pw1`

`Definition` – **Intervals (таблиця)** -

– **Start, End, Function** = `0,0.25,x/4`

– **Start, End, Function** = `0.25,0.75, -x^2/2+x/2-1/32`

– **Start, End, Function** = `0.75,1, (1-x)/4`

Щоб функція стала доступною, виконайте оновлення розв'язку.

Study 1: RMB > Update Solution

В гілку **1D Plot Group 1** додайте графік функції `pw1`.

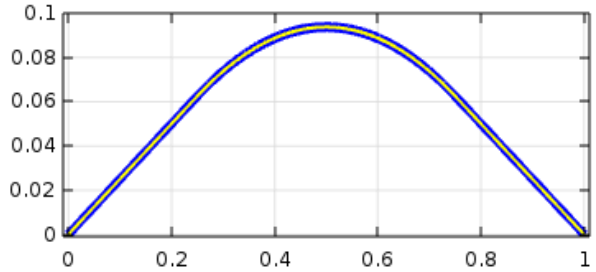
Results - 1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 2 :

`y-Axis Data` – **Expression** = `pw1(x)`

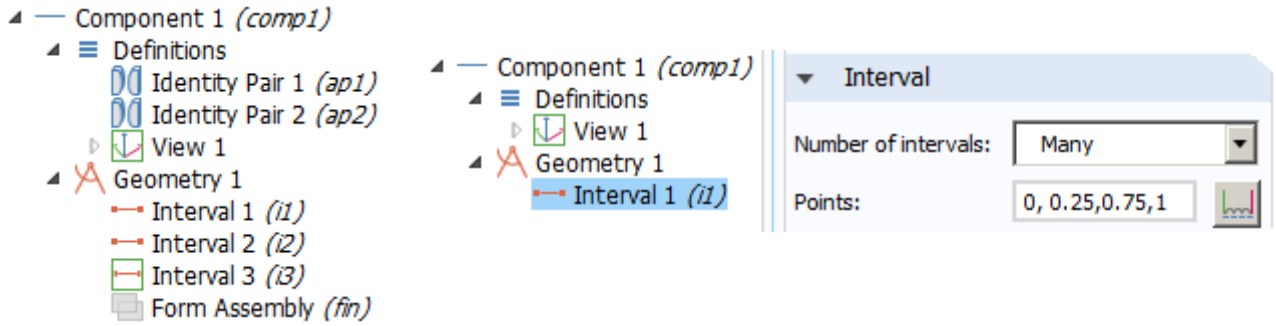
Стиль кривої налаштуйте так, щоб вона відрізнялася від графіка величини *u*.

LMB > Plot



Як бачите, графіки співпадають.

Зауваження. Щоб пари типу `Identity Pair` створювалися системою, в гілці Geometry потрібно побудувати три інтервали так, як зображено на наступному рисунку ліворуч. Креслення кількох відрізків за допомогою одного вузла **Interval 1** з точками всередині так, як зображено на наступному рисунку праворуч, не приведе до появи вузлів пар.



5.2.3. Статичний прогин струни під дією зосереджених сил.

Диференціальне рівняння рівноваги струни має вигляд $u''_{xx} = -f(x)$, де $f(x) = F(x)/T$, T – натяг струни, $F(x)$ – зовнішня (поперечна) сила, розрахована на одиницю довжини. Функція $u(x)$ являє вертикальне зміщення точок струни, які мали до навантаження горизонтальну координату *x*. Крайова задача, розв'язана нами в попередньому прикладі, має такий вигляд, і її можна інтерпретувати як рівняння статичного прогину струни під дією кусково-сталої зовнішньої сили. Схожий вигляд має задача рівноваги струни під дією зосереджених сил.

Нехай струна закріплена в точках $x_0 = 0$ і $x_n = L$, а зовнішні сили f_i прикладені в точках x_i $(i = 1, 2, \ldots, n-1)$, $0 = x_0 < x_1 < x_2 < \ldots < x_{n-1} < x_n = L$. Функцію $f(x)$, створювану набором зосереджених сил у внутрішніх точках x_i відрізка $[0, L]$, можна представити у вигляді $f(x) = \sum_{i=1}^{n-1} f_i \cdot \delta(x - x_i)$, де $\delta(x - x_i)$ - функція Дірака, яка дорівнює нулю всюди, окрім точки x_i, в якій вона обертається на нескінченність так, що інтеграл по відрізку, який містить точку x_i, дорівнює одиниці. Тоді задача про рівновагу струни під дією зосереджених сил набуває вигляду

$$u''_{xx} = -\sum_{i=1}^{n-1} f_i \cdot \delta(x - x_i),\ u(0) = 0,\ u(L) = 0. \tag{1}$$

У цієї задачі є «аналітичний» розв'язок у вигляді ламаної

$$u(x) = \sum_{i=1}^{n-1} f_i \cdot \left(\frac{x(L - x_i) + x_i(L - x)}{2L} - \frac{1}{2}\left| x - x_i \right| \right), \tag{2}$$

який ми використаємо для порівняння з чисельним розв'язком.

Для розв'язання задачі (1) в Comsol можна змоделювати дельта функцію Дірака за допомогою «вузької звичайної» функції, інтеграл від якої по відрізку її носія дорівнює одиниці. Наприклад, такою функцією може бути кусково - стала функція

$$\mathrm{dlt}(x, \varepsilon) = \begin{cases} 1/\varepsilon, x \in [-\varepsilon/2, \varepsilon/2] \\ 0, \quad x \notin [-\varepsilon/2, \varepsilon/2] \end{cases}$$

при малому значенні параметра ε. Та, якщо довжина її носія менше довжини скінченних елементів, то чисельний метод може «не помітити» таку «вузьку» функцію і ви отримаєте хибний результат. Для наближення дельта функції краще використовувати неперервну гладку функцію, наприклад, функцію Гауса $f(x) = e^{-x^2/2\sigma^2} / \sigma\sqrt{2\pi}$. Вона має дзвіноподібну форму і задовольняє умові $\int_{-\infty}^{\infty} f(x)dx = 1$. При невеликому значенні параметра σ функція Гауса має «вузький» графік і при малому (але не занадто) σ її можна застосовувати замість дельта-функції. Однак в COMSOL є інше рішення. Воно полягає у використанні вбудованого інструменту точкових джерел `Point Source`, а в одновимірному випадку – `Flux/Source`.

Приклад. Розв'яжемо крайову задачу (1) з наступними даними: $L=3, x_1=1, x_2=2, f_1=1, f_2=2$. Диференціальне рівняння набуває вигляду $u''_{xx}=-\delta(x-1)-2\delta(x-2)$, а «аналітичний» розв'язок (2) має вигляд $u(x)=5/2-x/6-|x-1|/2-|x-2|$.

Р о з в ' я з о к . За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics - PDE Interfaces - Coefficient Form PDE (c) -> Stationary

Untitled.mph (root): `Unit System` – <> = `None`

Побудуйте геометрію моделі, яка складається з відрізка [0,3] і двох точок x=1, x=2. Точки потрібні для того, щоб згодом ви могли вказати їх при заданні місця прикладання зосереджених сил.

`Geometry` -> **`Primitives`** -> Interval

Geometry 1 - Interval 1 (i1):

`Interval` – **Left endpoint, Right endpoint** = `0,3`

`Geometry` -> **`Primitives`** -> Point

Geometry 1 – Point 1 (pt1):

`Point` – **x** =`1,2`

LMB > Build All Objects

Побудуйте СкЕ сітку.

Component 1:

`Mesh Settings` –**Element size** = `Extremely fine`

LMB > Build Mesh

Задайте рівняння. Перевірте, що в ПВ вузла **Coefficient Form PDE (c) - Coefficient Form PDE 1** в секції вибору **`Domain Selection`** стоять номери `1,2,3` – індикатори всіх частин відрізка [0,3]. Коефіцієнти рівняння задайте в полях в нижній частині цієї ПВ.

Coefficient Form PDE (c) - Coefficient Form PDE 1:

`Diffusion Coefficient` – **c** = `-1`

`Absorption Coefficient` – **a** = `0`

`Source Term` – **f** =`0`

Решту коефіцієнти покладіть нулями: e_a=0, d_a=0, α=0, β=0, γ=0.

Ми вже казали, що член `f` являє зовнішній вплив, який можна задавати не в полі **`Source Term`** – **f**, а використовуючи спеціальні вузли джерел типу `Source` або `Flux/Source`. Зокрема, джерела можуть бути точковими.

Зауваження. В одновимірних моделях зовнішнє зосереджене зусилля задається командою `Physics` > **`Boundary - Boundaries`** – Flux/Source, а зовнішнє розподілене зусилля – командою `Physics` > **`Domain - Domains`** – Source.

Додайте в «фізику» **Coefficient Form PDE(c)** вузли типу `Flux/Source`, які будуть представляти зосереджені навантаження.

`Physics` -> **`Boundary - Boundaries`** – Flux/Source

Coefficient Form PDE (c) - Flux/Source 1:

`Boundary Selection` – **Selection,** < > = `Manual,2` (*точка* x=1)

Boundary Flux/Source – **g** =-1

Зауваження. Тут зосереджена сила додатна і діє вгору.

Додайте другу зосереджену силу.

Physics -> **Boundary - Boundaries** – Flux/Source

Boundary Selection – **Selection,** < > = Manual,3 (*точка* x=2)

Boundary Flux/Source – **g** =-2 (*величина зосередженої сили*)

Зауваження. У випадку Coefficient Form PDE інтерфейсу вузол Flux/Source в кінцевих точках розрахункової області (відрізка) реалізує умову $-\boldsymbol{n}\cdot(-c\nabla u-\alpha u+\gamma)=g-qu$. При обраних значеннях параметрів, $\alpha=0,\gamma=0,q=0$ вона має вид $\boldsymbol{n}\cdot c\nabla u=g$, де $\boldsymbol{n}=\pm\boldsymbol{i},\nabla u=\boldsymbol{i}\,u'_x$ (тут $\boldsymbol{i}$ – одиничний вектор вздовж осі X). Отже умова набуває вигляду $\pm u'_x=g/c$, який фактично задає значення похідної в кінцевих точках відрізка. Але для внутрішніх точок відрізка сенс умови полягає в тому, що вона задає стрибок похідної при переході через точку (а не значення похідної). Тоді умову можна інтерпретувати як точкове джерело (зовнішньої сили, тепла та ін.). Дійсно, з (1) після інтегрування випливає $u'_x=-\sum_{i=1}^{n-1}f_i\cdot H(x-x_i)+C_1$, де $H(x)$ функція Хевисайда. Якщо розглянути околицю точки $x=x_k$, то ліворуч від неї матимемо $u'_x(x_k-0)=-\sum_{i=1}^{k-1}f_i+C_1$, а праворуч $u'_x(x_k+0)_1=-\sum_{i=1}^{k}f_i+C_1$. Тоді стрибок похідної в точці x_k дорівнюватиме $u'_x(x_k+0)-u'_x(x_k-0)=-f_k$. На зовнішній межі одна з цих однобічних похідних дорівнює нулю, і тому вузол Flux/Source на ній задає величину пропорційну похідній (з множником c).

Задайте граничні умови.

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1,4 (*кінці відрізка* [0,3])

Dirichlet Boundary Condition – **r** = 0

Виконайте обчислення.

Home -> **Study** - Compute

В графічній зоні вузла **1D Plot Group 1 – Line Graph 1** ви отримаєте графік розв'язку.

Для порівняння побудуйте графік «точного» розв'язку. Для цього в графічну групу **1D Plot Group 1** додайте ще один лінійний графік.

1D Plot Group 1 -> **Add Plot** – Line Graph

1D Plot Group 1 - Line Graph 2:

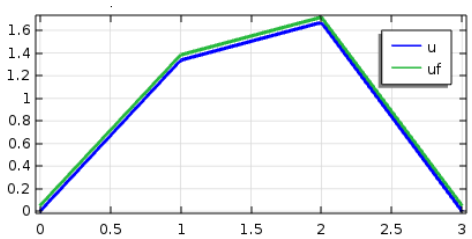
Selection – < > = 1,2,3 (*індикатори частин відрізка* [0,3])

y-Axis Data – **Expression** = 5/2-x/6-abs(x-1)/2-abs(x-2)+0.05

x-Axis Data – **Parameter, Expression** = Expression,x

LMB > Plot

В полі **Expression** доданок 0.05 використовується для того, щоб графіки наближеного і точного розв'язків не зливалися.



5.2.4. Крайова задача $((1+u)u')'=0, u(0)=0, u(1)=1$.

Розв'яжемо крайову задачу

$$\frac{d}{dx}\left((1+u)\frac{du}{dx}\right)=0,\quad u(0)=0, u(1)=1.$$

Вона є математичною моделлю задачі стаціонарної теплопровідності в одновимірному теплоізольованому з боків стрижні, кінці якого підтримуються при температурах 0 і 1, і з коефіцієнтом теплопровідності k, що залежить від температури u за формулою $k=1+u$.

Для тренування покажемо, як використовується загальна (генеральна) форма рівнянь Comsol Multiphysics.

Р о з в ' я з о к . За допомогою майстра створіть безрозмірну 1D модель задачі

File -> New -> Model Wizard -> 1D -> Mathematics -
PDE Interfaces - General Form PDE (g) -> Stationary

Untitled.mph (root): `Unit System` – <> = None

Геометрія моделі складається з відрізка [0,1].

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1) : LMB > Build All Objects

На створеному відрізку генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Задайте рівняння. Перевірте , що в ПВ вузла **General Form PDE (g) - General Form PDE 1** в секції вибору **`Domain Selection`** стоїть одиниця.

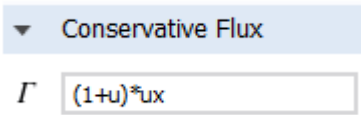
В розділі **`Equation`** наведено вид рівняння в загальній (генеральній) формі

$$e_a\frac{\partial^2 u}{\partial t^2}+d_a\frac{\partial u}{\partial t}+\nabla\cdot\Gamma=f\ .$$

Параметри рівняння задаються в полях нижньої частини цієї ПВ.

General Form PDE (g) - General Form PDE 1:

`Conservative Flux` – **Γ** = `(1+u)*ux`



`Source Term` – **f** = 0

`Damping or Mass Coefficient` – $\mathbf{d_a}$ = 0

`Mass Coefficient` – $\mathbf{e_a}$ = 0

Цим визначається рівняння $\frac{d}{dx}\left((1+u)\frac{du}{dx}\right)=0$.

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual

Boundary Selection – < > = 1 (*лівий кінець відрізка*)

Dirichlet Boundary Condition – **r** = 0

Physics -> **Boundaries** – Dirichlet Boundary Condition

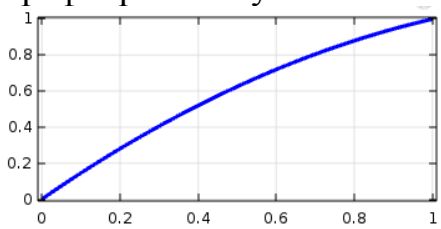
Dirichlet Boundary Condition 2:

Boundary Selection – **Selection** = Manual

Boundary Selection – < > = 2 (*правий кінець відрізка*)

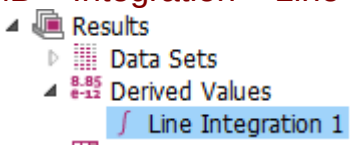
Dirichlet Boundary Condition – **r** = 1

Виконайте команду **Compute** і в графічній зоні вузла **Results - 1D Plot Group 1 – Line Graph 1** отримайте графік розв'язку.



Обчисліть середню температуру стрижня.

Results – Derived Values: RMB > Integration – Line Integration



Line Integration 1:

Selection – < > = 1 (*відрізок* [0,1])

Цим визначається відрізок інтегрування. Далі введіть вираз, який буде інтегруватися.

Expression – **Expression** = u

Натисніть кнопку Evaluate і в інформаційній зоні ви побачите значення інтеграла $\int_0^1 u\,d\,x$.

Dependent variable u
0.55556

Він представляє середнє значення температури, яке обчислюється за формулою $u_{cp} = \int_0^1 u\,d\,x \Big/ \int_0^1 d\,x$. Однак середнє значення можна обчислювати аналогічно обчисленню інтеграла.

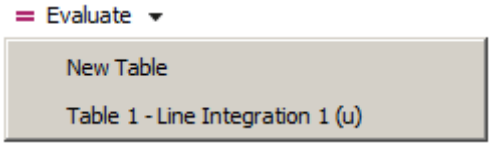
Results – Derived Values: RMB > Average – Line Average

Line Average 1:

Selection – < > = 1 (*відрізок* [0,1])

Expression – **Expression** = u

На кнопці-списку Evaluate виберіть Table 1 – Line Integration 1(u), тобто виконайте друк значення в попередню таблицю.



Ви отримаєте середнє значення `0.55556`, яке збігається зі значенням інтеграла, отриманим раніше.

Dependent variable u	Dependent variable u
0.55556	0.55556

Розглянуту крайову задачу можна інтерпретувати як задачу стаціонарної теплопровідності в плоскій стінці завтовшки одиниця із залежним від температури коефіцієнтом теплопровідності. Розв'язання математичної задачі від фізичної інтерпретації не залежить, однак графічне зображення розв'язку може бути наочнішим, якщо побудувати залитий контурний графік температури по товщині стінки. Покажемо, як це можна зробити.

Щоб побудувати поверхневий графік потрібно мати двовимірне джерело даних. Один із способів полягає в створенні в моделі компонента з двовимірною геометрією, на якій згодом буде побудований графік. Але, оскільки двовимірна фігура матиме прямокутну форму (зображатиме частину перерізу стінки), то простіше використати джерело даних типу `Grid 2D` (тобто компонент можна не створювати).

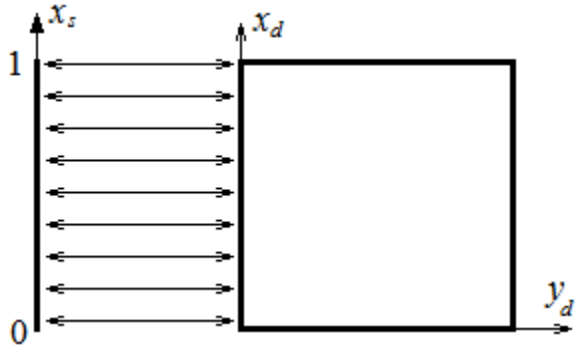
Results – Data Sets : RMB > More Data Sets - Grid 2D

Data Sets - Grid 2D 1:

`Data` – Source = `Data set`

`Data` – Data set = `Study 1/Solution 1`

За замовчуванням ДД `Grid 2D` представлятиме квадрат $[0,1]\times[0,1]$. Але двовимірного ДД замало. Справа в тому, що абсциса в ДД `Grid 2D` не пов'язана з поздовжньою координатою відрізка [0,1], який є розрахунковою областю нашої моделі. Для побудови над ДД `Grid 2D` контурного графіка розв'язку `u` нам знадобиться оператор відповідності (coupling operator). Він буде встановлювати відображення точок x_s відрізка-джерела [0,1] і точок $[x_d, y_d]$ квадрата-адресата за формуло $x_d = x_s$.



Оператори відповідності створюються у вузлах `Definitions` компонентів-джерел, і можуть використовуватися на всіх етапах роботи з моделями.

Для створення такого оператора виконайте інструкції.

Component 1:

`Definitions->`**`Coupling-Component Coupling`-General Extrusion**

Definitions - General Extrusion 1 (genext1):

Source Selection – **Selection – Manual,** < > = Manual,1 (*відрізок* [0,1])

Destination Map – **x-expression** = x

Тут в полі **x-expression** параметр x є абсцисою в розрахунковій області адресата (будь-якій геометрії, в якій існує координата x; в нашому випадку це буде абсциса точок ДД Grid 2D). В полі **Opertor name** залиште ім'я genext1 без змін. В результаті цей оператор задаватиме відповідність абсцис точок відрізка і абсцис точок ДД Grid 2D. При цьому використання оператора у вигляді genext1(u) означатиме обчислення в точці (x,y) адресата виразу u(x).

Оператор відповідності можна використовувати в будь-яких місцях моделі. Для конкретності перед його ім'ям слід писати префікс comp1 – ім'я компонента, в якому оператор створений, і з якого вибирається величина u (тобто застосовувати запис comp1.genext1(u)).

Переобчисліть модель.

Home -> Compute

Для побудови кольорового графіка температури створіть двовимірну графічну групу і в ній Surface графік.

Results -> **Plot Group** – 2D Plot Group

2D Plot Group 2:

Data – **Data set** = Grid 2D 1

2D Plot Group 2 -> Surface

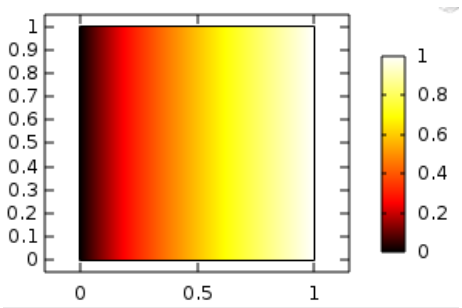
2D Plot Group 2 – Surface 1:

y-Axis Data – **Expression** = comp1.genext1(u)

Coloring and Style – **Color table** = Thermal

LMB > Plot

В прямокутнику Grid 2D буде побудований залитий контурний графік функції $U(x,y)=u(x)$, який в кольорі представлятиме розподіл температури по товщині стінки.



Для пояснення роботи General Extrusion оператора створимо ще один оператор, який буде виконувати відповідність $x_s \leftrightarrow \sqrt{\dfrac{x_d^2+y_d^2}{2}}$ між точками відрізка $x_s \in [0,1]$ і точками квадрата $(x_d, y_d) \in [0,1]\times[0,1]$.

Component 1:

Definitions->**Coupling-Component Coupling-General Extrusion**

Definitions - General Extrusion 2 (genext2):

Source Selection – **Selection – Manual,** < > = Manual,1 (*відрізок* [0,1])

Destination Map – **x-expression** = sqrt((x^2+y^2)/2)

Тут в полі **x-expression** вираз sqrt((x^2+y^2)/2) представляє відстань (з точністю до множника $\sqrt{2}$) точок квадрата до початку координат. В полі **Opertor name** залиште ім'я genext2. Використання виразу genext2(u) над двовимірним ДД Grid 2D буде означати обчислення в його точках виразу $u\left(\sqrt{(x^2+y^2)/2}\right)$.

Поновіть розв'язок.

Study 1: RMB > Update Solition

Створіть двовимірну графічну групу і в ній Surface графік.

Results -> **Plot Group** – 2D Plot Group

2D Plot Group 3:

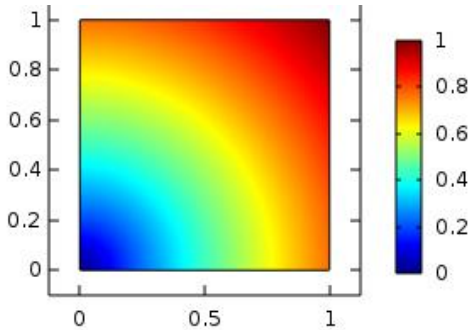
Data – **Data set** = Grid 2D 1

2D Plot Group 3 -> Surface

2D Plot Group 3 – Surface 1:

y-Axis Data – **Expression** = comp1.genext2(u)

LMB > Plot



Зауваження. При використанні General Extrusion оператора важливо, щоб значення виразу sqrt((x^2+y^2)/2) не виходило за межі геометрії джерела (відрізка [0,1]). В нашому прикладі потрібно дотримувати нерівність $0 \le \sqrt{x^2+y^2}\big/\sqrt{2} \le 1$ (що виконується).

Параметричний аналіз. Припустимо, що ми бажаємо проаналізувати розв'язок крайової задачі

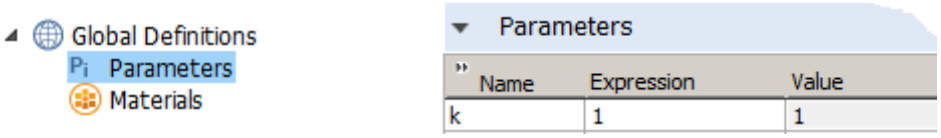
$$\frac{d}{dx}\left((1+ku)\frac{du}{dx}\right)=0,\ u(0)=0,\ u(1)=1$$

при різних значеннях параметра k. Звичайно, в моделі можна створити параметр k і використовувати його під час запису рівняння. Потім можна змінювати значення k в таблиці параметрів і для нового значення параметра кожен раз проводити обчислення. Проте розробниками системи передбачена можливість виконання одного обчислення відразу для всіх значень параметра. Це називається Parametric Sweep дослідженням (параметричним аналізом). Покажемо, як його виконати.

Продовжить роботу з поточною моделлю. Почніть з того, що в гілці **Global Definitions** згенеруйте вузол, який призначений для створення параметрів.

Global Definitions: RMB > Parameters

В ПВ створеного вузла **Parameters** в таблиці параметрів в колонці **Name** введіть ім'я параметра `k`, а в колонці **Expression** введіть якесь значення, наприклад, одиницю.



Тепер параметр потрібно використати у рівнянні. Для цього внесіть зміну в ПВ вузла **General Form PDE 1**.

General Form PDE (g) - General Form PDE 1:

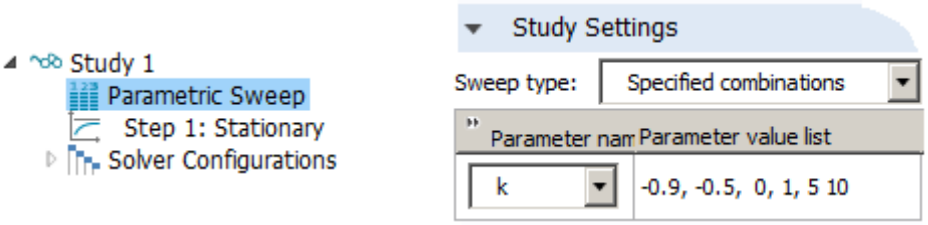
`Conservative Flux` – **Γ** = `(1+k*u)*ux`

Решту полів цієї ПВ залиште незмінними.

Для визначення залежності розв'язку рівняння від параметра в гілці **Study 1** потрібно створити відповідний вузол.

Study 1: RMB > Parametric Sweep

В ПВ створеного вузла **Parametric Sweep** в розділі **`Study Settings`** є таблиця для введення значень параметрів. Клацніть по лівій колонці таблиці правою кнопкою миші і виберіть команду Add. Із списку виберіть, створений нами раніше параметр `k`. У другому стовпці таблиці задайте бажані значення параметра, наприклад, `-0.9,-0.5,0,1,5,10`. У списку **Sweep type** залиште `Specified combinations` (ця опція використовується, коли аналіз виконується за кількома параметрами).



Виконайте команду Compute і в графічній зоні вузла **Results - 1D Plot Group 1 - Line Graph 1** отримайте графіки розв'язків, які відповідають різним значенням параметра *k*. Для відображення легенди виконайте наступні інструкції.

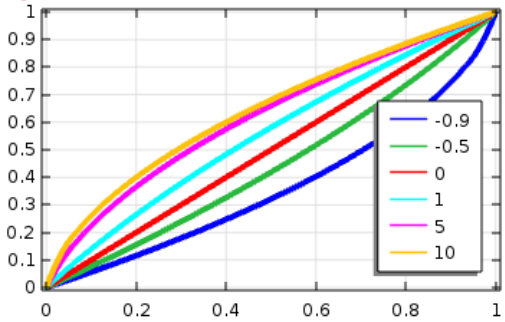
1D Plot Group 1:

`Legend` – **Show legends** = `True`

– **Position** = `Lower right`

1D Plot Group 1 – Line Graph 1:

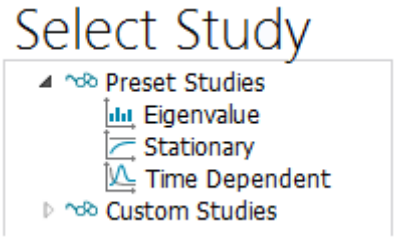
`Legend` – **Show legends** = `True`



5.2.5. Задача на власні значення.

Ще один тип досліджень, який іноді потрібно виконувати, полягає в пошуку власних чисел і власних функцій диференціального рівняння. Це багато в чому аналогічно нестаціонарним крайовим задачам – в рівнянні похідна за часом u'_t замінюється виразом $-\lambda u$, а друга похідна u''_{tt} – виразом $\lambda^2 u$. Наприклад, рівняння теплопровідності $u'_t = u''_{xx}$ набуває вигляду $u''_{xx} + \lambda u = 0$.

В системі Comsol Multiphysics для розв'язання таких задач призначений `Eigenvalue` тип дослідження, який, аналогічно типам `Stationary` і `Time Dependent`, обирається в панелі `Select Study`.



Тип дослідження `Eigenvalue` доступний тільки для коефіцієнтної і загальної форм рівнянь.

Приклад. Визначте власні числа і власні функції задачі

$$u'' + \lambda^2 u = 0,\ u(0) = 0,\ u(\pi) = 0.$$

Цей вибір обумовлений тим, що ми знаємо власні числа цієї задачі $\lambda_k = k$, $k \in \mathbf{Z}$ (цілі), і їм відповідають власні функції $C \sin kx$, які ми зможемо порівняти з розв'язком Comsol Multiphysics.

Р о з в ' я з о к . За допомогою майстра створіть 1D модель задачі

File -> New -> Model Wizard -> 1D -> Mathematics - PDE Interfaces - General Form PDE (g) -> Eigenvalue

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Геометрія моделі складається з відрізка $[0, \pi]$.

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1):

`Interval` – Left endpoint, Right endpoint = `0, pi`

LMB > Build All Objects

Побудуйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – Element Size = `Finer`

LMB > Build All

Задайте рівняння. Перевірте, що в ПВ вузла **General Form PDE (g) - General Form PDE 1** в секції вибору **`Domain Selection`** стоїть одиниця.

В розділі **`Equation`** записано рівняння в загальній (генеральній) формі, перетворене для розв'язання задач на власні значення

$$\lambda^2 e_a u - \lambda d_a u + \nabla \cdot \Gamma = f.$$

Параметри цього рівняння задаються в полях в нижній частині ПВ.

General Form PDE (g) - General Form PDE 1:

Conservative Flux – **Γ** = ux

Source Term – **f** = 0

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Mass Coefficient – $\mathbf{e_a}$ = 1

Цим задається рівняння $u'' + \lambda^2 u = 0$. Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

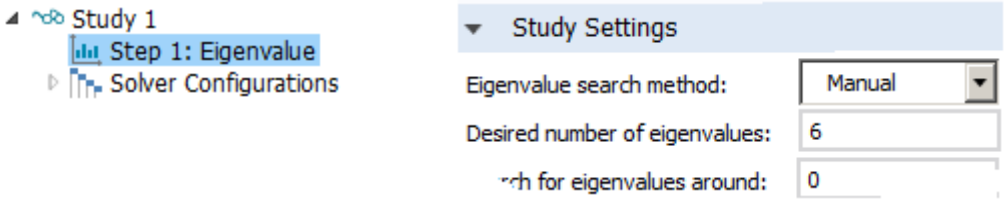
Boundary Selection – **Selection** = All boundaries

Dirichlet Boundary Condition – **r** = 0

Задайте кількість обчислюваних власних чисел.

Study 1 – Step 1: Eigenvalue:

Study Settings – **Desired number of eigenvalues** = 6



Перш, ніж розв'язувати задачу, призначте спосіб нормування власних функцій, які, як відомо, визначаються з точністю до сталого множника. Власні функції підбираються так, щоб їх норма дорівнювала одиниці.

Щоб відобразити дочірні вузли гілки **Study 1** (якщо в ДМ їх ще немає), виконайте команду

Study -> Show Default Solver

Тепер задайте тип норми.

Study 1 – Solver Configurations – Solution 1 - Eigenvalue Solver 1:

Output – **Scaling of eigenvectors** = Max

Цей вибір означатиме, що в якості норми функції застосовуватиметься її максимальне значення. Якщо ви залишите значення RMS (root mean square), то для норми буде використовуватися формула $\|f\| = \sqrt{\int_D f(x)\overline{f(x)}dx/|D|}$.

Виконайте команду Compute і в графічній зоні вузла **Results - 1D Plot Group 1 – Line Graph 1** отримайте графіки власних функцій.

Надрукуйте значення перших шести власних чисел і частот.

Results – Derived Values: RMB > Global Evaluation

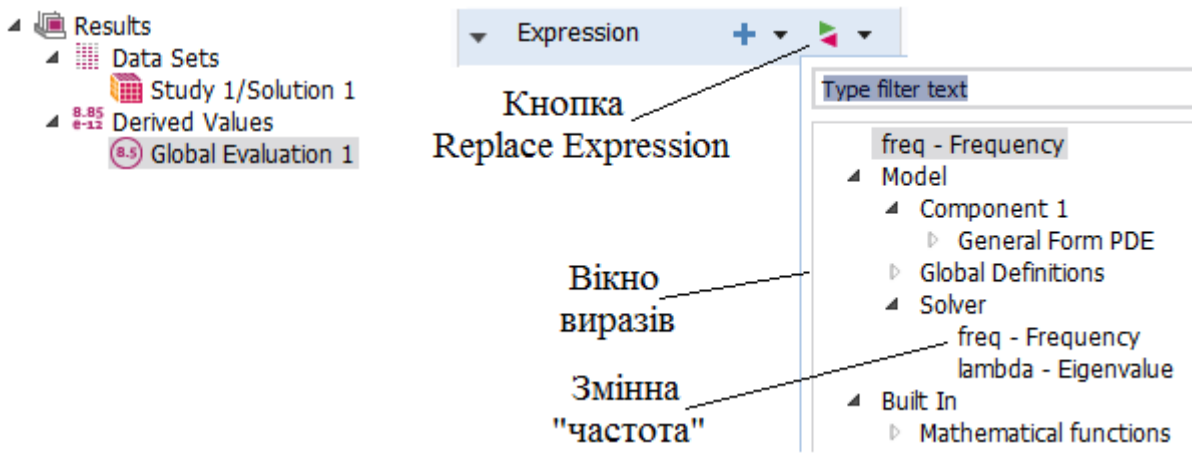
Global Evaluation 1:

Data – **Data set** = Study 1/Solution 1

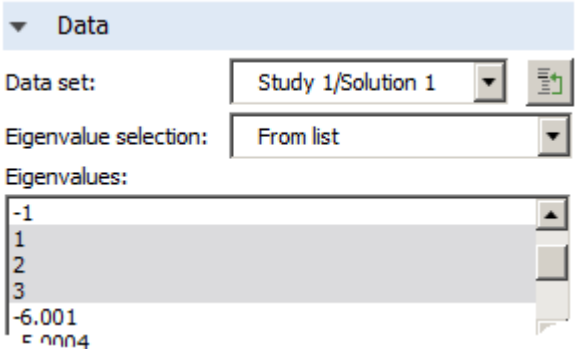
Data – **Eigenvalue selection** = From list

Щоб дізнатися (і ввести) ім'я змінної, яка представляє власні числа, в правій частині заголовка розділу **Expression** натисніть кнопку Replace expression . У вікні, яке з'явиться, розкрийте дерево змінних готових до вставки, знайдіть і виберіть подвійним клацанням змінну freq.

Expression – **Expression** = freq



В тій же панелі властивостей вузла **Global Evaluation 1** в списку **Eigenvalues**, утримуючи клавішу CTRL і клацаючи мишею по бажаним значенням, виберіть всі шість додатних власних чисел.



Клацніть вгорі ПВ по кнопці Evaluate, і надрукуйте ці власні числа в таблиці.

Eigenvalue	Frequency
1.0000	0.15915i
2.0000	0.31831i
3.0000	0.47747i
4.0001	0.63664i
5.0004	0.79584i
6.0010	0.95508i

У другому стовпці відобразяться значення ω_k, які пов'язані з власними числами λ_k формулою $\omega_k = \frac{\lambda_k}{2\pi} i$. Як видно з таблиці, четверте і далі власні числа визначені з деякою похибкою. Її можна зменшити, якщо СкЕ сітку зробити дрібнішою.

Mesh 1:

`Mesh Settings` –**Element size** = `Extremely fine`

Study 1: LMB > Compute

LMB> Build All

Надрукуйте самостійно нову таблицю власних чисел. Перші шість чисел будуть отримані вже з точністю до п'яти десяткових знаків.

Відкоригуйте графічну зону вузла **1D Plot Group 1**. Залиште в ній графіки власних функцій, які відповідають першим чотирьом додатним власним числам.

1D Plot Group 1:

`Data` – **Eigenvalue selection** = `From list`

Data – **Eigenvalues** = 1,2,3,4

Щоб виконати останній вибір, в списку **Eigenvalues** клацайте мишею по бажаним значенням, утримуючи клавішу CTRL.

1D Plot Group 1 - Line Graph 1:

Legend – **Show legends** = True

LMB > Plot

Зауважимо, що вибір знака власних функцій, які мають вигляд $\pm\sin\lambda_k x$, де $\lambda_k = k$, невизначений. Змінюючи розмір елементів, можна отримати ті ж власні числа, але з власними функціями іншого знака. На попередньому рисунку показані власні функції, побудовані при Extremely fine розмірах скінченних елементів.

Якщо необхідно побудувати графік тільки одного розв'язку, то в полі **Expression** графічного вузла **Line Graph** можна застосувати оператор with. Для виділення k-ї власної функції його слід використовувати наступним чином:

with(k,вираз),

де k – номер власного значення, другий аргумент являє вираз від змінної розв'язку, наприклад, u.

Для спроби побудуйте графік десятої власної функції. Для цього виконайте послідовність інструкцій:

Results -> **Plot Group** – 1D Plot Group

Results - 1D Plot Group 2:

Data – **Data set** = Study 1/Solution 1

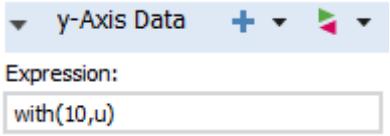
1D Plot Group 2 -> **Add Plot** – Line Graph

Results - 1D Plot Group 2 - Line Graph 1:

Selection – < > = 1

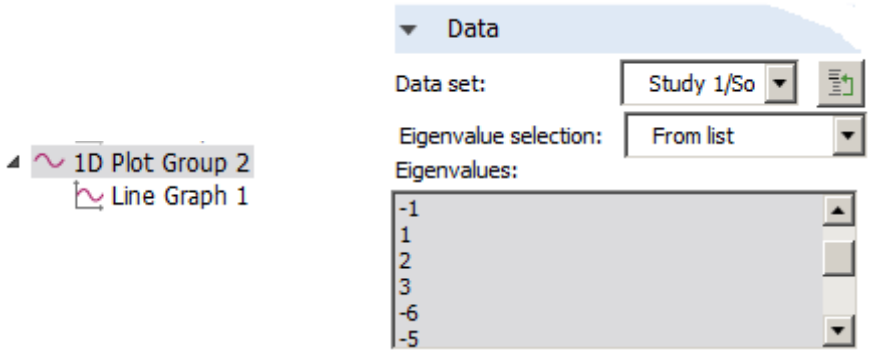
y-Axis Data – **Expression** = with(10,u)

x-Axis Data – **Parameter, Expression** = Expression, x



LMB > Plot

Останній графік відповідає десятому (по порядку в списку **Eigenvalue**) власному числу, побудованому системою. Список цих власних значень ви можете побачити в ПВ вузла **1D Plot Group 2**, якщо в розділі **`Data`** в списку **Eigenvalue selection** оберете значення `From List`.



В нашому випадку десяте власне число дорівнює 4.

Зауваження. Зверніть увагу, що розв'язок `u` представляє набір власних функцій, а змінна `freq` – власних значень.

Зауваження. Оператор `with` можна використовувати лише на етапі постобробки, тобто у вузлах гілки Results.

5.2.6. Поліноми Лежандра.

Чимало спеціальних функцій є власними функціями деяких диференціальних операторів. Зокрема, поліноми Лежандра $P_n(x)$ задовольняють диференціальному рівнянню

$$\frac{d}{dx}\left(\left(1-x^2\right)\frac{du}{dx}\right)+n(n+1)u=0,\ -1\le x\le 1. \qquad (1)$$

Однак, повторивши модель попереднього прикладу, ми не зможемо отримати ці функції. Справа в тому, що Comsol Multiphysics при `Eigenvalue` дослідженні завжди використовує однорідні граничні умови. Навіть тоді, коли ви додаєте ненульову граничну умову Діріхле $u=u_0\ne 0$, система інтерпретує її як $u=0$. Але поліноми Лежандра мають на кінцях відрізка $[-1,1]$ значення ± 1. Тому `Eigenvalue` дослідження тут не підходить. Якщо граничні умови неоднорідні, то для отримання розв'язку рівняння (1) (і аналогічних йому) можна виконати параметричний аналіз по параметру `n`.

Всі поліноми Лежандра на правому кінці відрізка $[-1,1]$ мають значення $+1$. На лівому кінці поліноми Лежандра парного порядку дорівнюють $+1$, а непарного -1. Тому граничні значення можна представити у вигляді

$$u(-1)=(-1)^n,\ u(1)=1. \qquad (2)$$

Отже дослідимо крайову задачу (1) – (2).

Р о з в ' я з о к . За допомогою майстра створіть стаціонарну 1D модель.

File->New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs - Helmholtz Equation(hzeq) -> Stationary

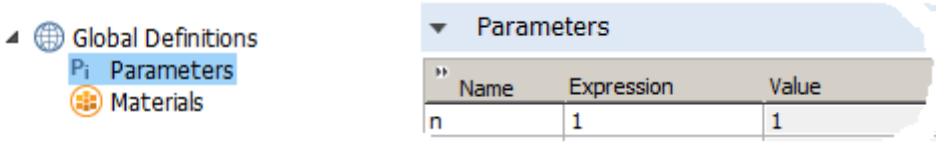
Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

В гілці **Global Definitions** створіть вузол параметрів.

Global Definitions: RMB > Parameters

В ПВ створеного вузла **Parameters** в таблиці параметрів в колонці **Name** введіть ім'я параметра n, а в колонці **Expression** введіть значення 1.



Геометрія моделі складається з відрізка [-1,1]. Створіть його.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1):

`Interval` – Left endpoint, Right endpoint = -1, 1

LMB > Build All Objects

Побудуйте СкЕ сітку.

Mesh 1:

`Mesh Settings` –Element size = Finer

LMB > Build All

Задайте рівняння. Перевірте, що в ПВ вузла **Helmholtz Equation (hzeq) - Helmholtz Equation 1** в секції вибору **`Domain Selection`** стоїть одиниця. В розділі **`Equation`** записано рівняння Гельмгольца в формі

$$\nabla\cdot(-c\nabla u)+\mathrm{a}\,u=f\,.$$

Параметри рівняння задайте в полях нижньої частини цього вікна.

Helmholtz Equation (hzeq) - Helmholtz Equation 1:

`Diffusion Coefficient` – c = x^2-1

`Source Term` – f = 0

`Absorption Coefficient` – a = n*(n+1)

Цим задається рівняння $\dfrac{d}{d\,x}\left(\left(1-x^2\right)\dfrac{d\,u}{d\,x}\right)+n(n+1)u=0\,.$

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

`Boundary Selection` – Selection, < > = Manual,1 (*лівий кінець відрізка*)

`Dirichlet Boundary Condition` – r = (-1)^n

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

`Boundary Selection` – Selection, < > = Manual,2 (*правий кінець відрізка*)

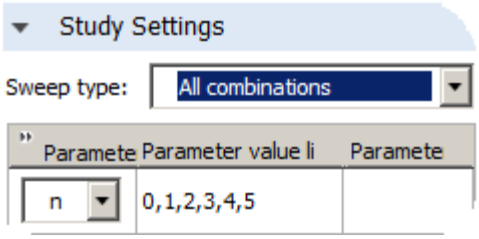
`Dirichlet Boundary Condition` – r = 1

Для виконання параметричного аналізу в гілці **Study 1** створіть вузол **Parametric Sweep 1**.

Study 1: RMB > Parametric Sweep

В ПВ створеного вузла **Parametric Sweep** в розділі **`Study Settings`** клацніть по таблиці параметрів правою кнопкою миші і виберіть команду Add. Із списку

виберіть, створений раніше параметр n. У другому стовпці таблиці задайте бажані значення цього параметра: 0,1,2,3,4,5.



Study 1: LMB > Compute

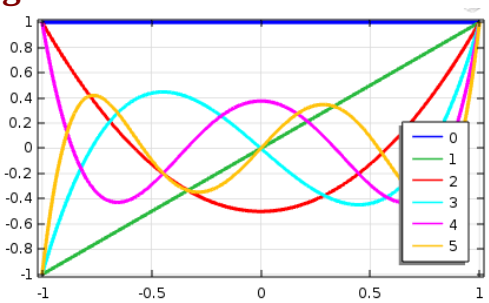
В гілці **Results** створюється вузол **1D Plot Group 1 - Line Graph 1**, в графічній зоні якого будуються графіки функцій, що є розв'язками крайової задачі при n=0,1,2,3,4,5. Для відображення на графіку легенди виконайте інструкції:

1D Plot Group 1:

Legend – **Show legends, Position** = True, Lower right

1D Plot Group 1 – Line Graph 1:

Legend – **Show legends** = True



Для порівняння побудуйте графік полінома $P_3(x)=\frac{1}{2}\left(5x^3-3x\right)$. Для цього в графічну групу **1D Plot Group 1** додайте ще один лінійний графік.

1D Plot Group 1 -> **Add Plot** – Line Graph

1D Plot Group 1 - Line Graph 2:

Selection – < > = 1 (*індикатор відрізка* [-1,1])

y-Axis Data – **Expression** = (5*x^3-3*x)/2

x-Axis Data – **Parameter, Expression** = Expression, x

LMB > Plot

Графік полінома Лежандра $P_3(x)$ накладається на криву попереднього рисунка, яка відповідає значенню параметра n = 3.

5.3. Нетипові крайові задачі.

5.3.1. Використання Global ODEs and DAEs інтерфейсу.

В застосуваннях виникають задачі, коли задаються інтегральні або усереднені характеристики шуканих величин, а не граничні умови. Але якісь граничні умови завжди існують і, якщо їх правильно підібрати, можна задовольнити бажаним обмеженням. Отже, окрім розв'язків ДРЧП, шуканими становляться параметри граничних умов або коефіцієнти рівнянь. Розглянемо декілька прикладів.

Приклад 1. Нескінченний циліндричний провідник радіуса R нагрівається постійним струмом, який виділяє в одиниці об'єму за одиницю часу тепло $Q = Const$. В усталеному осесиметричному випадку температура $T(r)$ задовольняє рівнянню

$$\frac{k}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) = -Q, \qquad (1)$$

де k – коефіцієнт теплопровідності, r – радіальна координата.

***Завдання* 1.1**. Температура поверхні відома T_0. Знайти потужність Q таку, що середня температура по перерізу циліндра дорівнює заданій величині $T_1 > T_0$. В модельній задачі покладемо k=1, R=1, $T_0 = 1, T_1 = 1.5$.

Розв'язання. Створіть одновимірну осесиметричну модель задачі.

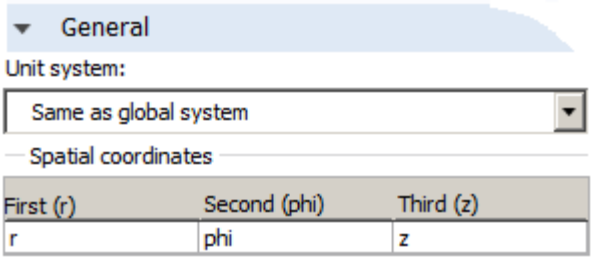
File->New->Model Wizard->1D Axisymmetric-> Mathematics – PDE Interfaces – Coefficient From PDE(c) + Variables = T -> Stationary

Зауваження. В режимі `1D Axisymmetric` немає доступу до інтерфейсів класичних ДРЧП. Тому ми обрали `Coefficient From PDE` інтерфейс.

Вимкніть в моделі спостереження за розмірностями фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

В ПВ вузла **Component 1** в розділі **`Spatial coordinates`** ви можете дізнатися (або поміняти) імена просторових змінних, які будуть використовуватися в моделі.

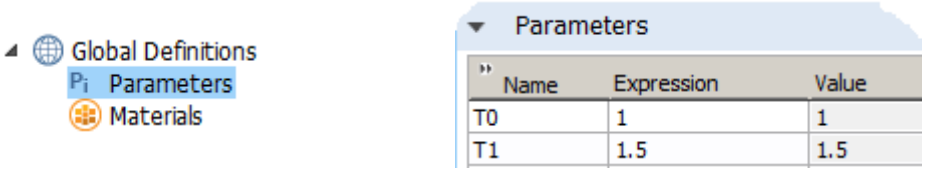


Зверніть увагу, що в режимі `1D Axisymmetric` за замовчуванням використовуються позначення координат (r, φ, z).

В гілці **Global Definitions** створіть вузол параметрів.

Global Definitions: RMB > Parameters

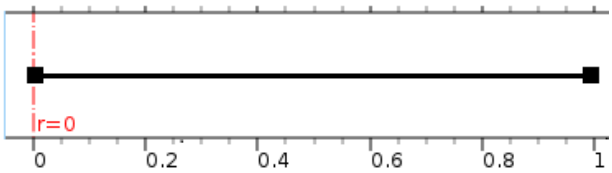
Задайте параметри відповідно до наступної таблиці.



Розрахункова область (переріз провідника) утворюється обертанням відрізка [0,1] навколо осі Z. Створіть цей відрізок.

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1) : LMB > Build All Objects



На створеному відрізку генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Налаштуйте рівняння. В ПВ вузла **Coefficient Form PDE (c) - CoefficientFromPDE 1** в розділі **`Equation`** наведено загальний вигляд коефіцієнтної форми ДРЧП

$$e_a \frac{\partial^2 T}{\partial t^2} + d_a \frac{\partial T}{\partial t} + \nabla \cdot \left(-c \nabla T - \alpha T + \gamma\right) + \beta \cdot \nabla T + \mathrm{a} T = f\,,\ \nabla = \frac{\partial}{\partial r}.$$

Значення коефіцієнтів рівняння введіть в полях нижньої частини цієї панелі.

Coefficient Form PDE (c) - CoefficientFromPDE 1:

`Diffusion Coefficient` – **c** = `r`

`Absorption Coefficient` – **a** = `0`

`Source Term` – **f** = `Q*r`

Інші коефіцієнти задайте нульовими, тобто $e_a = 0$, $d_a = 0$, $\alpha = 0, \beta = 0$, $\gamma = 0$.

Цим задається рівняння (1) у вигляді

$$\left(-r T_r'\right)_r' = r \cdot Q. \qquad (2)$$

Тут величина Q є невідомою. Вона буде підбиратися із додаткової умови, яка полягає в тому, що задана середня температура.

Гранична умова в точці $r = 0$ не задається, бо на початку координат вона непотрібна через осесиметричність. Задайте граничну температуру на поверхні циліндра.

`Physics` -> **`Boundary - Boundaries`** – Dirichlet Boundary Condition

Coefficient Form PDE (c) - Dirichlet Boundary Condition 1:

`Boundary Selection` – **Selection** = `Manual,2` (*точка* r=1)

`Dirichlet Boundary Condition` – **r** = `1`

Тепер створіть оператор зв'язку (Coupling Operators), який обчислюватиме середнє значення температури.

`Definitions` -> **`Coupling - Component Coupling`** – Average

Definitions - Average 1 (aveop1):

`Source Selection` – **Geometric entity level** = `Domain`

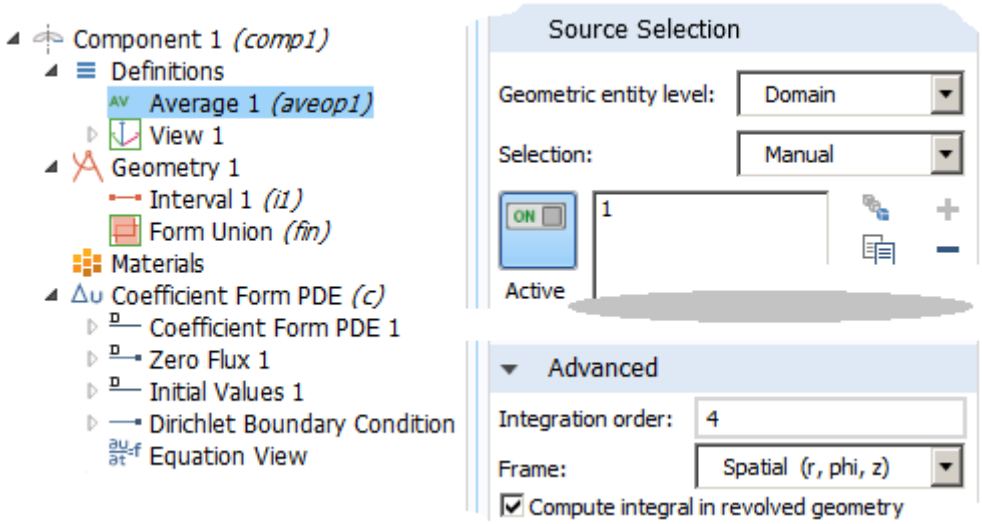
– **Selection** - < > = `1`

`Advanced` – **Compute integral in revolved geometry** = `True`

Останній прапорець примусить систему обчислювати середнє значення величини T по колу за формулою

$$\frac{1}{\pi R^2} \iint_{\text{по колу}} T\, d\, s = \frac{1}{\pi R^2} \int_0^{2\pi} \int_0^R T(r) \cdot r\, d\, r\, d\, \varphi.$$

Ім'я оператора `aveop1` вказується в полі **Operator name**. Оператор `aveop1(expr)` обчислює по обраній в ньому зоні середнє значення будь-якого виразу `expr`.



Використовуючи цей оператор, додайте в модель рівняння відносно невідомої `Q`.

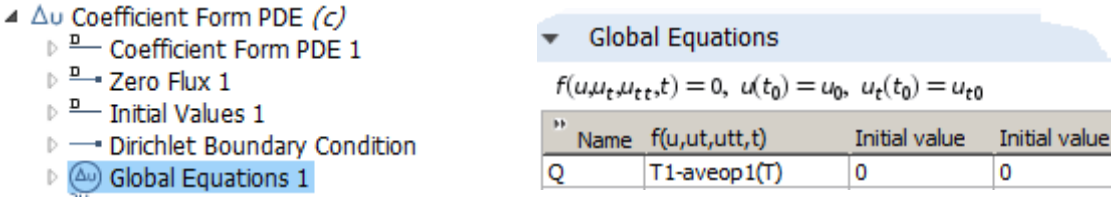
Створіть вузол `Global Equations`, в якому вкажіть невідому Q, а в якості рівняння напишіть умову, яка визначає середнє значення температури по колу.

`Physics` **`-> Global`** – Global Equations

Coefficient Form PDE (c) - Global Equations 1:

`Global Equations` – **Name** = `Q`

`Global Equations` – **f(u,ut,utt,t)** = `T1-aveop1(T)`



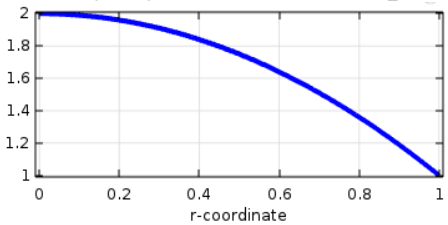
Розуміти запис в таблиці потрібно наступним чином. Задача буде розв'язуватися відносно невідомої Q одночасно з рівнянням у вузлі **Coefficient Form PDE 1** доки вираз в комірці **f(u,ut,utt,t)** не буде дорівнювати нулю, тобто доки середнє значення температури `aveop1(T)` не стане заданою величиною T_1.

Таким чином, в нашій задачі створено дві шукані величини: `T` та `Q`. При чому `T` – це полева змінна (функція від r), а Q – невідомий скаляр.

Виконайте обчислення.

`Home` **`-> Study`** - Compute

Після їх завершення в графічній зоні ви побачите одновимірний графік температури `T`, який належить вузлу **1D Plot Group 1 - Line Graph 1**.



У вузлі **2D Plot Group 2** буде створений поверхневий графік розподілення температури по колу. Для його побудови в гілці **Data Sets** створено джерело даних **Revolution 1D 1**. Скорегуйте його налаштування.

Results - Data Sets - Revolution 1D 1:

`Revolution Layers` – **Start angle, Revolution angle** = `0, 360`

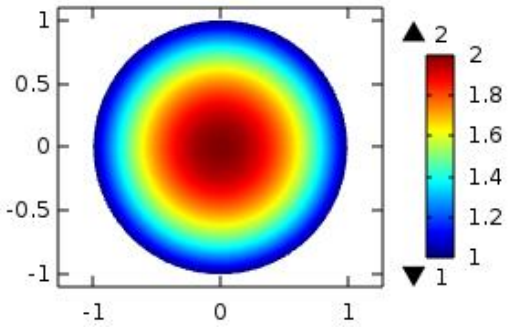
Тепер перебудуйте графік.

2D Plot Group 2:

`Plot Settings` – **Plot data set edges** = `False`

`Color Legend` – **Show maximum and minimum values** = `True`

LMB > Plot



Залишилося дізнатися величину потужності джерела тепла Q. Для друку значення цієї невідомої в гілці **Derived Values** створено вузол **State variable Q**.

Results - Derived Values - State variable Q:

LMB > Evaluate – New Table

В таблиці **Table 1** буде надрукований результат обчислення `Q=4.0000`.

State variable Q
4.0000

Збережіть модель в файл, але не закривайте його.

***Завдання* 1.2**. Температура поверхні циліндра відома T_0. Знайти потужність Q таку, що температура центральної точки дорівнює величині $T_1 > T_0$. В модельній задачі покладіть k=1, R=1, $T_0 = 1, T_1 = 1.5$.

Розв'язання. Для цієї задачі внесіть зміни в поточний файл. Рівняння теплопровідності залишиться, невідома потужність внутрішніх джерел тепла Q теж залишається. Але зміниться умова, з якої вона визначається.

Coefficient Form PDE (c) - Global Equations 1:

`Global Equations` – **f(u,ut,utt,t)** = `T1-at0(0,T)`

Global Equations

$f(u,u_t,u_{tt},t) = 0,\ u(t_0) = u_0,\ u_t(t_0) = u_{t0}$

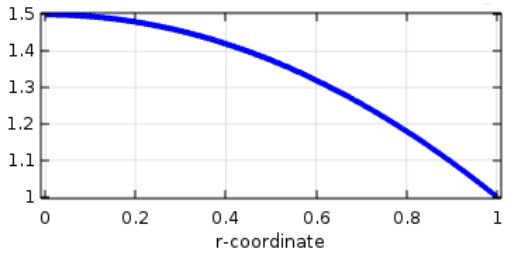
Name	f(u,ut,utt,t)	Initial value	Initial value
Q	T1-at0(0,T)	0	0

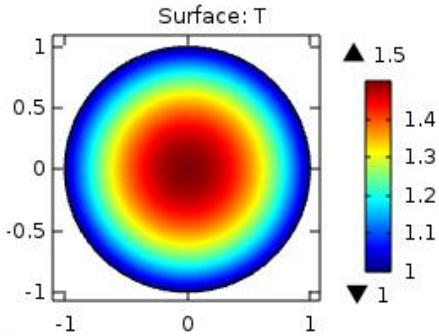
Тут оператор `at0(0,T)` використовується для обчислення значення температури в точці $r = 0$, тобто в центрі кола.

Виконайте обчислення.

`Home` -> `Study` - Compute

В результаті зміниться форма кривої, яка представляє залежність температури $T(r)$ від відстані до центра кола (наступний рисунок ліворуч), а також кольоровий графік розподілу температури по перерізу циліндра (наступний рисунок праворуч).





Надрукуйте значення знайденої потужності Q джерел тепла.

Results - Derived Values - State variable Q:

LMB > Evaluate –Table 1

В таблицю **Table 1** буде додана комірка з результатом обчислення `Q=2.0000`.

State variable Q	State variable Q
4.0000	2.0000

Збережіть файл, але не закривайте його.

***Завдання* 1.3**. Джерело тепла сталої потужності q діє всередині кола радіуса $r_0 < R$. Тіло випромінює тепло по закону Ньютона

$$T'_r\big|_{r=R} = g\left(T_2 - T\right)\big|_{r=R}, \qquad (3)$$

де g – коефіцієнт теплообміну з навколишнім середовищем з температурою T_2. Підібрати T_2 так, щоб середня температура внутрішнього кола радіуса $r_0 < R$ дорівнювала заданій величині $T_1 > T_0$. В модельній задачі покладіть k=1, R=1, $q = 2$, $r_0 = 0.5$, $T_1 = 1.5$, $g = 5$.

Розв'язання. Для цієї задачі внесіть зміни в поточний файл. Додайте значення нових параметрів моделі відповідно до наступної таблиці.

Parameters

Name	Expression	Value
T0	1	1
T1	1.5	1.5
r0	0.5	0.5
g	5	5
q	2	2

Для створення внутрішнього кола додайте в геометрію точку $r = r_0$.

`Geometry` **`-> Primitives->`** Point

Geometry 1 – Point 1:

`Point` – **x** =`r0`

LMB > Build All Objects

У вузлі **Coefficient Form PDE 1** обнуліть праву частину рівняння.

Coefficient Form PDE (c) - CoefficientFromPDE 1:

`Source Term` – **f** = `0`

Додайте в «фізику» окреме джерело тепла (типу `Source`) сталої потужності q, яке діятиме в колі $r \le r_0$.

`Physics` **`-> Domain - Domains`** – Coefficient Form PDE – Source

Coefficient Form PDE (c) - Coefficient Form PDE 1 – Source 1:

Domain Selection – **Selection** = Manual,1 (*відрізок* $[0,r_0]$)

Source Term – **f** = r*q

Тут множник r використано тому, що в циліндричних координатах при запису рівняння у вигляді (2) потужності всіх джерел тепла множаться на r.

Деактивуйте вузол граничної умови Діріхле і створіть вузол типу Flux/Source 1.

Coefficient Form PDE (c) - Dirichlet Boundary Condition 1: RMB > Disable

Physics -> **Boundary - Boundaries** – Flux/Source

Coefficient Form PDE (c) - Flux/Source 1:

Boundary Selection – **Selection,** < > = Manual,3 (*точка* r=1)

Boundary Flux/Source – **g** =g*T2*r

– **q** =g*r

Враховуючи, що $c = r$, ми задали умову $rT_r'\big|_{r=R} = g \cdot r \cdot T_2 - g \cdot r \cdot T\big|_{r=R}$. Це співпадає з граничною умовою (3).

Створіть оператор обчислення середнього значення температури по колу радіуса r_0.

Definitions **-> Coupling - Component Coupling** – Average

Definitions - Average 1 (aveop2):

Source Selection – **Geometric entity level** = Domain

– **Selection -** < > = 1 (*відрізок* $[0, r_0]$)

Advanced – **Compute integral in revolved geometry** = True

Нагадаємо, що остання опція визначає обчислення середнього значення по колу радіуса r_0 (а не по відрізку $[0, r_0]$).

В полі **Operator name** обране системою ім'я оператора aveop2 залиште.

Внесіть зміни у вузол **Global Equations 1.**

Coefficient Form PDE (c) - Global Equations 1:

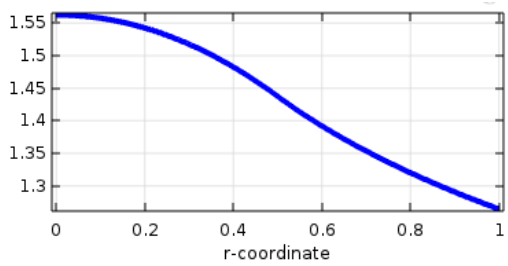
Global Equations – **Name** = T2

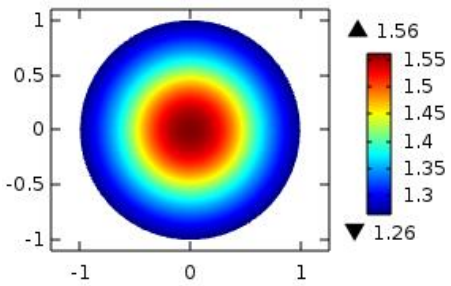
Global Equations – **f(u,ut,utt,t)** = T1- aveop2(T)

Виконайте обчислення.

Home **-> Study** - Compute

Крива, яка представляє залежність температури $T(r)$ від відстані до центра кола, буде перебудована (наступний рисунок ліворуч). Новим буде і кольоровий графік розподілу температури по перерізу циліндра (наступний рисунок праворуч).





Надрукуйте значення знайденої температури T2 зовнішнього середовища.

Results - Derived Values - State variable Q:

Label = State variable T2

`Expression` – **Expression** =`T2`

LMB > Evaluate –Table 1

В таблицю **Table 1** буде додана нова комірка з результатом обчислення `T2=1.2142`.

State variable Q	State variable Q	State variable T2
4.0000	2.0000	1.2142

Вправа. На відрізку $\Omega = [0,1]$ дано рівняння Пуассона $\Delta u = f$ з нульовими граничними умовами. Відомо, що f – стала, але така, що:

1. Інтеграл від розв'язку по відрізку [0.25,0.75] дорівнює одиниці.
2. Середнє значення розв'язку по всьому відрізку дорівнює одиниці.
3. Значення розв'язку u в точці $x_0 = 0.5$ дорівнює заданій величині a=1 (тобто $u(x_0) = a$).

Для кожної умови розв'язати крайову задачу для рівняння Пуассона і підібрати відповідне значення невідомого параметра f.

5.3.2. Використання в рівняннях операторів.

Диференціальні рівняння в коефіцієнтах, граничних умовах і вільних членах можуть містити не тільки арифметичні операції, функції, параметри і змінні моделі, а і оператори системи або користувача. Частіше за все використовується оператор `integrate(...)`. Але зустрічаються випадки застосування інших операторів. Розглянемо кілька учбових задач, рівняння яких містять оператори.

Приклад. Розв'язати крайову задачу $-u''_{xx} = f(x), 0 < x < 1,\ u(0) = -1,\ u(1) = 1$ з наступним вільним членом f:

1. $f(x) = 1 - \int_0^1 u\,dx$;
2. $f(x) = 1 + 64 \cdot \int_0^x u\,dx$;
3. $f(x) = \begin{cases} -1 + 10 \cdot \bar{u}_{[1/3,2/3]}, & 1/3 \le x \le 2/3 \\ -1, & \text{інакше} \end{cases}$, де $\bar{u}_{[1/3,2/3]}$ позначає середнє значення величини u на відрізку [1/3,2/3];
4. $f(x) = \begin{cases} 10 \cdot u(0.75), & 1/3 \le x \le 2/3 \\ 0, & \textit{інакше} \end{cases}$;
5. $f(x) = u(0.6) + u(0.7) + u(0.8) + u(0.9)$;
6. $f(x) = -10\left(1 - x^2\right) + 10 \cdot \delta(x - 2/3) \cdot u(2/3)$, де $\delta(x)$ - дельта-функція Дірака.

При створенні моделей для цих задач на відрізку [0,1] генеруватиметься одна фізика (рівняння Пуассона) і використовуватимуться однакові граничні умови

Діріхле. В моделях змінюватиметься вільний член $f(x)$ або створюватимуться додаткові вузли для його корегування.

Розв'язання. Створіть модель зі стаціонарним Classical PDEs – Poisson's Equation (poeq) інтерфейсом.

File -> New -> Model Wizard -> 1D -> Mathematics - Classical PDEs – Poisson's Equation (poeq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Встановіть прапорець, який визначатиме відображення індикаторів елементів геометрії.

Component 1 – Definitions - View 1:

View – **Show geometry labels** = True

Геометрія моделей складатиметься з відрізка [0,1].

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1) : LMB > Build All Objects

Побудуйте СкЕ сітку.

Mesh 1:

Mesh Settings –**Element size** = Extra fine

LMB> Build All

1. $-u''_{xx} = 1 - \int_0^1 u\,dx,\ 0 < x < 1,\ u(0) = -1,\ u(1) = 1.$

Створіть оператор інтегрування по відрізку [0,1].

Definitions **-> Coupling - Component Coupling** – Integration

Component 1 - Definitions - Integration 1 (intop1):

Source Selection – **Geometric entity level** = Domain

– **Selection -** < > = 1 (*відрізок* [0,1])

Ім'я оператора intop1 в полі **Operator name** залиште без змін. Нагадаємо, що оператор intop1(expr) інтегрує будь-який вираз expr, заданий на зоні дії оператора, по цій зоні (зараз по відрізку [0,1]).

Перевірте, що в ПВ вузла **Poisson's Equation (poeq) - Poisson's Equation 1** в розділі **Domain Selection** в секції вибору стоїть одиниця. В розділі **Equation** записано рівняння Пуассона в формі

$$\nabla \cdot (-c\nabla u) = f(x).$$

Параметри рівняння задайте в полях нижньої частини ПВ.

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = 1

Source Term – **f** = 1-intop1(u)

Цим задається рівняння $-u''_{xx} = 1 - \int_0^1 u\,dx$.

Задайте граничні умови.

Physics -> **Boundariy - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** <> = Manual,1 (*лівий кінець відрізка*)

Dirichlet Boundary Condition – **r** =-1

Physics -> **Boundariy - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

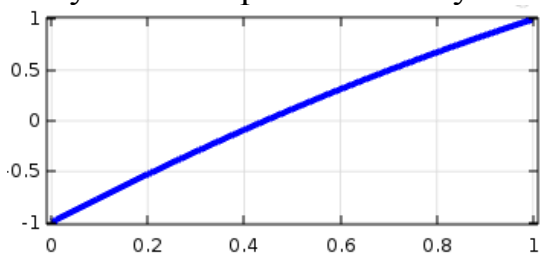
Boundary Selection – **Selection,** <> = Manual,2 (*правий кінець відрізка*)

Dirichlet Boundary Condition – **r** = 1

Виконайте обчислення

Home **-> Study** – Compute

і отримайте графік розв'язку. Стиль кривої налаштуйте на свій розсуд.



Аналітичний розв'язок U цієї задачі має вид

$$U(x) = \frac{1}{2}(-2 + \frac{64x}{13} - \frac{12x^2}{13})$$

Для перевірки ви можете самостійно добудувати графік функцій U в гілці **1D Plot Group 1**.

2. $-u''_{xx} = 1 + 64 \cdot \int_0^x u\,d\,x,\ 0 < x < 1,\ u(0) = -1,\ u(1) = 1.$

В правій частині рівняння використана первісна $\int_0^x u\,d\,x$ шуканої величини.

Внесіть зміни в параметри рівняння.

Poisson's Equation (poeq) - Poisson's Equation 1:

Source Term – **f** = 1+64*intop1(u*(x<=dest(x)))

Для зображення первісної було застосовано інтеграл зі змінною верхньою межею інтегрування, який конвертовано в визначений інтеграл по відрізку

$$\int_a^x u(\xi)d\,\xi = \int_0^1 u(\xi)\,(\xi \le x)\,d\,\xi\,.$$

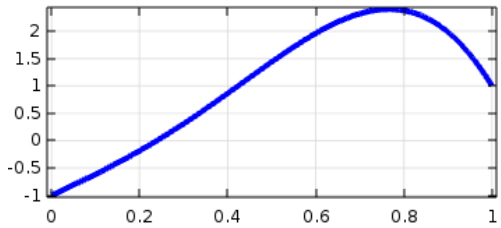
Тут додатковий множник $(\xi \le x)$, який дорівнює +1 або 0, обнуляє підінтегральний вираз на ділянці $\xi > x$. Щоб розрізнити просторову координату x і змінну інтегрування ξ (вона теж змінюється по відрізку [0,1]), зовнішню абсцису x в операторі інтегрування замінено оператором dest(x).

В результаті задається рівняння $-u''_{xx} = 1 + 64 \cdot \int_0^x u\,d\,x$.

Виконайте обчислення моделі.

Home **-> Study** - Compute

і отримайте наступний графік розв'язку.



В цьому прикладі корисно побудувати графік первісної розв'язку.

`Results` -> **`Plot Group`** – 1D Plot Group

`1D Plot Group 2` -> **`Add Plot`** – Line Graph

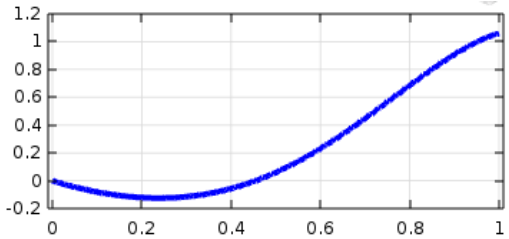
1D Plot Group 2 - Line Graph 1:

`Selection` – < > = 1 (*відрізок* [0,1])

`y-Axis Data` – **Expression** = `intop1(u*(x<=dest(x)))`

`x-Axis Data` – **Parameter, Expression** = `Expression,x`

LMB > Plot



Вільний член рівняння $1+64\cdot\int_0^x u\,d\,x$ (як і первісна) на правій частині відрізка [0,1] додатний, а на лівій – від'ємний (приблизно). Отже і випуклість кривої *u* на цих ділянках різна.

3. Продовжить роботу з поточним файлом.

Для застосування оператора обчислення середнього значення розв'язку на відрізку [1/3,2/3] в геометрії потрібно мати такий відрізок. Для цього в розрахункову область додайте дві точки – кінці відрізка.

`Geometry` -> **`Primitives`**-> Point

Geometry 1 – Point 1:

`Point` – **x** =`1/3, 2/3`

LMB > Build All Objects

Створіть оператор обчислення середнього значення по відрізку [1/3,2/3].

`Definitions` -> **`Coupling - Component Coupling`** – Average

Component 1 - Definitions - Average 1 (aveop1):

`Source Selection` – **Geometric entity level** = `Domain`

– **Selection** - < > = 2 (*відрізок* [1/3,2/3])

Ім'я оператора `aveop1` в полі **Operator name** залиште без змін.

Перевірте, що граничні умови Діріхле залишилися заданими на кінцях відрізка. Зміни могли статися, бо після додавання внутрішнього відрізка [1/3,2/3] індикатори точок помінялася.

У вузлі **Poisson's Equation 1** змініть вільний член.

Poisson's Equation (poeq) - Poisson's Equation 1:

`Source Term` – **f** = `-1`

В гілці **Poisson's Equation (poeq)** створіть вузол типу `Source`, який представлятиме додатковий зовнішній вплив на відрізку [1/3,2/3].

`Physics` **-> Domain - Domains** – Poisson's Equation – Source

Poisson's Equation (poeq) – Source 1:

Domain Selection – **Selection** = `Manual,2` (*відрізок* [1/3,2/3])

На відрізку [1/3,2/3] буде діяти зовнішня сила.

Source Term – **f** = `10*aveop1(u)`

В результаті задається рівняння $-u''_{xx} = f(x)$, де

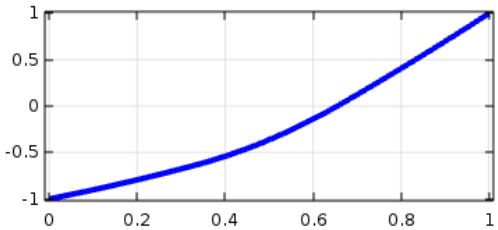
$$f(x) = \begin{cases} -1 + 10 \cdot \bar{u}_{[1/3,2/3]} \ , & 1/3 \le x \le 2/3 \\ -1 \ , & \text{інакше} \end{cases},$$

і $\bar{u}_{[1/3,2/3]}$ позначає середнє значення величини u на відрізку [1/3,2/3].

Виконайте обчислення моделі

`Home` **-> Study** - Compute

і у вузлі **1D Plot Group 1** отримайте графік розв'язку.



Аналітичний розв'язок цієї задачі має вид:

$$U(x) = \begin{cases} -1 + \frac{53}{57}x + \frac{x^2}{2} \ , & x < \frac{1}{3} \\ -1 + \frac{53}{57}x + \frac{1}{342}(65 - 390\,x + 756\,x^2) \ , & \frac{1}{3} \le x \le \frac{2}{3} \\ -1 + \frac{53}{57}x + \frac{1}{114}(-65 + 130\,x + 57\,x^2) \ , & x > \frac{2}{3} \end{cases}$$

Для перевірки можете самостійно побудувати графік функції U в гілці **1D Plot Group 1**.

4. $-u''_{xx} = f(x), 0 < x < 1,\ u(0) = -1,\ u(1) = 1$, де вільний член $f(x)$, заданий на відрізку [1/3,2/3], є сталою величиною, пропорційною значенню розв'язку u в фіксованій точці, наприклад,

$$f(x) = \begin{cases} 10 \cdot u(0.75) \ , & 1/3 \le x \le 2/3 \\ 0 \ , & \text{інакше} \end{cases}.$$

Внесіть зміни в параметри рівняння, а також в поля ПВ вузла **Source 1**.

Poisson's Equation (poeq) - Poisson's Equation 1:

Source Term – **f** = `0`

Poisson's Equation (poeq) – Source 1:

На відрізку [1/3,2/3] буде діяти зовнішня сила.

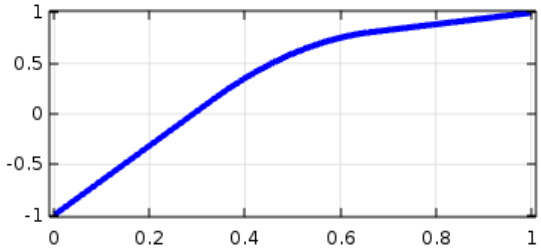
Source Term – **f** = `10*at1(0.75,u)`

Нагадуємо, що оператор `at1(x0,expr)` використовується в 1D геометріях для обчислення виразу `expr` в точці з координатою `x0`.

Виконайте обчислення моделі.

`Home -> Study` - Compute

і отримайте наступний графік розв'язку.



Аналітичний розв'язок цієї задачі має вигляд:

$$U(x) = \begin{cases} -1 + \dfrac{24}{7}x, & x < \dfrac{1}{3} \\ \dfrac{1}{21}(-31 + 132\,x - 90\,x^2), & \dfrac{1}{3} \le x \le \dfrac{2}{3} \\ \dfrac{1}{7}(3 + 4\,x), & x > \dfrac{2}{3} \end{cases}$$

Для перевірки можете самостійно побудувати графік функції U в гілці **1D Plot Group 1**.

5. $-u''_{xx} = f(x), 0 < x < 1,\ u(0) = -1,\ u(1) = 1$, де вільний член $f(x)$ є сталою, яка обчислюється за формулою

$$f(x) = u(0.6) + u(0.7) + u(0.8) + u(0.9).$$

Деактивуйте вузол **Source 1** і внесіть зміни в ПВ вузла **Poisson's Equation 1**.

Poisson's Equation (poeq) – Source 1: RMB > Disable

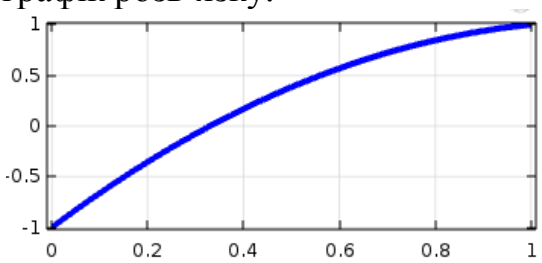
Poisson's Equation (poeq) - Poisson's Equation 1:

Source Term – **f** = `sum(at1(0.5+i*0.1,u),i,1,4)`

Виконайте обчислення моделі.

`Home -> Study` - Compute

і отримайте наступний графік розв'язку.



Аналітичний розв'язок цієї задачі має вид:

$$U(x) = \frac{1}{2}\left(-2 + \frac{92}{13}x - \frac{40}{13}x^2\right)$$

Для перевірки самостійно добудуйте/перебудуйте графік функції U в гілці **1D Plot Group 1**.

6. $f(x) = -10\left(1 - x^2\right) + 10 \cdot \delta(x - 2/3) \cdot u(2/3)$. Другий доданок в правій частині рівняння імітує зосереджене зовнішнє зусилля або джерело, яке пропорційне розв'язку в точці x=2/3.

Внесіть зміни в параметри рівняння і додайте вузол точкового джерела **Flux/Source 1**.

Poisson's Equation (poeq) - Poisson's Equation 1:

Source Term – **f** = -10*(1-x^2)

Physics -> **Boundary - Boundaries** – Flux/Source

Poisson's Equation (poeq) - Flux/Source 1:

Boundary Selection – **Selection,** < > = Manual,3 (*точка* x=2/3)

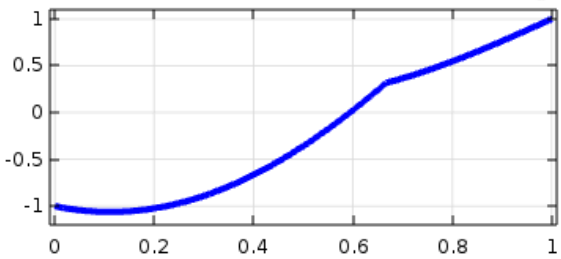
Boundary Flux/Source – **g** = 10*at1(2/3,u)

Тут точка прикладання зусилля x=2/3 існує в геометрії моделі, а величина зусилля 10*at1(2/3,u) за допомогою оператора at1(2/3,...) береться з тієї ж точки x=2/3.

Виконайте обчислення моделі

Home -> **Study** - Compute

і отримайте наступний графік розв'язку.



Аналітичний розв'язок цієї задачі має вид:

$$U(x) = \begin{cases} -1 - \dfrac{1981}{1782}x + 5\,x^2 - \dfrac{5}{6}x^4\,, & x < \dfrac{2}{3} \\ \dfrac{989}{891} - \dfrac{7621}{1782}x + 5x^2 - \dfrac{5}{6}x^4\,, & x > \dfrac{2}{3} \end{cases}$$

Для перевірки самостійно добудуйте/перебудуйте графік функції U в гілці **1D Plot Group 1**.

5.4. Демонстраційні приклади.

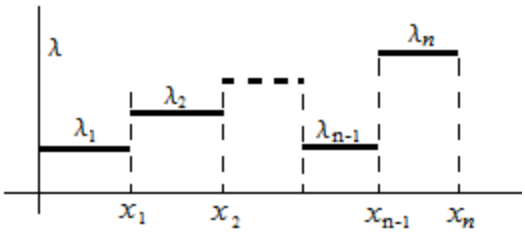
5.4.1. Теплопровідність в багатошаровій плоскій стінці.

Дано набір n плоских контактуючих необмежених пластин товщиною d_i з коефіцієнтами теплопровідності λ_i $(i = 1, 2, \ldots, n)$; тепло передається через пластини нормально їх граням, внутрішніх джерел тепла немає. Температура на зовнішніх поверхнях дорівнює u_0 і u_n. Треба знайти розподіл температури всередині такої багатошарової плоскої стінки.

Математична модель задачі полягає в розв'язанні стаціонарного рівняння теплопровідності $\dfrac{d}{d\,x}\left(\lambda(x)\dfrac{d\,u}{d\,x}\right) = 0$ з кусково-сталою функцією $\lambda(x)$:

$$\lambda(x)=\begin{cases}\lambda_1, & x_0 \le x < x_1 \\ \dots & \\ \lambda_n, & x_{n-1} \le x \le x_n\end{cases},$$

$$0 = x_0 < x_1 < \dots < x_{n-1} < x_n, \quad d_i = x_i - x_{i-1}$$



Розв'язок цієї задачі відомий. Він має вигляд

$$u = \frac{u_0 + u_n}{2} + \frac{u_n - u_0}{2\sum_{i=1}^{n} d_i / \lambda_i}\left(\frac{x - x_0}{\lambda_1} + \frac{x - x_n}{\lambda_n} + \sum_{i=1}^{n-1}\left(\frac{1}{\lambda_{i+1}} - \frac{1}{\lambda_i}\right)\left|x - x_i\right|\right) \qquad (1)$$

і ми зможемо порівняти його з розв'язком, який побудує система.

Приклад. Розглянемо задачу розподілу температури в тришаровій стінці при наступних значеннях: $\{x_i\}_{i=0}^{3} = \{0,1,2,3\}$, $\{\lambda_i\}_{i=1}^{3} = \{1,2,5\}$, $u_0 = 4$, $u_3 = 1$.

Р о з в ' я з а н н я . За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Heat Equation(hteq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Геометрія моделі складатиметься з трьох прилеглих відрізків [0,1], [1,2] і [2,3].

`Geometry` -> **`Primitives`** -> Interval

Geometry 1 - Interval 1 (i1) :

Interval – Number of intervals = `Many`

Interval – Points = `0,1,2,3`

Geometry 1 - Interval 1 (i1) : LMB > Build All Objects

Шари (відрізки) розрізняються коефіцієнтом теплопровідності. На кожному з них «працює» своє рівняння. Тому створіть в моделі ще дві «фізики», додавши в неї два рівняння.

`Physics` -> **`Domains`** -> Classical PDEs - Heat Equation

`Physics` -> **`Domains`** -> Classical PDEs - Heat Equation

В результаті в гілці **Heat Equation (hteq)** з'являться два нових вузла **Heat Equation 2** і **Heat Equation 3**. Налаштуйте їх параметри.

Heat Equation 2:

`Domain Selection` – **Selection,** <> = `Manual, 2` (*відрізок* [1,2])

`Diffusion Coefficient` – **c** = `-2`

`Source Term` – **f** = `0`

`Damping or Mass Coefficient` – $\mathbf{d_a}$ = `0`

Heat Equation 3:

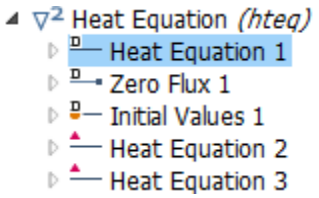
`Domain Selection` – **Selection,**<> = `Manual,3` (*відрізок* [2,3])

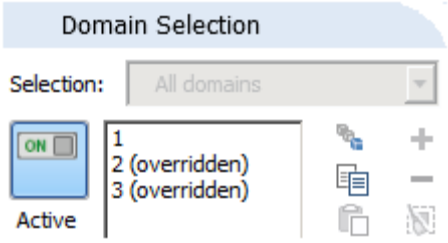
`Diffusion Coefficient` – **c** = `-5`

`Source Term` – **f** = `0`

`Damping or Mass Coefficient` – $\mathbf{d_a}$ = `0`

Відкрийте ПВ вузла **Heat Equation 1**. Його розділ **`Domain Selection`** повинен мати наступний вигляд.





Тут видно, що відрізки з індикаторами 2 і 3 не входять до зони дії першого рівняння. В ПВ цього вузла введіть наступні значення коефіцієнтів рівняння.

Heat Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = 0

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** < > = Manual, 1 (*точка x*=0)

Dirichlet Boundary Condition – **r** = 4

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection,** < > = Manual,4 (*точка x*=3)

Dirichlet Boundary Condition – **r** = 1

На внутрішніх границях умови задавати непотрібно. На них автоматично реалізовуватиметься неперервність теплового поля (температури і теплового потоку).

Побудуйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

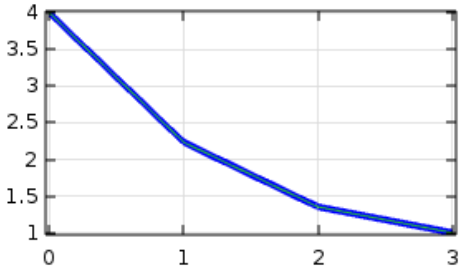
Mesh Settings – **Element Size** =Finer

LMB> Build All

Виконайте обчислення.

Study 1: LMB > Compute

В гілці **Results** створюється вузол **1D Plot Group 1 - Line Graph 1**, в графічній зоні якого будується ламана, що представляє графік розв'язку нашої тришарової задачі теплопровідності.



Для порівняння побудуйте графік розв'язку (1) при обраних значеннях параметрів n=3, $\{x_i\}_{i=0}^{3} = \{0,1,2,3\}$, $\{\lambda_i\}_{i=1}^{3} = \{1,2,5\}$, $u_0 = 4, u_3 = 1$. Для цього в графічну групу **1D Plot Group 1** додайте ще один лінійний графік.

1D Plot Group 1 -> **Add Plot** – Line Graph

1D Plot Group 1 - Line Graph 2:

Selection – < > = 1,2,3 (*індикатори всіх трьох відрізків*)

y-Axis Data – **Expression** =5/2-
15/17*(6/5*x-3/5-abs(x-1)/2-3*abs(x-2)/10)

x-Axis Data – **Parameter, Expression** = Expression, x

LMB > Plot

Графік введеного виразу, побудованому за формулою (1), в точності накладається на ламану останнього рисунка.

Вправа 1. Розв'яжіть попередню задачу, використовуючи на відрізку [0,3] єдине рівняння з кусково заданим коефіцієнтом теплопровідності. Побудуйте залитий контурний графік, який представляє розподіл температури по товщині багатошарової стінки.

Вправа 2. Дослідіть стаціонарний розподіл температури в багатошаровій плоскій стінці з заданим теплообміном на границях.

Дано n плоских контактуючих необмежених пластин товщиною $d_i = x_i - x_{i-1}$ $(0 = x_0 < x_1 < \ldots < x_n)$ з коефіцієнтами теплопровідності λ_i $(i = 1, 2, \ldots, n)$. Температура середовищ T_l, T_r, що омивають ліву і праву стінки задані. Знайти розподіл температури в такій багатошаровій плоскій стінці.

Задача полягає в розв'язанні рівняння теплопровідності

$$\frac{d}{dx}\left(\lambda(x)\frac{du}{dx}\right) = 0$$

з кусково-сталою функцією $\lambda(x)$. На лівому і правому кінцях при $x = x_0 = 0$ і $x = x_n$ маємо теплообмін з навколишнім середовищем за законом Ньютона

$$-\lambda_1 u'_x(0) = \alpha_l (T_l - u(0)) \quad \text{і} \quad -\lambda_n u'_x(x_n) = \alpha_r (u(x_n) - T_r),$$

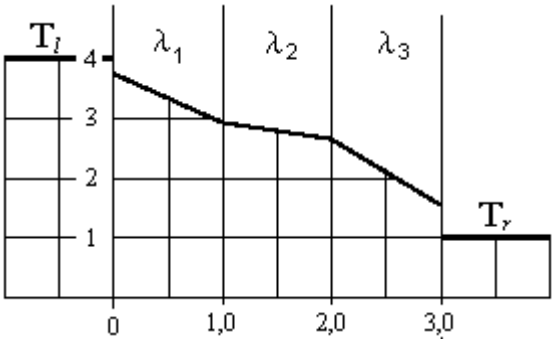
де α_l, α_r – коефіцієнти теплообміну на лівому і правому торцях.

Розв'язок цієї задачі відомий. Він має вигляд

$$u = \frac{T_l + T_r}{2} - \frac{\eta \cdot (T_l - T_r)}{2}\left(\frac{1}{\alpha_l} - \frac{1}{\alpha_r} + \frac{x - x_0}{\lambda_1} + \frac{x - x_n}{\lambda_n} + \sum_{i=1}^{n-1}\left(\frac{1}{\lambda_{i+1}} - \frac{1}{\lambda_i}\right)|x - x_i|\right),$$

де $\eta = 1/\left(1/\alpha_l + \sum_{i=1}^{n} d_i/\lambda_i + 1/\alpha_r\right)$.

Розгляньте задачу розподілу температури в тришаровій плоскій стінці при наступних значеннях параметрів: $T_l = 4$, $T_r = 1$, $\{x_i\}_{i=0}^{3} = \{0,1,2,3\}$, $\alpha_l = 2$, $\alpha_r = 1$, $\{\lambda_i\}_{i=1}^{3} = \{2/3, 2, 1/2\}$.



5.4.2. Теплопровідність в багатошаровій циліндричній стінці.

Стаціонарне однорідне рівняння теплопровідності має вигляд $\nabla\cdot(\lambda\nabla u)=0$, де λ – коефіцієнт внутрішньої теплопровідності. Якщо стінка являє собою порожнистий n - шаровий циліндр, вісь якого збігається з віссю z, температура встановилася, граничні умови не залежать від кутової координати і координати z, коефіцієнт внутрішньої теплопровідності λ залежить тільки від радіуса r, то ми маємо осесиметричну задачу. У цьому випадку температура циліндра залежить тільки від радіальної координати r, і ми приходимо до крайової задачі для ЗДР:

$$\frac{d}{dr}\left(\lambda(r)\,r\,\frac{du}{dr}\right)=0,\quad u(r_0)=u_0,\quad u(r_n)=u_n,$$

де $0<r_0<r_1<\ldots<r_n$ і $\lambda(r)=\begin{cases}\lambda_1, & r_0\le r<r_1\\ \ldots & \\ \lambda_n, & r_{n-1}\le r\le r_n\end{cases}$.

Розв'язок цієї задачі відомий. Він має вигляд

$$u(r)=\frac{u_0+u_n}{2}-\frac{q}{2}\left(\frac{1}{\lambda_1}\ln\frac{r}{r_0}+\frac{1}{\lambda_n}\ln\frac{r}{r_n}+\sum_{i=1}^{n-1}\left(\frac{1}{\lambda_{i+1}}-\frac{1}{\lambda_i}\right)\left|\ln\frac{r}{r_i}\right|\right),\qquad(1)$$

де

$$q=\frac{u_0-u_n}{\sum_{i=1}^{n}h_i/\lambda_i}\ \text{ і }\ h_i=\ln\frac{r_i}{r_{i-1}}.\qquad(2)$$

Розв'язок (1) при $r_0\le r\le r_n$ дає розподіл температури в багатошаровій циліндричній трубі, коли температура u_0, u_n внутрішньої і зовнішньої поверхонь відомі. Вираз (1) ми зможемо порівняти з розв'язком, який побудує система.

Приклад. Розглянемо задачу розподілу температури в тришаровій трубі при наступних значеннях параметрів: $\{r_i\}_{i=0}^{3}=\{3,4,5,6\}$, $\{\lambda_i\}_{i=1}^{3}=\{1,2,5\}$, $u_0=8, u_{n=3}=4$.

Р о з в ' я з а н н я . За допомогою майстра створіть 1D модель задачі

File -> New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Heat Equation(hteq) -> Stationary

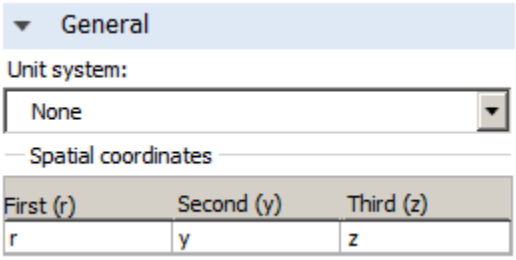
Зауваження. Можна було б створити 1D Axisymmetric модель.

В ПВ вузла **Component 1 (comp1)** відключіть контроль розмірностей фізичних величин, а в таблиці **Spatial coordinates** замість x введіть r.

Component 1 (comp1):

Unit System – <> = None

Spatial coordinates – **First** = r



Геометрія моделі складатиметься з відрізка [3,6], і на ньому двох розділювальних точок x=4 і x=5.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1:

Interval – Left endpoint, Right endpoint = 3, 6

Geometry -> **Primitives ->** Point

Geometry 1 – Point 1(pt1):

Point – r =4, 5

Точки створені для того, щоб розбити відрізок [3,6] на частини для виділення ділянок, на яких «працюватимуть» рівняння.

Перевірте, що в ПВ вузла **Geometry 1 - Form Union (fin)** в списку **Action** вибрано Form a union.

Додайте в модель ще дві «фізики» (два рівняння теплопровідності).

Physics -> **Domains ->** Classical PDEs - Heat Equation

Physics -> **Domains ->** Classical PDEs - Heat Equation

В гілці **Heat Equation (hteq)** з'являються два нових вузла **Heat Equation 2** і **Heat Equation 3**. Налаштуйте їх параметри.

Heat Equation 2:

Domain Selection – Selection, < > = Manual, 2 (*відрізок* [4,5])

Diffusion Coefficient – c = -2*r

Source Term – f = 0

Damping or Mass Coefficient – d_a = 0

Heat Equation 3:

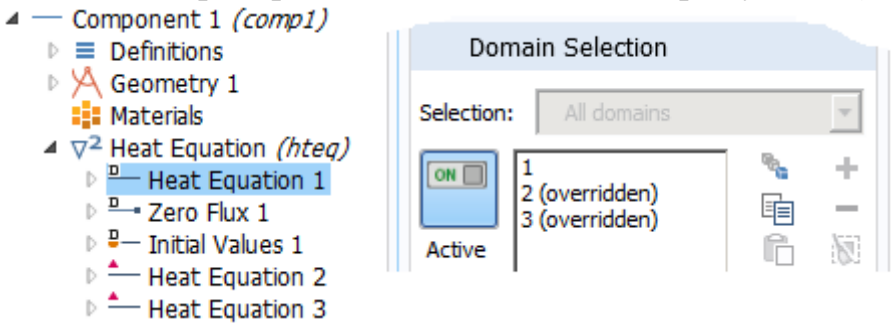
Domain Selection – Selection, < > = Manual, 3 (*відрізок* [5,6])

Diffusion Coefficient – c = -5*r

Source Term – f = 0

Damping or Mass Coefficient – d_a = 0

Відкрийте ПВ вузла **Heat Equation 1**. По виду секції вибору в розділі **Domain Selection** видно, що перше рівняння діє тільки на відрізку [3,4] (індикатор 1).



Введіть значення коефіцієнтів рівняння на цьому відрізку.

Heat Equation 1:

Diffusion Coefficient – **c** = -r

Source Term – **f** = 0

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** < > = Manual,1 (*точка x*=3)

Dirichlet Boundary Condition – **r** = 8

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection,** < > = Manual,4 (*точка x*=6)

Dirichlet Boundary Condition – **r** = 4

На внутрішніх границях реалізовуватиметься неперервність теплового поля (температури і теплового потоку) автоматично.

Побудуйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

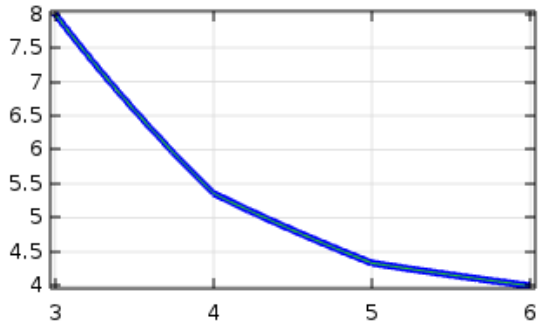
Mesh Settings – **Element Size** =Finer

Mesh -> **Build** -> Build Mesh

Виконайте обчислення задачі.

Study 1: LMB > Compute

В гілці **Results** створюється вузол **1D Plot Group 1 - Line Graph 1**, в графічній зоні якого будується криволінійна ламана, що зображає розв'язок задачі теплопровідності в тришаровому циліндрі.



В нашій задачі корисно зобразити розподіл температури по перерізу циліндра.

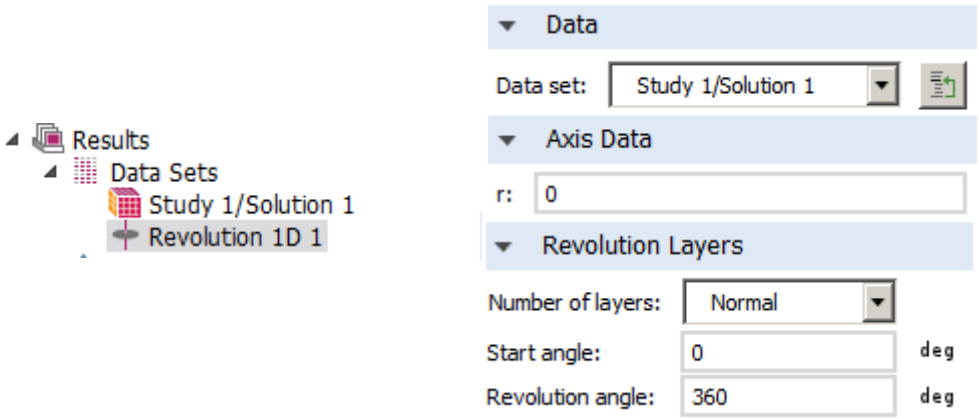
Створіть і налаштуйте джерело даних типу Revolution 1D.

Results – Data Sets: RMB > Revolution 1D

Data Sets - Revolution 1D 1 :

Data – **Data set:** =Study 1/Solution 1

Axis Data – **r:** =0



Тут в розділі **Axis Data** поле **r** визначає координату точки обертання (відрізок буде обертатися навколо точки r=0). В полях **Start angle** і **Revolution angle** задаються початковий кут і кут повороту відрізка в градусах.

Тепер створіть двовимірну графічну групу і в ній Surface графік, які використовуватимуть створене джерело даних.

Results **-> Plot Group** – 2D Plot Group

2D Plot Group 2:

Data – **Data set** = Revolution 1D 1

2D Plot Group 2 **->** Surface

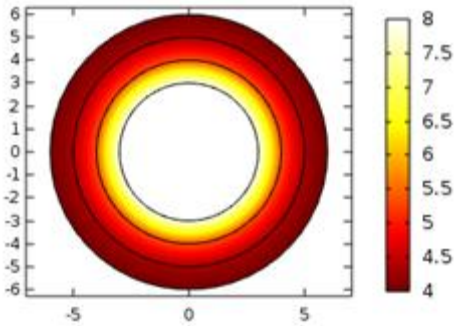
2D Plot Group 2 – Surface 1:

Expression – **Expression** = u

Coloring and Style – **Color table** = ThermalLight

LMB > Plot

Буде побудовано наступне зображення.



Для порівняння побудуйте графік «точного» розв'язку (1). Однак вирази (1) і (2) досить складні і просте введення цих формул буде вельми громіздким. Тому спочатку створіть константи, які використовуються в цих формулах. Їх зручно задати в таблиці глобальних параметрів.

В гілці **Global Definitions** згенеруйте вузол, призначений для створення параметрів.

Global Definitions: RMB > Parameters

В ПВ створеного вузла **Parameters** в таблиці параметрів в колонці **Name** вводьте імена параметрів, а в колонці **Expression** – їх значення.

Parameters

Name	Expression	Value	De
r0	3	3	
r1	4	4	
r2	5	5	
r3	6	6	
u0	8	8	
u3	4	4	
lmd1	1	1	
lmd2	2	2	
lmd3	5	5	
h1	log(r1/r0)	0.28768	
h2	log(r2/r1)	0.22314	
h3	log(r3/r2)	0.18232	
q	(u0-u3)/(h1/lmd1+h2/lm...	9.1802	

Name:

q

Expression:

(u0-u3)/(h1/lmd1+h2/lmd2+h3/lmd3)

Вводити значення можна безпосередньо в комірки таблиці або в поля **Name** і **Expression**, розташовані нижче. Це особливо корисно для довгих виразів. Наприклад, для змінної q, вираз в комірці таблиці видний не цілком. Але активувавши його рядок, в полі **Expression** ви повністю побачите вираз і зможете його відредагувати.

Задані параметри можна зберегти в текстовий файл, якщо натиснути на кнопку Save to File, розташовану під таблицею.

Залишилося створити функцію, яка буде реалізовувати обчислення за формулою (1).

Global Definitions:

Home-> **Definitions - Functions ->** Globsl - Analitic

Global Definitions - Analytic 1:

Function name = U

Definition – **Expression** = (u0+u3)/2-
q/2*(log(r/r0)/lmd1+log(r/r3)/lmd3+
(1/lmd2-1/lmd1)*abs(log(r/r1))+
(1/lmd3-1/lmd2)*abs(log(r/r2)))

Definition – **Arguments** = r

Для контролю виконайте побудову графіка функції на відрізку [3,6], задавши його кінці в розділі **Plot Parameters**.

Plot Parameters – **Low limit, Upper limit** = 3, 6

LMB> Plot

Для того, щоб функція стала доступною треба оновити розв'язок (або повністю переобчислити модель командою Compute).

Study 1: LMB > Update Solution

Тепер можна порівняти чисельний розв'язок і аналітичний. Для цього в графічну групу **1D Plot Group 1** додайте другий лінійний графік.

`1D Plot Group 1` **`-> Add Plot`** – Line Graph

1D Plot Group 1 - Line Graph 2:

`Selection` – `< > = 1,2,3` (*індикатори всіх трьох відрізків*)

`y-Axis Data` – **Expression** `= U(r)`

`x-Axis Data` – **Parameter, Expression** `= Expression, r`

Графік криволінійної ламаної, побудованої за формулою (1), співпаде з графіком розв'язку, побудованим системою.

Введення формул (1) і (2) вельми громіздке і створення аналітичної функції $U(x)$ забирає багато зусиль. На допомогу може прийти Matlab з можливістю створювати складні функції, використовуючи власні засоби програмування. Викладання мови програмування Matlab не є нашою метою. Ми будемо припускати, що читач знайомий з цією темою. Якщо ні, то пропустіть цей фрагмент тексту до кінця параграфа і поверніться до нього тоді, коли будете готові.

Другий спосіб створення функції за формулами (1) – (2).

Спочатку в Matlab створіть функцію, яка буде реалізовувати обчислення за формулами (1), (2) при $\{r_i\}_{i=0}^{3} = \{3,4,5,6\}$, $\{\lambda_i\}_{i=1}^{3} = \{1,2,5\}$, $u_0 = 8$, $u_{n=3} = 4$.

```
function u = stcylheat(r)
    % аргумент r - вектор-стовпець
     r=r';
     % вхідні дані задачі
     R=[3,4,5,6];          % радіуси шарів
     lmbd=[1,2,5];         % коеф. теплопровідності
     n=length(lmbd);
     u0=8; un=4;           % температури границь
     % Обчислення q за формулою (2)
     H = diff(log(R));
     qz=sum(H./lmbd);
     q=(u0-un)/qz;
     % реалізація формули (1)
     difInvLmbd=diff(1./lmbd);
     outR=R(2:n);          % внутрішні радіуси
     invOutR=1./outR;
     s2=difInvLmbd*abs(log(invOutR'*r));
     s0=(u0+un)/2;
     s1=log(r/R(1))/lmbd(1)+log(r/R(n+1))/lmbd(n);
     u=s0-q/2*(s1+s2);
end
```

Для перевірки працездатності функції в командному вікні Matlab виконайте наступний код

```
r=[3:0.1:6]'; u=stcylheat(r); plot(r,u,'LineWidth',3);
```

В результаті повинен бути побудований графік криволінійної ламаної, наведений раніше.

Тепер ви повинні переконатися, що в налаштуваннях системи COMSOL Multiphysics включена підтримка зовнішніх процесів. Для цього виконайте команду меню File - Preferences. Відкриється однойменне вікно. На сторінці **Security** повинен бути встановлений перемикач `Allow external processes and libraries` (встановіть його, якщо це ще не зроблено).

Потім ви повинні повідомити системі COMSOL Multiphysics, де знаходиться файл створеної функції (він повинен мати ім'я `stcylheat.m`). Найпростіше скопіювати файл цієї функції в той же каталог, в якому ви зберегли файл поточної моделі.

Тепер в COMSOL Multiphysics створіть функцію, яка буде оболонкою функції Matlab.

Global Definitions: RMB > Functions - MATLAB

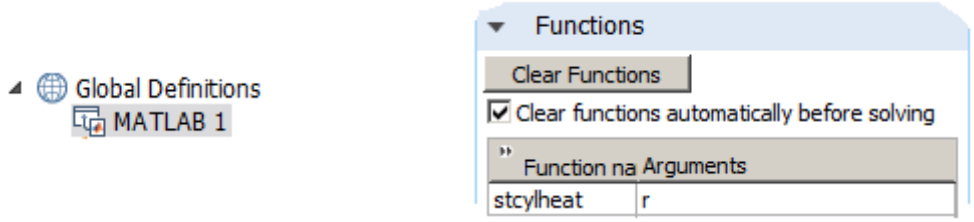
В ПВ створеного вузла в розділі **Functions** в таблиці функцій в стовпці **Function name** введіть ім'я `stcylheat`, а в стовпці **Arguments** введіть `r`.

Functions – **Function name** = `stcylheat`

– **Arguments** = `r`

Для зручності встановіть перемикач

– **Clear functions automatically before solving** = `stcylheat`



Для перевірки працездатності функції в розділі **Plot Parameters** введіть діапазон зміни аргументу і побудуйте криву

Plot Parameters – **Low limit, Upper limit** = `3, 6`

LMB> Plot

Буде побудований графік розв’язку за формулами (1), (2).

Зауваження. Якщо в Matlab функція `stcylheat` працює правильно, ви виконали всі описані налаштування COMSOL, але при виконанні команди Plot отримуєте повідомлення про помилку, то спробуйте в Matlab на початку коду функції закоментувати інструкцію `r=r'`. Можливо ваш COMSOL передає Matlab функціям аргумент у вигляді рядка (а не стовпця, як це передбачається в тілі функції `stcylheat`).

Функцію `stcylheat` можна використовувати в будь-якому місці моделі, але тільки після виконання команди Compute.

`Home` **-> Study** – Compute

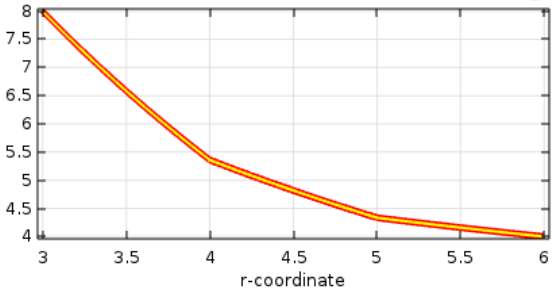
Якщо з якоїсь причини ви внесли зміни в код Matlab функції, то слід використати кнопку Clear Functions і повторно виконати команду Compute.

Графічно порівняйте чисельний розв'язок і створену функцію.

1D Plot Group 1 - Line Graph 2:

y-Axis Data – **Expression** = stcylheat(r)

Виконайте команду Plot. Вона спрацює без помилок, якщо команда Compute виконувалася після того, як вузол stcylheat був створений. Графіки криволінійних ламаних повинні накластися один на другий.



5.4.3. Задача з розривним розв'язком.

Розв'яжемо наступну контактну задачу:

$$\frac{d^2u}{dx^2}=0,\ 0<x<1;\quad \frac{d^2w}{dx^2}+w=0,\ 1<x<2; \tag{1}$$

$$u(0)=1,\ \left.\frac{du}{dx}\right|_{x=1}=\left.\frac{dw}{dx}\right|_{x=1},\ w(1)=\frac{u(1)}{1+u(1)},\ \left.\frac{dw}{dx}\right|_{x=2}=0.$$

Вона моделює дифузію в лівому шарі ($0\le x\le 1$) і дифузію з ланцюговою реакцією в правому (ланцюгова реакція еквівалентна наявності джерел дифундуючої речовини, інтенсивність яких пропорційна концентрації w).

Розв'язати поставлену задачу в Comsol можна кількома способами: з використанням двох компонентів, кожен з яких містить свою «фізику» (в нашому випадку своє рівняння), або з двома «фізиками» в одному компоненті. Тут ми наведемо три способи. У всіх випадках розв'язок першого рівняння буде реалізовано в одній «фізиці», а другого рівняння – в іншій. Особливістю реалізації моделі в перших двох способах буде організація взаємодії «фізик» за допомогою операторів зв'язку, які використовуватимуться для моделювання контактних граничних умов. В третьому розв'язку ми обійдемося без цих операторів.

Розв'язок 1. Використання двох компонентів.

Створіть файл моделі і в ньому перший компонент.

File -> New -> Model Wizard -> 1D -> Mathematics –
Classical PDEs – Poisson's Equation (poeq) -> **Stationary**

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Створіть в компоненті відрізок [0,1].

Geometry->**Primitives->** Interval

В ПВ створеного вузла **Interval 1** залиште значення параметрів і закінчить створення геометрії натисканням на кнопку Build All Objects.

На створеному відрізку генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Задайте параметри рівняння Пуасона.

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = 0

Цим на відрізку [0,1] задається рівняння $u''_{xx} = 0$.

Задайте ліву граничну умову $u(0) = 1$.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection ,** <> = Manual,1 (*точка* x=0)

Dirichlet Boundary Condition – **r** = 1

Залишилося задати другу граничну умову, але ми зробимо це пізніше.

В поточний файл додайте другий компонент.

Home -> Model - Add Component –> 1D

Його геометрія буде складатися з відрізка [1,2]. Створіть його.

Component 2 – Geometry 2:

Geometry -> **Primitives** – Interval

Component 2 – Geometry 2 - Interval 1:

Interval – **Left endpoint, Right endpoint** = 1, 2

LMB > Build All Objects

На створеному відрізку генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 2 (comp1) - Mesh 2 : LMB > Build All

Додайте в компонент «фізику» - рівняння Гельмгольца.

Component 2: RMB > Add Physics = Add Physics ->

Mathematics – Classical PDEs – Helmholtz Equation (hzeq) ->X

В дереві моделі з'явиться гілка **Component 2 - Helmholtz Equation (hzeq)**, яка буде представляти друге рівняння задачі. В її ПВ в списку **Domain Selection** - **Selection** автоматично обраний відрізок [1,2], який являє розрахункову область компонента. Тут же змініть ім'я шуканої величини.

Component 2 - Helmholtz Equation (hzeq):

Dependent Variables – **Dependent variable** = w

Задайте параметри рівняння Гельмгольца $\nabla \cdot (-c\nabla w) + a\,w = f$.

Helmholtz Equation (hzeq) - Helmholtz Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = 0

Absorption Coefficient – **a** = 1

Цим на відрізку [1,2] задається рівняння $w''_{xx} + w = 0$.

Права гранична умова $w'_x(2) = 0$ встановлюється автоматично у вузлі **Zero Flux 1**. Зауважимо, що в цьому вузлі задається також умова на лівому кінці відрізка [1,2], однак її потрібно буде перевизначити.

Залишилося пов'язати рівняння компонентів за допомогою граничних умов

$$u'_x(1) = w'_x(1), \quad w(1) = \frac{u(1)}{1 + u(1)}. \qquad (2)$$

Для цього можна використати оператори відповідності (extrusion operators). Вони виконують відображення розрахункової області компонента-джерела на геометрію компонента-адресата. Коли таке відображення (оператор) сконструйоване, то воно дозволяє в адресаті отримати доступ до шуканих величин і змінних джерела. Необхідність створення таких операторів пояснюється тим фактом, що геометрії компонентів не пов'язані, і будь-яка точка з координатами (x, y, z) з розрахунковою області одного компонента ніяк не пов'язана з точкою, що має такі ж числові значення координат в іншому компоненті.

Оператори відповідності відносяться до групи операторів зв'язку (coupling operators) і для їх створення на панелі інструментів `Definitions` є кнопка-список **`Component Coupling`**. В цьому прикладі для встановлення відповідності точки x=1 різних геометрій ми використаємо оператор тотожного відображення `Identity Mapping`. Коли він викликається в точках адресата, то його аргумент обчислюється у відповідних точках джерела.

Component 2 (comp2):

`Definitions->`**`Coupling - Component Coupling`** **- Identity Mapping**

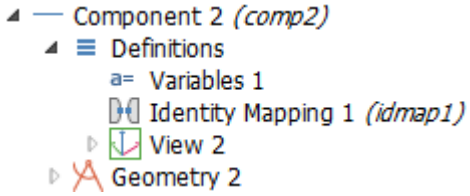
Component 2 (comp2) – Definitions - Identity Mapping 1 (idmap1):

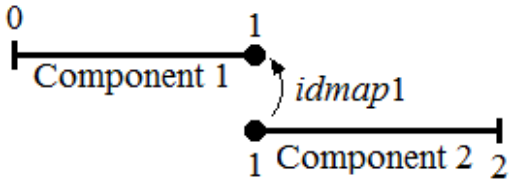
`Source Selection` – **Geometric entity level** = `Boundary`

– **Selection -** < > = `1` (*лівий край відрізка* [1,2])

`Frames` – **Destination geometry** = `Geometry 1`

Цим створюється відповідність точки x=1 другого компонента і точки x=1 першого компонента.





Використовуючи створений оператор зв'язку `idmap1`, можна задати першу граничну умову (2).

Component 1 (comp1):

`Physics` -> **`Boundaries`** – Flux/Source

Poisson's Equation (poeq) - Flux/Source 1:

`Boundary Selection` – **Selection,** < > = `2` (*правий кінець відрізка* [0,1])

`Boundary Flux/Source` – **`g`** = `-comp2.idmap1(comp2.wx)`

Умова типу **Flux/Source** має вигляд $-\boldsymbol{n}\cdot(-c\nabla u)=g-qu$, де $\boldsymbol{n}$ – одиничний вектор зовнішньої нормалі до границі (відрізка [0,1] в точці x=1). В нашому випадку $c=-1, q=0$, а значення g за допомогою оператора відповідності `idmap1` дорівнюватиме `wx` (похідній по x розв'язку `w`, яка обчислюється в точці x=1). Префікс `comp2` вказує ім'я компонента, в якому визначений оператор `idmap1` і величина `wx`. В результаті у вузлі **Flux/Source 1** реалізується умова $u'_x(1)=w'_x(1)$.

Аналогічно створюється друга гранична умова (2).

Component 1 (comp1):

Definitions -> **Component Coupling – Identity Mapping**

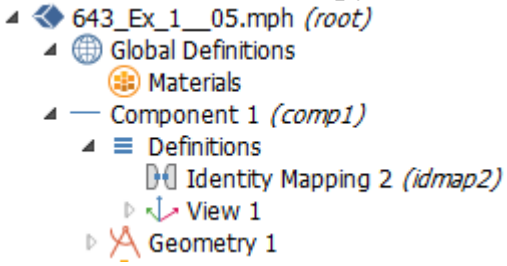
Component 1 (comp1) – Definitions - Identity Mapping 2 (idmap2):

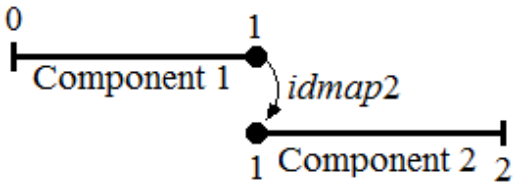
Source Selection –Geometric entity level = Boundary

Source Selection –Selection - < > = 2 (*правий край відрізка* [0,1])

Frames – Destination geometry = Geometry 2

Цим створюється (обернена до idmap1) відповідність точки x=1 першого компонента і точки x=1 другого.





За допомогою створеного оператора idmap2 в точці x=1 другого компонента можна отримати доступ до будь-якої величини в точці x=1 першого компонента. Замість громіздкого виразу, що містить звернення до цього оператора, нам буде зручно використати допоміжну змінну.

Component 2 (comp2) – Definitions: RMB > Variables

Component 2 (comp2) – Definitions – Variables 1:

Geometric Entity Selection – Geometric entity level = Boundary

– Selection = Manual,1 (*лівий кінець* [1,2])

Variables – Name, Expression = cb, comp1.idmap2(comp1.u)

– Name, Expression = nofc, cb/(1+cb)

Variables

Name	Expression
cb	comp1.idmap2(comp1.u)
nofc	cb/(1+cb)

Використовуючи створений оператор зв'язку idmap2, ми записали вираз для обчислення змінної cb. Вона представляє значення величини u в точці x=1 першого компонента, але вже визначена у другому компоненті. Змінна nofc створена для зручності.

Використовуючи створені змінні, задайте другу граничну умову (2).

Component 2 (comp2):

Physics -> **Boundaries** – Dirichlet Boundary Condition

Helmholtz Equation (hzeq) - Dirichlet Boundary Condition 1:

Boundary Selection – Selection ,<> = 1 (*ліва точка відрізка* [1,2])

Dirichlet Boundary Condition – r = nofc

Ці налаштування реалізують умову $w(1)=\dfrac{u(1)}{1+u(1)}$.

Виконайте обчислення задачі.

Study 1: LMB > Compute

Після їх завершення в гілці Results автоматично створяться дві графічні групи, в яких будуть побудовані графіки кожного з розв’язків. Також з’являться

два джерела даних **Study 1/Solution 1(1)** і **Study 1/Solution 1(2)**, що містять шукані величини першої і другої фізики. Ми хочемо об'єднати криві, побудувавши спільний графік.

Results -> **Plot Group** – 1D Plot Group

Results - 1D Plot Group 3:

1D Plot Group 3 -> **Add Plot** – Line Graph

Results - 1D Plot Group 3 - Line Graph 1:

Data – **Data set** = Study 1/Solution 1(1)

Selection – **Selection** - < > = 1

y-Axis Data – **Expression** = u

x-Axis Data – **Parameter, Expression** = Expression, x

Results - 1D Plot Group 3:

1D Plot Group 3 -> **Add Plot** – Line Graph

Results - 1D Plot Group 3 - Line Graph 2:

Data – **Data set** = Study 1/Solution 1(2)

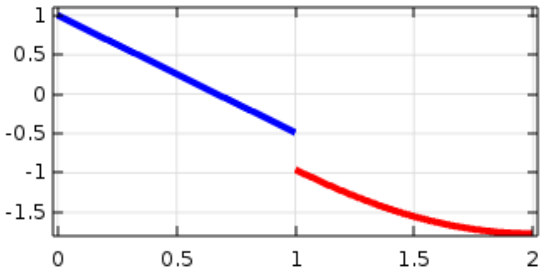
Selection – **Selection** - < > = 1

y-Axis Data – **Expression** = w

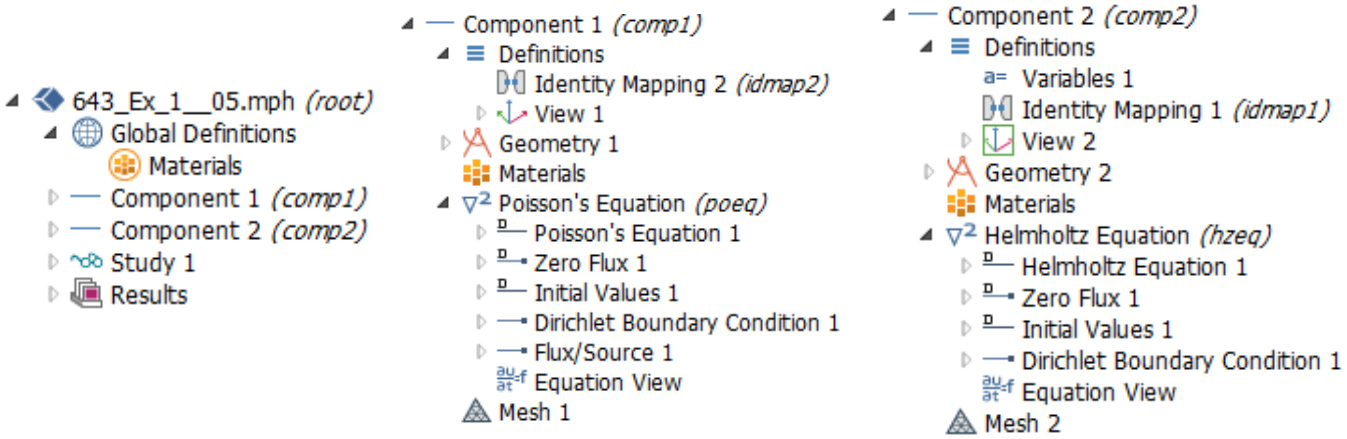
x-Axis Data – **Parameter, Expression** = Expression, x

Налаштуйте стилі оформлення кривих на свій розсуд.

LMB > Plot



На наступному рисунку наведено зображення ділянок ДМ поточного файлу.



Вправа. Для моделі поточного файлу:

1. Обчисліть значення величин $u(1), w(1), u'(1), w'(1)$ і перевірте виконання умов стикування (2).
2. У поточної задачі існує формульний розв’язок

$$u \cong 1 - 1.4887265 \cdot x$$

$$w \cong 0.73624502 \cdot \cos x - 1.6087247 \cdot \sin x$$

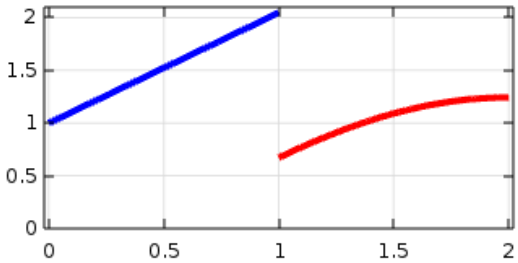
Порівняйте його з розв'язком, отриманим системою, побудувавши графік в гілці **1D Plot Group 3**.

Зауваження. Умова $w(1) = \dfrac{u(1)}{1+u(1)}$ нелінійна. Тому можна очікувати неєдинність розв'язку. Дійсно, якщо в першому компоненті **Component 1** обрати наступну початкову умову

Component 1 (comp1) – Poisson's Equation (poeq) – Initial Values 1:

`Initial Values` – **Initial value for u:** = 2

і виконати обчислення моделі, то буде отримано інший розв'язок з графіком



Розв'язок 2. Відрізняється від першого розв'язку тим, що обидві «фізики» реалізуються в спільному компоненті, розрахункова область якого складається з двох прилеглих відрізків. Перше рівняння (1) буде «працювати» на відрізку [0,1], а друге - на відрізку [1,2]. Взаємодія рівнянь за допомогою граничних умов (2) буде моделюватися з використанням операторів зв'язку (оператори зв'язку можуть встановлювати відповідність між елементами однієї геометрії).

File -> New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs – Poisson's Equation (poeq) ->
Classical PDEs – Helmholtz Equation (hzeq) + Variables = w -> Stationary

Зверніть увагу, що для розв'язку рівняння Гельмгольца ми обрали ім'я w.

Untitled.mph (root): `Unit System` – <> = None

Геометрія моделі буде складатися з двох прилеглих відрізків [0,1] і [1,2]. Створіть їх

Geometry->**Primitives->** Interval

Параметри першого відрізка співпадають зі значеннями за замовчуванням. Створіть ще один інтервал.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 2:

`Interval` – **Left endpoint, Right endpoint** = 1,2

Тепер потрібно задати спосіб комбінування створених інтервалів.

Існує два способи формування геометрії, які також визначають ім'я вузла: `Form Union` і `Form Assembly`. За замовчуванням використовується створення об'єднання (`Form Union`) при якому система виконує логічне об'єднання всіх геометричних елементів. Альтернативою є збірка (assembly). В цьому випадку система інтерпретує геометрію як колекцію об'єктів, точки яких можуть бути не пов'язані (наприклад, точка x=1 лівого відрізка не пов'язана з точкою x=1 правого). В такому випадку для сполучення величин на межах об'єктів можна використовувати спеціальні інструменти: пари (pairs) і оператори зв'язку.

В нашому прикладі ми використаємо спосіб формування геометрії за допомогою збірки (Assembly).

Geometry 1 - Form Union(fin)

Form Union/Assembly – **Action** = Form an assembly

Form Union/Assembly – **Create pairs** = False (*зняти прапорець*)

LMB > Build All

Зверніть увагу, що назва вузла **Form Union** в ДМ змінилася на **Form Assembly**. Крім того, точка x=1 відрізка [0,1] отримала індикатор з номером 2. Ця ж точка, але як частина відрізка [1,2], отримала індикатор з номером 3. Для того, щоб на рисунках розрахункової області можна було спостерігати номери/індикатори елементів геометрії, виконайте налаштування

Component 1 – Definitions - View 1:

View – **Show geometry labels** = True

Генеруйте СкЕ сітку.

Component 1 – Mesh 1:

Mesh Settings –**Element size** = Extra fine

LMB > Build All

В ПВ першої «фізики» визначте відрізок, на якому буде «працювати» рівняння, і задайте його параметри.

Poisson's Equation (poeq):

Domain Selection – **Selection** = Manual

– < > = 1 (*відрізок* [0,1] *залишаємо, а інший видаляємо*)

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = 0

Цим задається рівняння $u''_{xx} = 0$ на відрізку [0,1].

Для задання лівої граничної $u(0) = 1$ умови виконайте інструкції:

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** - <> = 1 (*лівий кінець відрізка* [0,1])

Dirichlet Boundary Condition – **r** = 1

Другу граничну умову ми поставимо пізніше.

Налаштуйте параметри рівняння Гельмгольца $\nabla \cdot (-c\nabla w) + a\,w = f$.

Helmholtz Equation (hzeq):

Domain Selection – **Selection** = Manual

Domain Selection – < > = 2 (*залишаємо тільки відрізок* [1,2])

Helmholtz Equation (hzeq) - Helmholtz Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = 0

Absorption Coefficient – **a** = 1

Цим визначається рівняння $w''_{xx} + w = 0$ на відрізку [1,2].

Права гранична умова $w'_x(2) = 0$ задається автоматично в вузлі **Helmholtz Equation (hzeq) - Zero Flux 1**.

Для задання контактної граничної умови (2) створіть оператори лінійної відповідності.

`Definitions` **-> Coupling-Component Couplings-Linear Extrusion**

Definitions - Linear Extrusion 1 (linext1):

Source Selection – **Geometric entity level** = `Boundary`

– **Selection -** < > = 3 (т. x=1 *відрізка* [1,2])

Використовуйте прокручування колеса миші для переміщення фокуса по черзі між індикаторами 2 і 3.

Source Vertices – **Source vertex 1** = 3 (т. x=1 *відрізка* [1,2])

Destination Vertices – **Destination vertex 1** = 2 (т. x=1 *відрізка* [0,1])

Цим створюється відповідність $1_{[1,2]} \to 1_{[0,1]}$ точки x=1 відрізка [1,2] і точки x=1 відрізка [0,1].

Створіть оператор відповідності в зворотну сторону.

`Definitions` **-> Coupling-Component Couplings-Linear Extrusion**

Definitions - Linear Extrusion 2 (linext2):

Source Selection – **Geometric entity level** = `Boundary`

– **Selection -** < > = 2 (т. x=1 *відрізка* [0,1])

Source Vertices – **Source vertex 1** = 2 (т. x=1 *відрізка* [0,1])

Destination Vertices – **Destination vertex 1** = 3 (т. x=1 *відрізка* [1,2])

Задайте «внутрішню» граничну умову $u'_x(1) = w'_x(1)$.

Poisson's Equation (poeq):

`Physics` -> **Boundary - Boundaries** – Flux/Source

Poisson's Equation (poeq) - Flux/Source 1:

Boundary Selection –**Selection -** < > = 2 (т. x=1 *відрізка* [0,1])

Boundary Flux/Source – **g** = `-linext1(wx)`

Задайте «внутрішню» граничну умову $w(1) = \dfrac{u(1)}{1+u(1)}$.

Helmholtz Equation (hzeq):

`Physics` -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Helmholtz Equation (hzeq) - Dirichlet Boundary Condition 1:

Source Selection –**Selection -** < > = 3 (т. x=1 *відрізка* [1,2])

Dirichlet Boundary Condition – **r** = `linext2(u)/(1+linext2(u))`

Виконайте обчислення моделі.

`Home` -> Study - Compute

Після їх завершення в гілці **Results** автоматично створюються дві одновимірні графічні групи, в яких побудовані графіки розв'язків `u` і `w`. Ми хочемо побудувати один спільний графік. Для цього можна створити локальну (в компоненті) змінну `U`, яка на різних відрізках буде збігатися з розв'язками `u` і `w`, а потім побудувати її графік.

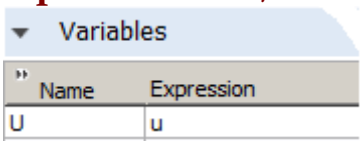
`Definitions` **->Variables** – Local Variables

Definitions – Variables 1:

Geometric Entity Selection – **Geometric entity level** = `Domain`

– **Selection** = Manual,1 (*відрізок* [0,1])

Variables – **Name, Expression** = U, u



Цим ми повідомили, що величина U на відрізку [0,1] збігається з u. Ще раз виконайте команду

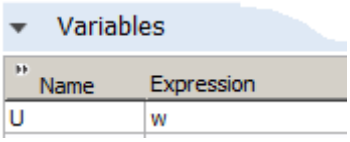
Definitions -> **Variables** – Local Variables

Definitions – Variables 2:

Geometric Entity Selection – **Geometric entity level** = Domain

– **Selection** = Manual,2 (*відрізок* [1,2])

Variables – **Name, Expression** = U, w



Цим ми повідомляємо, що величина U на відрізку [1,2] збігається з w.

Для того, щоб змінна U стала доступною, оновіть обчислення.

Study 1: RMB > Update Solution

Тепер можна будувати графік змінної U.

Results -> **Plot Group** – 1D Plot Group

Results - 1D Plot Group 3:

Data – **Data set** = Study 1/Solution 1

1D Plot Group 3 -> **Add Plot** – Line Graph

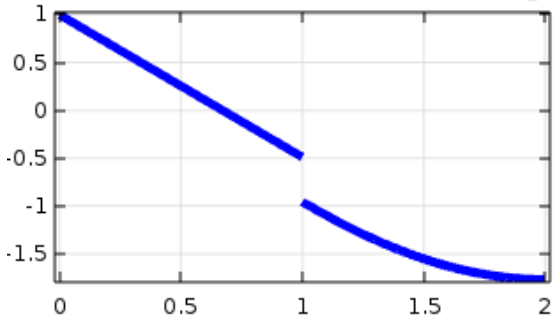
1D Plot Group 3 - Line Graph 1:

Selection – **Selection** = Manual,1,2 (*обидва відрізка* [0,1] і [1,2])

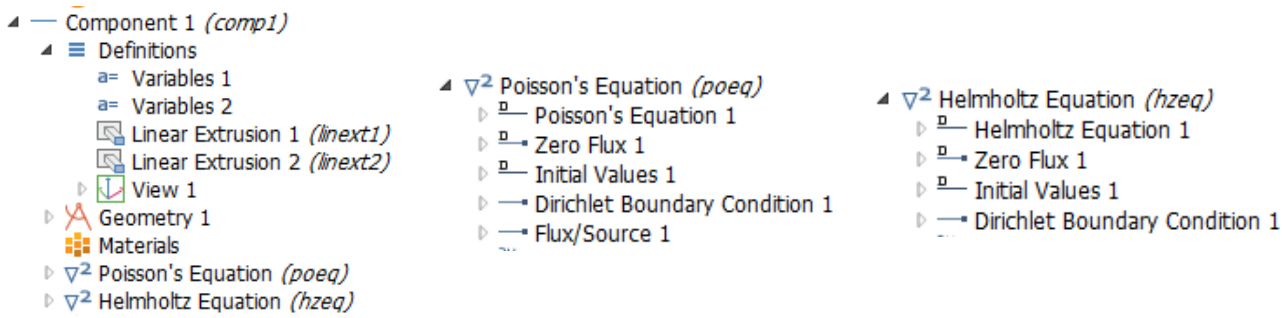
y-Axis Data – **Expression** = U

x-Axis Data – **Parameter, Expression** = Expression, x

Налаштуйте стиль графіка і виконайте команду Plot. В результаті буде побудований той же графік, що і при першому способі створення моделі.



На наступному рисунку наведено зображення ділянок ДМ поточного файлу.



Збережіть файл, наприклад, з ім'ям `DiscontDiffusion_1.mph`, але не закривайте його.

Зауваження. Для ототожнення точок різних геометрій можна застосувати оператор тотожного відображення `Identity Mapping`. Коли він викликається в точках адресата, то його аргумент обчислюється у відповідних точках джерела. Покажемо, які зміни потрібно внести в модель, щоб використати такий оператор.

Відкрийте файл останнього прикладу і деактивуйте (або видаліть) вузли операторів лінійної відповідності.

Definitions - Linear Extrusion 1 (linext1) : RMB > Disable

Definitions - Linear Extrusion 2 (linext2) : RMB > Disable

Тепер створіть оператори тотожного відображення.

`Definitions` -> **`Component Coupling`** - **Identity Mapping**

Definitions - Identity Mapping 1 (idmap1):

`Source Selection` – **Geometric entity level** = `Boundary`

– **Selection -** < > = 3 (*лівий край відрізка* [1,2])

Використовуйте прокручування колеса миші для переміщення фокуса по черзі між індикаторами 2 і 3.

Цим створюється відповідність $1_{[1,2]} \to 1_{[0,1]}$ (насправді точка x=1 відрізка [1,2] відповідатиме будь-якій точці, де оператор `idmap1` використовуватиметься).

`Definitions` -> **`Component Coupling`** - **Identity Mapping**

Definitions - Identity Mapping 2 (idmap2):

`Source Selection` –**Geometric entity level** = `Boundary`

`Source Selection` –**Selection -** < > = 2 (*правий край відрізка* [0,1])

Останнім створюється відповідність $1_{[0,1]} \to 1_{[1,2]}$.

Після цього у вузлах граничних умов імена операторів `linext1` і `linext2` слід замінити іменами операторів тотожного відображення `idmap1` і `idmap2`.

Poisson's Equation (poeq) - Flux/Source 1:

`Boundary Flux/Source` – **g** = `-idmap1(wx)`

Helmholtz Equation (hzeq) - Dirichlet Boundary Condition 1:

`Dirichlet Boundary Condition` – **r** = `idmap2(u)/(1+idmap2(u))`

Виконайте обчислення і отримайте ті ж графіки, що і раніше.

Розв'язок 3. Простішою є модель, в якій в одному компоненті створюється дві фізики і не використовуються оператори зв'язку. Для цього відрізки геометрії поєднуються операцією Union, а внутрішні граничні умови (2)

реалізуються у вузлах `Flux/Source` і `Dirichlet Boundary Condition` виразами без використання операторів зв'язку.

Відкрийте файл `DiscontDiffusion_1.mph`, створений у другому розв'язку, і змініть в ньому спосіб поєднання відрізків.

Geometry 1 - Form an assembly (fin)

`Form Union/Assembly` – Action = `Form a union`

LMB > Build All

Назва вузла **Form Assembly** в ДМ зміниться на **Form Union**. Точка x=1 отримає єдиний індикатор 2 (при Form Assembly вона позначалася двома індикаторами 2 і 3).

Деактивуйте або видаліть вузли операторів зв'язку (в принципі їх можна не чіпати).

Definitions - Linear Extrusion 1 (linext1) : RMB > Disable

Definitions - Linear Extrusion 2 (linext2) : RMB > Disable

У вузлах `Flux/Source` і `Dirichlet Boundary Condition` запишіть ті ж самі формули без використання операторів зв'язку. Не забудьте знову вибрати точки, бо зміна їх нумерації призвела до загублення індикаторів в секціях вибору цих вузлів.

Poisson's Equation (poeq) - Flux/Source 1:

`Boundary Selection` –Selection - < > = `2` (т. x=1)

`Boundary Flux/Source` – g = `-wx`

Helmholtz Equation (hzeq) - Dirichlet Boundary Condition 1:

`Source Selection` –Selection - < > = `2` (т. x=1)

`Dirichlet Boundary Condition` – r = `u/(1+u)`

Виконайте обчислення моделі

`Home` -> Study - Compute

і в гілці **1D Plot Group 3** отримайте той же графік, що і раніше.

Вправа. Розв'яжіть наступну контактну крайову задачу

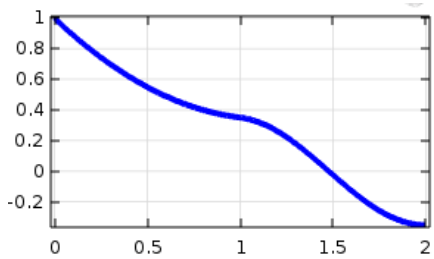
$$\frac{d^2u}{dx^2}=1,\ \ 0<x<1;\quad \frac{d^2w}{dx^2}+9w=0,\ \ 1<x<2;$$

з умовами

$$u(0)=1,\ \left.\frac{du}{dx}\right|_{x=1}=\left.\frac{dw}{dx}\right|_{x=1},\ \ w(1)=u(1),\ \ \left.\frac{dw}{dx}\right|_{x=2}=0.$$

1. Використовуючи два компонента і два оператора відповідності (лінійної відповідності або тотожного відображення).
2. Використовуючи дві «фізики» в одному компоненті і два оператора відповідності.
3. Використовуючи два рівняння на збірці (`Assembly`) і умову `Continuity`.
4. Використовуючи дві фізики в одному компоненті без використання операторів відповідності.

5. Записавши задачу одним рівнянням $u''_{xx} + K(x)u = F(x)$ з розривними коефіцієнтом $K(x) = \begin{cases} 0\,, x < 1 \\ 9\,, x \geq 1 \end{cases}$ і правою частиною $F(x) = \begin{cases} 1\,, x < 1 \\ 0\,, x \geq 1 \end{cases}$.



5.4.4. Прогин вільно опертої балки.

Рівняння рівноваги зігнутої балки має вигляд [7]

$$\frac{d^2}{d\,x^2}\left(E\,J\,\frac{d^2}{d\,x^2}u\right) = q(x), \tag{1}$$

де $q(x)$ –зовнішня (поперечна) сила, розрахована на одиницю довжини балки, E, J – модуль пружності матеріалу балки і момент інерції її перерізу. Функція $u(x)$ являє вертикальне зміщення точок осі балки, які мали до навантаження горизонтальну координату x.

Введемо позначення $M = -E\,J\,\frac{d^2u}{d\,x^2}$. Функцію $M(x)$ називають згинальним моментом балки в точці x. Тоді рівняння (1) набуває виду $\frac{d^2M}{d\,x^2} = -q$, і ЗДР 4-го порядку можна записати у вигляді системи двох диференціальних рівнянь 2 - го порядку.

$$\begin{cases} E\,J\,\dfrac{d^2u}{d\,x^2} = -M(x) \\ \dfrac{d^2M}{d\,x^2} = -q(x) \end{cases} \tag{2}$$

В матричному записі система має вигляд

$$\frac{d}{d\,x}\left(\begin{pmatrix} E\,J & 0 \\ 0 & 1 \end{pmatrix}\frac{d}{d\,x}\begin{pmatrix} u \\ M \end{pmatrix}\right) + \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} u \\ M \end{pmatrix} = \begin{pmatrix} 0 \\ -q \end{pmatrix}. \tag{3}$$

Приклад. Розв’язати крайову задачу прогину вільно опертої балки, що знаходиться під дією кусково-сталої навантаги q. Для визначеності покласти $L = 1$ (довжина балки), $E = 1, J = 1$,

$$q = \begin{cases} q_1\,, 0 \leq x < 1/3 \\ q_2\,, 1/3 \leq x < 2/3, \\ q_3\,, 2/3 \leq x < 1 \end{cases} \text{ де } q_1 = -1, q_2 = 0.95, q_3 = -1.$$

Знаки величин q_i вказують на те, що зусилля q_1 і q_3 діють вниз, а q_2 - вгору.

Р о з в ’ я з а н н я . За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D ->

Mathematics - PDE Interfaces - Coefficient Form PDE (c)

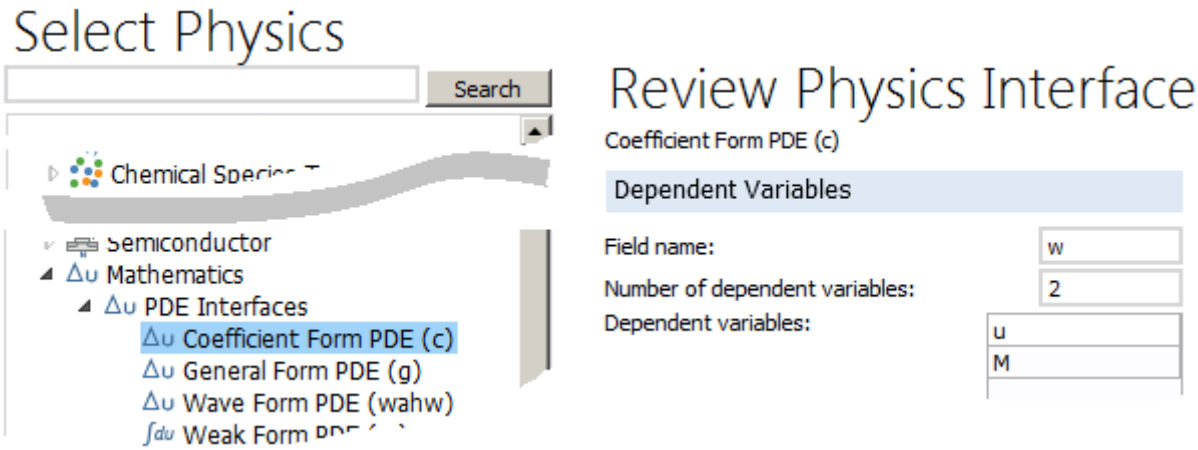
Після виконання команди Add не поспішайте переходити до наступного вікна. Спочатку праворуч в панелі `Review Physics Interface` в полі **Number of dependent variables** задайте кількість невідомих функцій `2`, і нижче введіть їх імена `u`, `M`. В полі **Field name** введіть `w`.

`Review Physics Interface:`

`Dependent Variables` – **Field name** = `w`

– **Number of dependent variables** = `2`

– **`Dependent variables`** = `u, M`



Натисніть кнопку Study і виберіть тип дослідження `Stationary`.

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Задайте параметри моделі.

Global Definitions: RMB > Parameters

В ПВ створеного вузла **Parameters** в таблиці параметрів в колонці **`Name`** вводьте імена параметрів, а в колонці **`Expression`** – їх значення.

Parameters

Name	Expression	Value
q1	-1	-1
q2	0.95	0.95
q3	-1	-1
E	1	1
J	1	1

Геометрія моделі складатиметься з відрізка [0,1] і двох розділювальних точок.

`Geometry` -> **`Primitives`** -> Interval

Geometry 1 - Interval 1 (i1) :

Interval – **Number of intervals** = `Many`

Interval – **Points** = `0,1/3,2/3,1`

LMB > Build All Objects

Побудуйте СкЕ сітку.

Mesh 1:

`Mesh Settings` –**Element size** = `Finer`

LMB > Build All

Задайте систему рівнянь. Перевірте, що в ПВ вузла **Coefficient Form PDE (c) - Coefficient Form PDE 1** в секції вибору **Domain Selection** обрано весь відрізок (стоять індикатори 1, 2, 3). В розділі **Equation** записано загальний (векторний) вид системи рівнянь

$$e_a \frac{\partial^2 \boldsymbol{u}}{\partial t^2} + d_a \frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \left(-c \nabla \boldsymbol{u} - \alpha \boldsymbol{u} + \gamma\right) + \beta \nabla \boldsymbol{u} + \mathrm{a}\, \boldsymbol{u} = \boldsymbol{f},$$

де $\mathbf{u} = \begin{pmatrix} u \\ M \end{pmatrix}$ є вектором розв'язків, $\nabla = \frac{\partial}{\partial x}$ – оператор диференціювання. Коефіцієнти системи є матрицями розміру 2×2 і задаються в полях нижньої частини ПВ. Відповідно до (3) вводимо наступні значення (вільний член $\boldsymbol{f}$ буде заданий пізніше).

Diffusion Coefficient – $\mathbf{c} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$

Absorption Coefficient – $\mathbf{a} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$

Source Term – $\mathbf{f} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

Для інших коефіцієнтів вводимо $e_a = d_a = \alpha = \beta = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$, $\gamma = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

З курсу опору матеріалів [7] відомо, що для вільно опертої балки довжини L граничні умови мають вигляд

$$u(0) = 0, M(0) = 0,\ u(L) = 0, M(L) = 0.$$

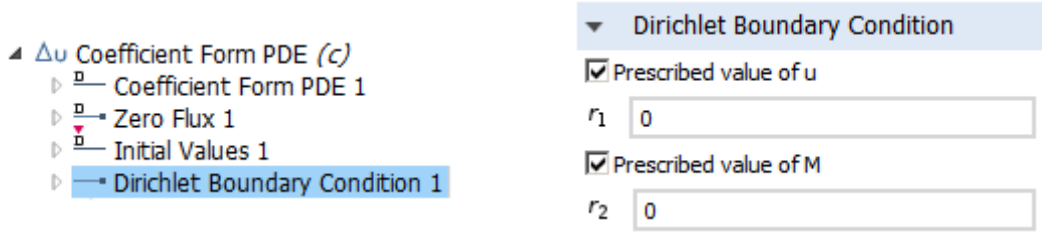
Задайте ці умови.

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1,4 (т. x=0 і т. x=1)

Dirichlet Boundary Condition – $\mathbf{r_1}, \mathbf{r_2}$ = 0,0



Додайте в «фізику» три вузли типу Source, на яких задайте значення зовнішнього розподіленого зусилля.

Physics -> **Domain - Domains** – Coefficient Form PDE – Source

Coefficient Form PDE (c) - Coefficient Form PDE 1 – Source 1:

Domain Selection – **Selection** = Manual,1 (*відрізок* [0,1/3])

Source Term – $\mathbf{f} = \begin{pmatrix} 0 \\ -q_1 \end{pmatrix}$

Coefficient Form PDE 1 – Source 1: RMB > Duplicate

Coefficient Form PDE 1 – Source 2:

Domain Selection – **Selection** = Manual,2 (*відрізок* [1/3,2/3])

Source Term – **f** = $\begin{pmatrix} 0 \\ -q_2 \end{pmatrix}$

Coefficient Form PDE 1 – Source 2: RMB > Duplicate

Coefficient Form PDE 1 – Source 3:

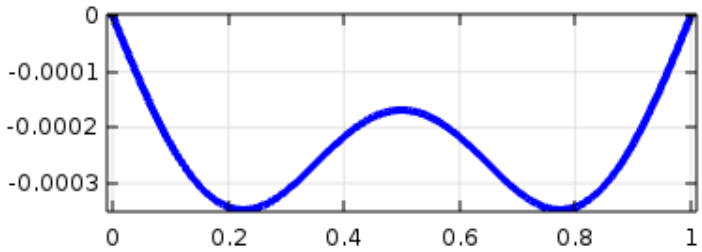
Domain Selection – **Selection** = Manual,3 (*відрізок* [2/3,1])

Source Term – **f** = $\begin{pmatrix} 0 \\ -q_3 \end{pmatrix}$

Виконайте обчислення.

Home **-> Study** - Compute

В результаті буде побудована наступна крива (графік нейтральної осі балки).



Створіть графік згинального моменту M, обчисленого системою. Крім того, побудуйте також графіки виразу −uxx (при E=1, J=1 маємо $M = -u''_{xx}$).

Results - 1D Plot Group 1: RMB > Duplicate

1D Plot Group 2 - Line Graph 1 :

y-Axis Data – **Expression** = M

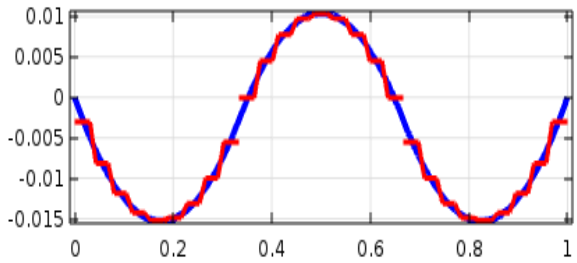
1D Plot Group 2 - Line Graph 1 : RMB > Duplicate

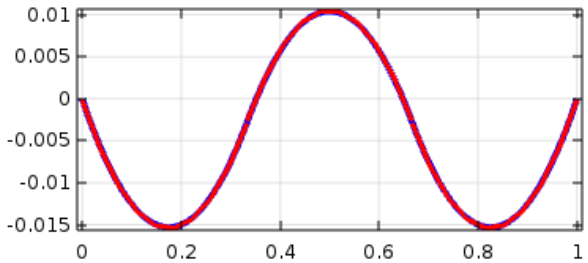
1D Plot Group 2 - Line Graph 2 :

y-Axis Data – **Expression** = −uxx

Coloring and style – **Color, Width** = Red, 2

LMB > Plot (*наступний рисунок ліворуч*)





Як бачите, графік другої похідної uxx є кусково-сталим. Його можна покращити, якщо застосувати вищій порядок функцій форми. За замовчуванням встановлено другий (Quadratic) порядок. Встановіть третій (Cubic) порядок цих функцій.

Coefficient Form PDE (c):

Discretization – **Element order** = Cubic

Заново обчисліть модель і перебудуйте графіки в гілці **1D Plot Group 2**. Ви отримаєте графік, зображений на попередньому рисунку праворуч.

Побудуємо 2D графік подовжніх напружень (див. п. 2.3.4). Відповідно до гіпотези плоских перерізів, точки балки, що мали до деформування координати (x, y), переходять в точки з координатами (X,Y)

$$X = x - \frac{u'_x(x)\ y}{\sqrt{1+(u'_x(x))^2}} \quad \text{і} \quad Y = u(x) + \frac{y}{\sqrt{1+(u'_x(x))^2}}. \qquad (4)$$

Якщо поперечний переріз балки прямокутний, то поздовжні напруження визначаються за формулою

$$\sigma_x(x, y) = \frac{M(x)\cdot y}{J}, \qquad (5)$$

де y - вертикальна координата точок балки ($y = 0$ відповідає нейтральній осі). Величина $\sigma_x(x, y)$ буде використовуватися для функціонального забарвлення «поверхні балки».

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Його геометрія складатиметься з одного прямокутника.

`Geometry` **-> Primitives** – Rectangle

Component 2 - Geometry 2 - Rectangle1:

Size – **Width, Height** = `1,0.2` (*два поля*)

Position – **Base, x, y** = `Corner,0,-0.1` (*три поля*)

LMB > Build All Objects

Зауваження. Формули (4), (5) вважають, що вісь балки співпадає з віссю X.

Побудуйте в області прямокутну сітку.

`Mesh` **-> Generators** – Mapped

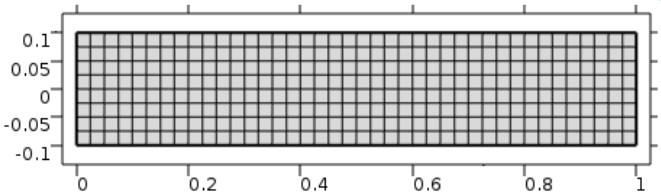
Задайте розміри клітинок сітки.

Component 2 - Mesh 1 – Size:

Element Size – **Custom**= `True`

Element Size Parameters – **Maximum element size**= `0.025`

LMB > Build All



Щоб на геометрії іншого компонента використовувати розв'язок з першого, в ньому (першому компоненті) потрібно створити оператор зв'язку.

Component 1:

`Definitions->` **Coupling-Component Coupling - General Extrusion**

Component 1 (comp1) – Definitions - General Extrusion 1 (genext1):

Source Selection–**Selection – Manual,** < > = `Manual,1,2,3` (*відрізок* [0,1])

Destination Map – **x-expression** = `x`

Знову застосуйте команду = Compute. Ніяких додаткових обчислень в моделі не виконуватиметься, але тепер в гілці **Results - Data Sets** ми зможемо створити

джерело даних, яке буде використовувати двовимірну геометрію другого компонента.

Results – Data Sets - Study 1/Solution 1: RMB > Duplicate

Data Sets - Study 1/Solution 1 (2):

Solution – **Component**= Component (comp2)

В гілку **Results** додайте графічну групу 2D Plot Group і створіть в ній вузол Surface.

Home **-> Results -** Add Plot Group **->** 2D Plot Group

2D Plot Group 3:

Data– **Data set** = Sudy 1/Solution 1(2)

Plot Settings – **Plot data set edges** = False

2D Plot Group 3 **-> Add Plot -** Surface

2D Plot Group 3 – Surface 1:

Data– **Data set** = From parent

Expression – **Expression** = 1

Coloring and Style – **Coloring, Color** =Uniform, Black

– **Wireframe** = True

У вузлі **Surface 1** створіть підвузол деформації.

2D Plot Group 3 – Surface 1: RMB > Deformation

2D Plot Group 3 – Surface 1 – Deformation 1 :

Expression – **x component** =

-comp1.genext1(ux)/sqrt(1+comp1.genext1(ux)^2)*y

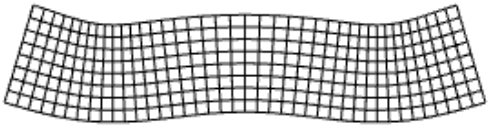
Expression – **y component** =

comp1.genext1(u)+y/sqrt(1+comp1.genext1(ux)^2)-y

Введені тут вирази відповідають вектору зсуву $(\Delta x, \Delta y)$, який визначається за допомогою формули (4) як $\Delta x = X - x, \Delta y = Y - y$.

LMB > Plot

В результаті графічна зона вузла **Surface 1** буде містити деформоване каркасне зображення балки.



В графічній групі **2D Plot Group 2** створіть ще один вузол Surface, який ми використаємо для функціонального забарвлення зони балки.

2D Plot Group 3 – Surface 1: RMB > Duplicate

2D Plot Group 3 – Surface 2:

Expression – **Expression** = comp1.genext1(M)*y

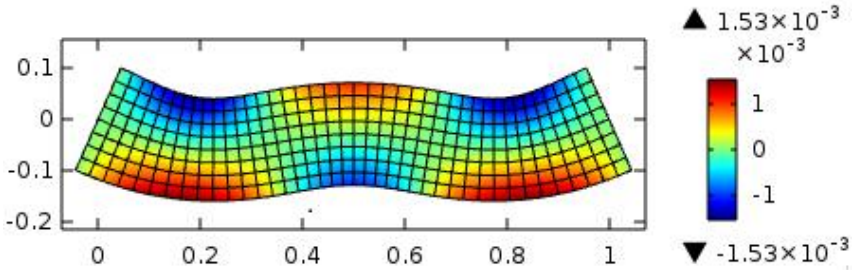
Введений вираз відповідає формулі (5).

Coloring and Style – **Coloring, Color** =Color table, Rainbow

– **Color legend** = True

– **Wireframe** = False

LMB > Plot



На останньому рисунку деформоване зображення балки з функціональним забарвленням поєднане з її каркасним зображенням.

Якщо всі три зовнішні впливи обрати рівними, то ми отримаємо задачу про прогин вільно опертої балки, яка знаходиться під дією рівномірно розподіленого навантаження. Рівняння осі такої балки має вигляд

$$u = \frac{q}{24\,E\,J}\left(L^3 x - 2Lx^3 + x^4\right).$$

Змінимо значення параметрів нашої задачі $q_1 = q_2 = q_3 = -1$, розв'яжемо її в COMSOL, і порівняємо графіки.

Parameters:

`Parameters` – **Name, Expression** = `q2 ,-1`

`Home` **`->`** **`Study`** - Compute

Results - 1D Plot Group 1: RMB > Duplicate

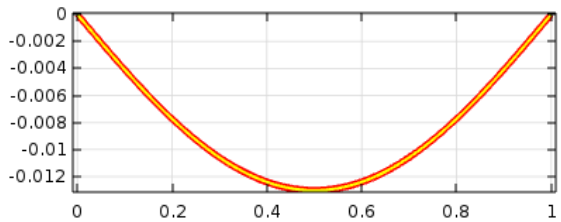
1D Plot Group 4 - Line Graph 1 : RMB > Duplicate

1D Plot Group 4 - Line Graph 2 :

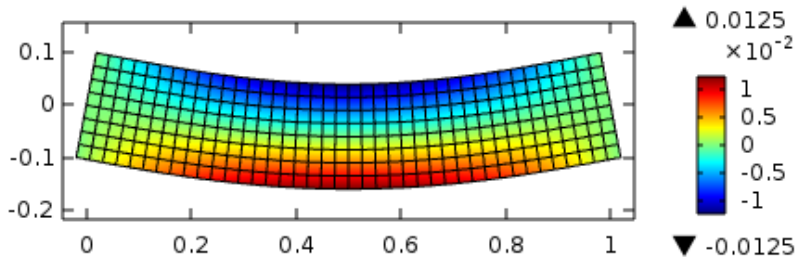
`y-Axis Data` – **Expression** = `-(x-2*x^3+x^4)/24`

Змініть самостійно стиль ламаної, щоб криві відрізнялися.

LMB > Plot



Звичайно, що зміниться графік у вузлі **2D Plot Group 3**.



5.4.5. Затиснена на краях балка під дією зосереджених сил.

Рівняння згину балки, яка знаходиться під дією зовнішніх навантаг наведено в попередньому параграфі (формула (1)). Для зосереджених навантаг потрібно покласти $q(x)=\sum_{i=1}^{n-1} q_i \cdot \delta(x-x_i)$, де q_i зусилля, прикладені в точках x_i, $\delta(x)$ – дельта-функції Дірака. Рівняння (1), так само як і в попередньому параграфі, зводиться до системи (2) (або (3)). Тому послідовність дій при побудові моделі повторює попередній приклад, за винятком того, що ми повинні змоделювати точкові зусилля.

Розв'яжемо цю задачу, використовуючи дві фізики в одному компоненті. Тобто кожне з рівнянь системи (2) попереднього параграфа розглядатимемо як окрему «фізику». Ще одна особливість, яка відрізняє цю задачу від попередньої, полягає в використанні граничних умов затискання

$$u(0)=0, u'(0)=0,\ u(L)=0, u'(L)=0.$$

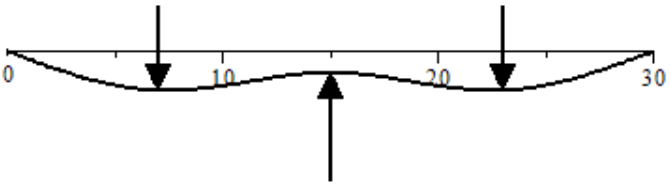
Як наслідок, в системі (2) для першого рівняння є 4 граничні умови, а для другого – жодної. Цю колізію ми подолаємо, використовуючи `Global ODEs and DAEs` інтерфейс.

Для досліджуваної задачі відомий формульний розв'язок

$$u(x)=\frac{1}{12EJ}\sum_{i=1}^{n} G(x,x_i)\ , \qquad (1)$$

де $G(x,x_i)=(x-x_i)^3+\frac{6}{L^2}x^2(L-x_i)^2x_i-\frac{2}{L^3}x^3(L-x_i)^2(L+2x_i)+|x-x_i|^3$, і ми порівняємо його з розв'язком, отриманим системою.

Приклад. Покладемо $E=2, J=54$ і розв'яжемо крайову задачу для балки довжиною L=30 і висотою 6, яка знаходиться під дією зосереджених навантаг $\{q_i\}=\{-1, 1, -1\}$, прикладених в точках $\{x_i\}=\{7, 15, 23\}$. Наступний рисунок пояснює постановку задачі.



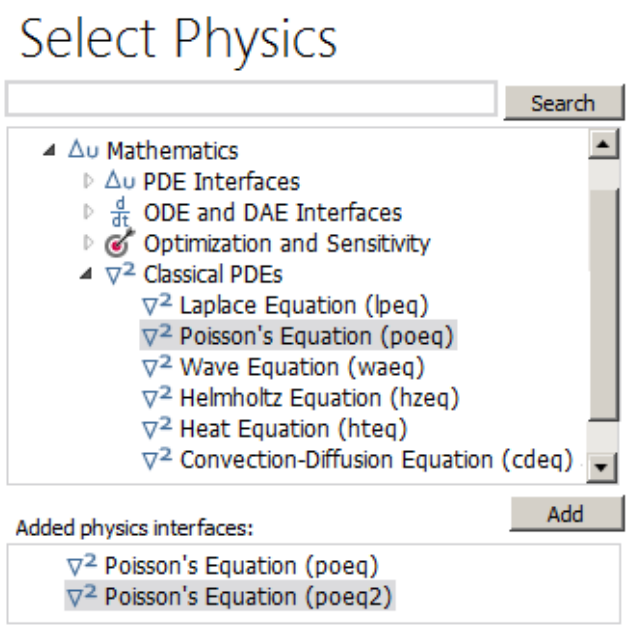
Розв'язання. Почніть створення моделі наступною послідовністю команд

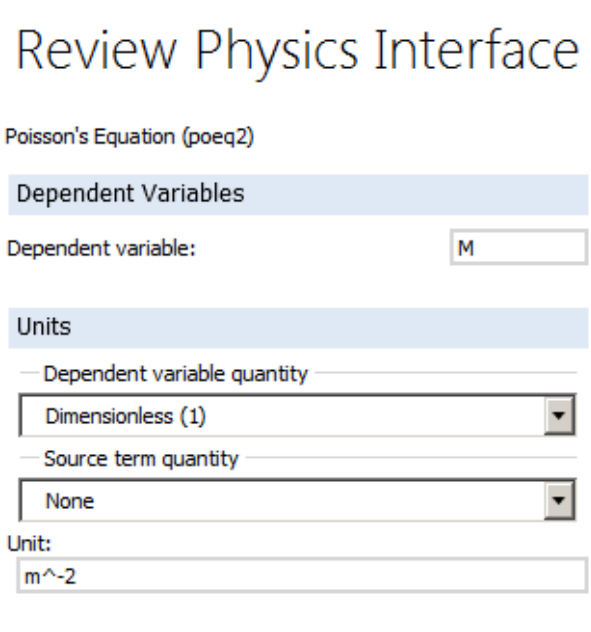
File -> New -> Model Wizard -> 1D

На етапі вибору «фізики» у вікні `Select Physics` за допомогою кнопки Add виберіть відразу дві «фізики»:

`Mathematics - Classical PDEs - Poisson's Equation (poeq)`
`Mathematics - Classical PDEs - Poisson's Equation (poeq)`

При виборі другого рівняння Пуассона задайте ім'я `M` розв'язку, потім виконайте команду Study.





На етапі `Select Study` виберіть `Stationary` (стаціонарний тип дослідження) і натисніть кнопку Done.

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): **`Unit System`** – <> = `None`

Задайте параметри моделі.

Global Definitions: RMB > Parameters

В ПВ створеного вузла **Parameters** в таблиці параметрів в колонці **`Name`** вводьте імена параметрів, а в колонці **`Expression`** – їх значення.

Parameters

Name	Expression
q1	-1
q2	1
E	2
J	54
L	30

Розрахункова область моделі складається з відрізка [0,30] і точок, в яких будуть прикладатися зовнішні навантаги.

`Geometry` -> **`Primitives`** -> Interval

Geometry 1 - Interval 1 (i1) :

Interval – **Number of intervals** = `Many`

Interval – **Points** = `0,7,15,23,30`

LMB > Build All Objects

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – **Element Size** = `Extra Fine`

LMB> Build All

В ПВ вузла першої «фізики» **Poisson's Equation (poeq)** перевірте, що в секції вибору **`Domain Selection`** обрано весь відрізок (стоять індикатори 1, 2, 3, 4). Потім задайте параметри рівняння.

Poisson's Equation (poeq) - Poisson's Equation 1:

`Diffusion Coefficient` – **c** = `E*J`

Source Term – **f** = M

Цим задається перше рівняння $-E\,J\,u''_{xx} = M(x)$.

Задайте нульові граничні умови для функції u.

Poisson's Equation (poeq):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1,5 (т. x=0 і т. x=30)

Dirichlet Boundary Condition – **r** = 0

Задайте параметри другого рівняння Пуассона,.

Poisson's Equation 2 (poeq2) - Poisson's Equation 1:

Перевірте, що в секції вибору **Domain Selection** обрано весь відрізок (стоять індикатори 1, 2, 3, 4)

Diffusion Coefficient – **c** = 1

Source Term – **f** = 0

Цим визначається рівняння $M''_{xx} = 0$. Праву частину потрібного рівняння $M''_{xx} = -\sum_{i=1}^{n-1} f_i \cdot \delta(x - x_i)$ задайте за допомогою вузлів типу Flux/Source.

Physics -> **Boundaries** – Flux/Source

Heat Equation (hteq) - Flux/Source 1:

Boundary Selection – **Selection,** < > = Manual,2,4 (т. x=7 і т. x=23)

Зосереджені навантаги в точках x=7 і x=23 однакові , і їх можна задати в одному вузлі Flux/Source.

Boundary Flux/Source – **g** =q1

Physics -> **Boundaries** – Flux/Source

Heat Equation (hteq) - Flux/Source 2:

Boundary Selection – **Selection,** < > = Manual,3 (т. x=15)

Boundary Flux/Source – **g** =q2

Граничних умов для величини M в нашій задачі немає, а є умови $u'(0) = 0$, $u'(L) = 0$. Щоб їх реалізувати в кінцевих точках прикладаються якісь (поки що невідомі) згинальні моменти $m_1 = M(0)$ і $m_2 = M(L)$. Вважатимемо їх додатковими невідомими задачі, які, до того ж, «прив'язані» до кінцевих точок $x = 0$ і $x = L$.

В COMSOL скалярну величину в точці можна реалізувати за допомогою зондової змінної domain point probe. Вона містить значення «полевої» величини в точці. В нашій моделі будуть потрібні зонди, які відслідковуватимуть значення похідних $u'_x(0)$, $u'_x(L)$. Створіть ці зонди.

Definitions : RMB > Probes – Domain Point Probe

Definitions - Domain Point Probe 1:

Point Selection – **Coordinate** = 0 (*точка* x=0)

Definitions - Domain Point Probe 1 – Point Probe Expression 1 (ppb1):

Variable name = Z1

Expression – **Expression** = ux

Definitions : RMB > Probes – Domain Point Probe

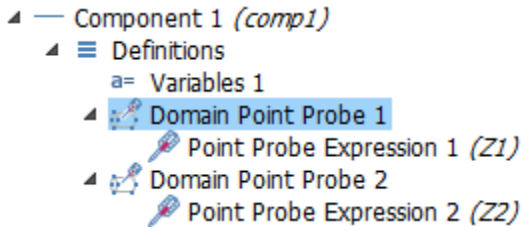
Definitions - Domain Point Probe 2:

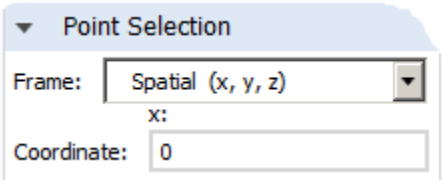
Point Selection – **Coordinate** = L (*точка x*=30)

Definitions - Domain Point Probe 1 – Point Probe Expression 1 (ppb2):

Variable name = Z2

Expression – **Expression** = ux





В результаті, змінні Z1 і Z2 співпадатимуть зі значеннями похідної u'_x в кінцевих точках відрізка [0, L].

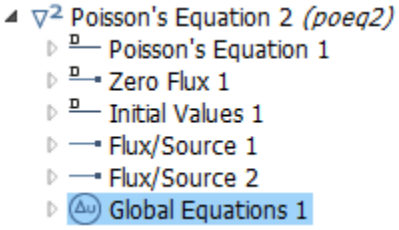
Додайте в модель вузол Global Equations, в якому вкажіть m1, m2 в якості невідомих, а замість рівнянь напишіть імена зондових змінних.

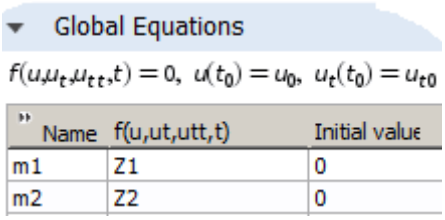
Physics -> **Global** – Global Equations

Poisson's Equation 2 (poeq2) - Global Equations 1:

Global Equations – **Name, f(u,ut,utt,t)** = m1,Z1

Global Equations – **Name, f(u,ut,utt,t)** = m2,Z2





Розуміти запис в таблиці рівнянь потрібно наступним чином. Задача буде розв'язуватися відносно невідомих m1 і m2 (граничні значення M) одночасно з величинами u та M, доки вирази в комірках **f(u,ut,utt,t)** не будуть дорівнювати нулю, тобто доки зондові змінні Z1 і Z2 (а вони співпадають зі значеннями похідних $u'_x(0)$ і $u'_x(L)$) не стануть нулями.

Тепер можна додавати в фізику другого рівняння граничні умові Діріхле $M(0)=m_1$ і $M(L)=m_2$ так, як ніби величини m_1 і m_2 відомі.

Poisson's Equation 2 (poeq2):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation 2 (poeq2) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1 (т. *x*=0)

Dirichlet Boundary Condition – **r** = m1

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation 2 (poeq2) - Dirichlet Boundary Condition 2:

Boundary Selection – **Selection** = Manual,5 (т. *x*=30)

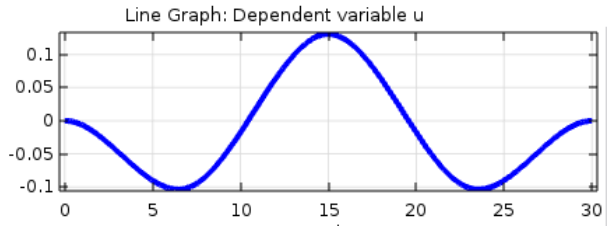
Dirichlet Boundary Condition – **r** = m2

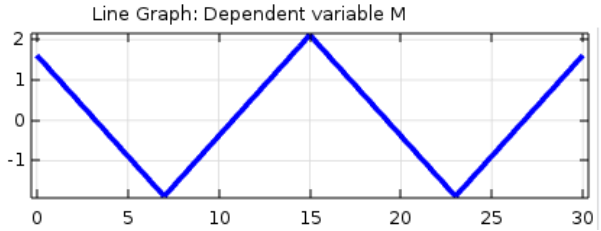
В результаті в моделі створено дві шукані полеві величини u,M, та дві скалярні невідомі m1 і m2.

Виконайте обчислення.

`Home -> Study` - Compute

Після їх завершення будуть створені дві гілки **1D Plot Group 1** і **1D Plot Group 2**, в яких побудовані графіки нейтральної осі `u` балки (наступний рисунок ліворуч) і згинального моменту `M` (наступний графік праворуч).





В гілці **Probe Plot Group 3** буде створено графіки зондів. Оскільки у нас модель статична, то вони є просто двома точками з майже нульовими ординатами (величинами `m1` і `m2`). Їх значення можна дізнатися з автоматично створеної таблиці **Probe Table 1**.

Gradient of u, x component, Point Probe Expression 1	Gradient of u, x component, Point Probe Expression 2
-1.0408E-17	-8.6736E-19

Зауваження. Замість зондових змінних можна створити звичайні змінні, які обчислюють значення похідної u'_x в кінцевих точках відрізка [0,30]. Для цього деактивуйте вузли зондів і створіть в компоненті вузол локальних змінних.

Definitions - Domain Point Probe 1: RMB > Disable

Definitions - Domain Point Probe 2: RMB > Disable

Component 1 - Definitions: RMB > Variables

Component 1 - Definitions – Variables 1:

`Geometric Entity Selection` – **Geometric entity level** = `Entire model`

Останнім визначається зона дії змінних - по всій моделі.

`Variables` – **Name, Expression** = `Z1, at0(0,ux)`

– **Name, Expression** = `Z2, at0(L,ux)`

Переобчисліть модель.

Study 1: RMB > Update Solution

і отримайте ті ж самі графіки.

Порівняємо графіки формульного розв'язку (1) і наближеного, отриманого системою. При обраних координатах $\{x_i\}=\{7,15,23\}$ прикладання зосереджених навантаг і їх значеннях $\{q_i\}=\{-1,1,-1\}$, після деяких спрощень формула аналітичного розв'язку (1) набуває вигляду

$$u(x)=\frac{1}{12960}\left(91350-10590\,x+353x^2-10|x-7|^3+10|x-15|^3-10|x-23|^3\right).$$

Реалізуємо її в графічній групі **1D Plot Group 1**.

1D Plot Group 1 - Line Graph 1: RMB > Duplicate

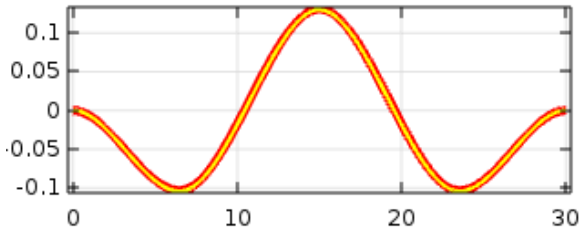
1D Plot Group 1 - Line Graph 2:

`y-Axis Data` – **Expression** =`(91350-10590*x+353*x^2-10*abs(x-7)^3+10*abs(x-15)^3-10*abs(x-23)^3)/12960`

x-Axis Data – **Parameter, Expression** = Expression,x

Coloring and Style – **Color, Width** = Yellow, 2

LMB > Plot



Як бачите, графіки аналітичного і наближеного розв'язків співпадають.

Для тренування читача побудуємо графік подовжніх напружень на поверхні тривимірного зображення балки (див. п. 2.3.4). Координати точок зігнутої балки обчислюватимемо за формулами (4) попереднього параграфа, а для функціонального забарвлення її поверхні використаємо формулу (5).

Якщо СкЕ сітку на поверхні балки будувати непотрібно, то графічне зображення зігнутої балки можна створити, використовуючи джерела даних Grid 2D або Grid 3D. Але спочатку, для передачі значень величини u в джерело даних іншої розмірності, в першому компоненті створіть оператор зв'язку.

Component 1:

Definitions-> **Coupling-Component Coupling - General Extrusion**

Component 1 (comp1) – Definitions - General Extrusion 1 (genext1):

Source Selection–**Selection**–**Manual,**<>=Manual,1,2,3,4 (*відрізок* [0,30])

Destination Map – **x-expression** = x

Щоб оператор став працездатним, оновіть розв'язок.

Study 1: RMB > Update Solution

Створіть джерело даних Grid 3D, яке має розміри тривимірної балки.

Results -> Data Set - More Data Sets -> Grid - Grid 3D

Data Sets – Grid 3D 1:

Data – **Source, Data set** = Data set,Study 1/Solution 1 (*два поля*)

В розділі **Parameter Bounds** в полях **Minimum** і **Maximum** введіть діапазони зміни координат $0 \le x \le L$, $-3 \le y \le 3$, $-1 \le z \le 1$.

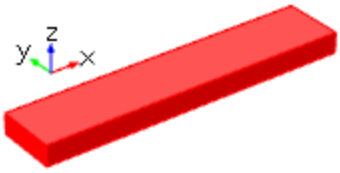
Paramemter Bounds –

– **First parameter** – **Name, Minimum, Maximum** = x,0,L

– **Second parameter** – **Name, Minimum, Maximum** = y,-3,3

– **Third parameter** – **Name, Minimum, Maximum** = z,-1,1

LMB > Plot



Створіть групу тривимірних графіків і в ній вузол Surface.

Results -> Plot Group - 3D Plot Group

3D Plot Group 4 :

Data – **Data set** = Grid 3D 1

Plot Settings – **Plot Data set edges** = False

Color Legend – **Show legends** = True

– **Show maximum and minimum values** = True

RMB > Surface

3D Plot Group 4 - Surface 1 :

Data – **Data set** = From parent

Expression – **Expression** = genext1(M)*y/J

Coloring and Style – **Coloring, Color** =Color table, Rainbow

– **Color legend** = True

У вузлі **Surface 1** створіть підвузол деформації.

2D Plot Group 2 – Surface 1: RMB > Deformation

2D Plot Group 2 – Surface 1 – Deformation 1 :

Expression – **x component** =

-comp1.genext1(ux)/sqrt(1+comp1.genext1(ux)^2)*y

Expression – **y component** =

comp1.genext1(u)+y/sqrt(1+comp1.genext1(ux)^2)-y

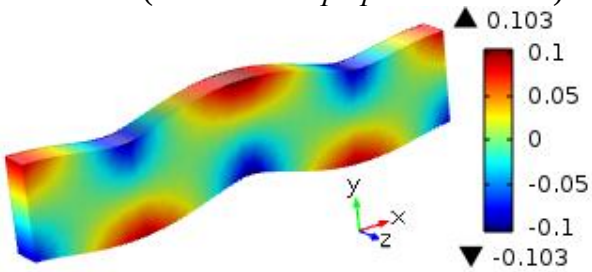
– **z component** = 0

Введені тут вирази відповідають вектору зсуву $(\Delta x, \Delta y, 0)$, де $\Delta x = X - x$, $\Delta y = Y - y$, а нові координати точок (X, Y) визначаються за допомогою формули (4) попереднього параграфа.

Scale – **Scale factor** = 7

LMB > Plot

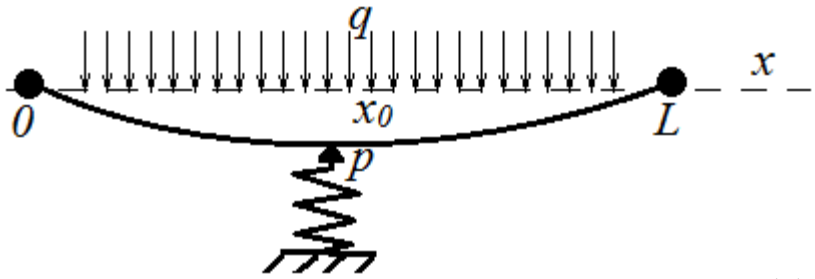
LMB > Show Grid (*кнопка над графічним вікном*)



Вправа. Дослідіть викладену задачу, використовуючи підхід попереднього параграфа, розв'язуючи систему рівнянь за допомогою **Coefficient Form PDE** інтерфейсу. Зосереджені сили моделюйте з використанням вузлів типу Flux/Source.

5.4.6. **Струна з пружним упором .**

Диференціальне рівняння рівноваги струни має вигляд $u''_{xx} = -f(x)$, де $f(x) = q(x)/T$, T - натяг струни, $q(x)$ - зовнішня (поперечна) сила, розрахована на одиницю довжини. Функція $u(x)$ являє вертикальне зміщення точок струни, які мали до навантаження горизонтальну координату x. В точці струни x_0 прикладена зосереджена навантага, створювана кінцем пружини жорсткістю k.



У стані рівноваги без зовнішнього навантаження, коли $q(x)=0$, пружина торкається струни і знаходиться в «розслабленому» стані. Після навантаження струна прогинається і тисне на пружину, яка протидіє з силою $p=-k\cdot u(x_0)/H$, пропорційною зміщенню її кінця $u(x_0)$ (вертикальне зміщення точки x_0 з'єднання струни і кінця пружини), H – довжина пружини.

Математична постановка задачі визначення форми струни, яка закріплена на кінцях, має вигляд

$$u''_{xx}=q(x)+p\cdot\delta(x-x_0),$$

$$p=-k\cdot u(x_0)/H,\ u(0)=0, u(L)=0,$$

де $\delta(x)$ - дельта–функция Дірака.

Приклад 1. Розв'язати модельну задачу, коли натяг струни $T=1$, довжина $L=2$, $x_0=1$, H=1, і на струну діє розрахована на одиницю довжини навантага $q(x)=-1$.

Розв'язок 1. Перший спосіб дослідження цієї задачі в COMSOL полягатиме в розв'язанні одновимірного рівняння Пуассона $u''_{xx}=q(x)$, в гілку фізики якого буде додано вузол типу `Flux/Source` з невідомою зовнішньою зосередженою силою $p=-k\cdot u(x_0)/H$. Вона буде обчислюватися додаванням скалярної невідомої p в допоміжному вузлі `Global Equations`.

За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs - Poisson's Equation (poeq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

В гілці **Global Definitions** створіть вузол параметрів.

Global Definitions: RMB > Parameters

В ПВ створеного вузла **Parameters** в таблиці параметрів в колонці **Name** введіть імена параметрів, а в колонці **Expression** - їх значення.

Parameters

Name	Expression	Value
q	-1	-1
k	8	8

Геометрія моделі складається з відрізка [0,2] і точки $x_0=1$. Створіть їх.

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1):

`Interval` – **Left endpoint, Right endpoint** = `0,2`

Geometry -> **Primitives->** Point

Geometry 1 – Point 1:

Point – **x** = 1

LMB > Build All Objects

Точка потрібна для того, щоб згодом ви могли вказати місце, в якому буде діяти зосереджена сила.

На створеному відрізку генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

В ПВ вузла **Poisson's Equation (poeq)** перевірте, що в секції вибору стоять цифри 1 і 2 (індикатори ділянок відрізка). Потім задайте параметри рівняння.

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = 1

Source Term – **f** = q

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = 1,3 (*індикатори кінців відрізка*)

Dirichlet Boundary Condition – **r** = 0

Збережіть модель в файл з ім'ям, наприклад, StringWithSpring_01.mph, щоб згодом можна було б продовжити роботу з цього місця. Тепер збережіть файл з іншим ім'ям, наприклад, StringWithSpring_02.mph, і продовжить роботу з ним.

Додайте в гілку фізики **Poisson's Equation (poeq)** вузол типу Flux/Source, в якому буде задаватися зосереджене навантаження.

Physics -> **Boundary - Boundaries** – Flux/Source

Poisson's Equation (poeq) - Flux/Source 1:

Boundary Selection – **Selection,** < > = Manual,2 (*точка x*=1)

Boundary Flux/Source – **g** = p

Тут p невідома реакція пружини.

Тепер потрібно створити оператор, за допомогою якого в додаткове рівняння буде передаватися значення прогину струни $u(x_0)$ в точці x_0. Для цього можна використати інтегральний оператор зв'язку. Оператори зв'язку застосовуються для передачі значень між моделями різних фізичних процесів. Характер передаваних значень може бути різним. Зокрема, це можуть бути деякі інтегральні характеристики моделі або значення розрахункових величин в точках. Оператори зв'язку (Coupling Operators) створюються в гілці **Definitions** компонента, з якого вибираються дані. Після цього їх можна використовувати на будь-якому етапі моделювання в будь-якому місці поточного або іншого компонента

Для створення операторів зв'язку призначена кнопка – список Component Coupling панелі інструментів Definitions,. Вона дозволяє конструювати оператори зв'язку різних типів. Зокрема є можливість створювати оператор інтегрування (Integration).

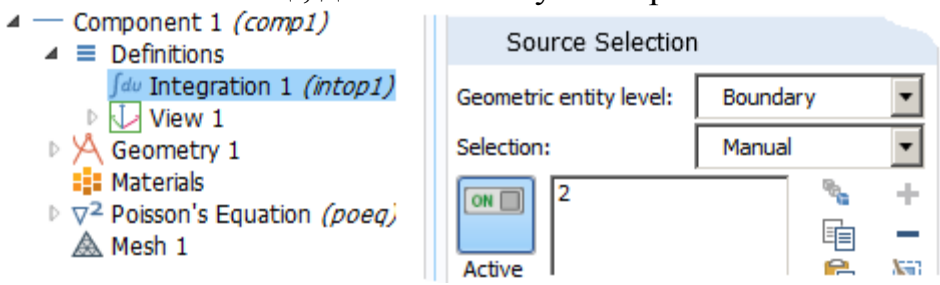
Definitions **-> Coupling - Component Coupling** – Integration

Component 1 - Definitions - Integration 1 (intop1):

Source Selection – **Geometric entity level** = Boundary

– **Selection -** < > = 2 (*точка* x=1)

Ім'я оператора інтегрування intop1 вказується в полі **Operator name**. Оператор intop1(expr) інтегрує будь-який вираз expr по вказаній в ньому зоні. Якщо в якості області використовується точка, то оператор повертає значення виразу expr в цій точці. Наприклад, intop1(u) буде обчислювати значення величини u в точці, для якої він був створений.



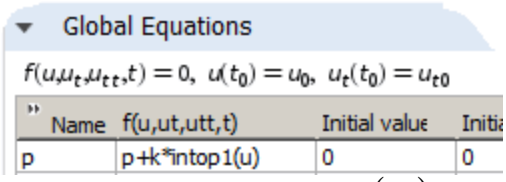
Отже оператор інтегрування (в нашому випадку оператор вибору значення величин в точці $x_0 = 1$) створено. Додайте в модель рівняння відносно невідомої p, в якому застосуйте цей оператор,

Physics **-> Global** – Global Equations

Poisson's Equation (poeq) - Global Equations 1:

Global Equations – **Name** = p

Global Equations – **f(u,ut,utt,t)** = p+k*intop1(u)

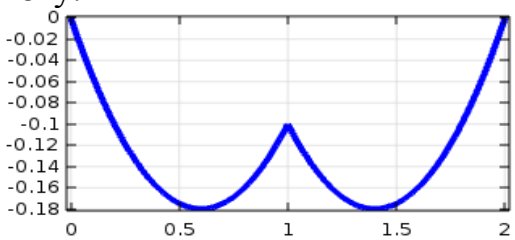


Цим ми додали в задачу рівняння $p + k \cdot u(x_0) = 0$, і створили додаткову невідому p. Нагадаємо, це ім'я використовується у вузлі **Flux/Source 1** в якості точкової навантаги.

Нарешті обчисліть модель.

Home->Compute

В графічній зоні вузла **Results - 1D Plot Group 1 – Line Graph 1** буде побудовано графік розв'язку.



Крім того, буде створено вузол **Results - Derived Values - State variable p**, призначений для друкування величини зосередженої навантаги p.

Results - Derived Values - State variable p:

LMB: Evaluate – New Table

Буде створена таблиця з однією коміркою і результатом обчислення p=0.8.

State variable p
0.80000

Обчисліть вертикальне зміщення точки x_0 струни.

`Results` **`->`** **`Derived Values`** – Point Evaluation

Results - Derived Values - Point Evaluation 1:

`Selection` – **Selection** = `2` (*точка x*=1)

`Expression` – **Expression** = `u`

LMB: Evaluate – Table 1

В таблицю буде додано комірку з результатом обчислення $u(x_0) = -0.1$.

State variable p	Dependent variable u, Point: 2
0.80000	-0.100000

Обчисліть реакції R_i вузлів закріплення. З точністю до знака вони збігаються з похідною u'_x в цих точках ($T = 1$ і $R_i = \mp T\, d\, u / d\, x_{\text{на кінці } i}$).

`Results` **`->`** **`Derived Values`** – Point Evaluation

Results - Derived Values - Point Evaluation 2:

`Selection` – **Selection** = `3` (*правий кінець відрізка* [0,2])

`Expression` – **Expression** = `ux`

LMB: Evaluate – Table 1

В таблицю буде додано комірку з новим результатом обчислення.

State variable p	Dependent variable u, Point: 2	Gradient of u, x component, Point: 3
0.80000	-0.100000	0.60000

Отже реакція кінців відрізка $R_1 = R_3 = 0.6$. Їх сума разом з реакцією пружини $p = 0.8$ співпадає (з протилежним знаком) з сумарною силою $\int_0^2 q(x)\, d\, x = -2$, яка діє на струну зверху.

Збережіть поточний файл моделі.

Вправа. Перевірте отриманий розв'язок шляхом розв'язання задачі

$$u''_{xx} = q(x) + p_0 \cdot \delta(x - x_0),\ u(0) = 0,\ u(L) = 0,$$

де $p_0 = 0.8$. Побудуйте графік розв'язку і порівняйте його з отриманим в поточному прикладі.

Розв'язок 2. В другому способі дослідження цієї задачі ми уникнемо створення додаткової шуканої змінної p. Різниця з першим розв'язком починається з налаштувань вузла `Flux/Source`. Загрузіть файл `StringWithSpring_01.mph`. В ньому побудована майже вся потрібна модель, але без зосередженої сили, створюваною пружиною.

Додайте в гілку фізики **Poisson's Equation (poeq)** вузол типу `Flux/Source`, який буде моделювати зосереджене навантаження.

`Physics` -> **`Boundary`** **`-`** **`Boundaries`** – Flux/Source

Poisson's Equation (poeq) - Flux/Source 1:

`Boundary Selection` – **Selection,** < > = `Manual,2` (*точка x*=1)

`Boundary Flux/Source` – **g** = `-k*u` (*сила реакції пружини*)

Тут `u` представляє вертикальне зміщення точки `x=1` струни, яке співпадає з вертикальним зміщенням кінця пружини, а добуток `-k*u` представляє реакцію пружини і одночасно величину зосередженої навантаги в точці.

Обчисліть модель.

`Home` -> Compute

В графічній зоні вузла **Results - 1D Plot Group 1 – Line Graph 1** буде побудовано криву таку ж саму, як і в першому розв'язку.

В цьому місці доречно перевірити працездатність операторів `up(expr)` і `down(expr)`. Вони використовуються при роботі з величинами на примітивах, які мають розмірність на одиницю меншу ніж геометрія моделі. Оператор `up(expr)` вибирає значення `expr` з боку зовнішньої нормалі до примітива. Оператор `down` повертає значення з іншого боку. Для розривної величини `expr` значення цих операторів різняться з боків стрибка, в той час як просте застосування ідентифікатора `expr` повертає середню величину. Якщо один з боків розташований зовні геометрії або змінна не визначена на якомусь з боків, то там ці оператори повертають 0.

Обчисліть значення похідної $u'_x(1)$ в точці $x_0 = 1$.

`Results` -> **`Derived Values`** – Point Evaluation

Results - Derived Values - Point Evaluation 1:

`Selection` – **Selection** = 2 (*точка x*=1)

`Expression` – **Expression** = `ux`

LMB: Evaluate – New Table

Gradient of u, x component, Point: 2
6.6613E-16

Як бачите, надруковано практично нульове значення. Але насправді, з графіка розв'язку видно, що похідна в точці $x_0 = 1$ зазнає розриву, а не дорівнює нулю.

Обчисліть значення $u'_x(1-0) = \lim\limits_{\varepsilon \to +0} u'_x(1-\varepsilon)$, тобто границю похідної при наближенні до точки $x_0 = 1$ зліва. Для цього поміняйте вираз у вузлі **Point Evaluation 1**.

Derived Values - Point Evaluation 1:

`Expression` – **Expression** = `down(ux)`

LMB: Evaluate – Table 1

Gradient of u, x component, Point: 2	down(ux), Point: 2
6.6613E-16	0.40000

Отже $u'_x(1-0) = 0.4$.

Обчисліть значення «правої похідної» $u'_x(1+0) = \lim\limits_{\varepsilon \to +0} u'_x(1+\varepsilon)$, змінивши вираз у вузлі **Point Evaluation 1**.

Derived Values - Point Evaluation 1:

`Expression` – **Expression** = `up(ux)`

LMB: Evaluate – Table 1

Gradient of u, x component, Point: 2	down(ux), Point: 2	up(ux), Point: 2
6.6613E-16	0.40000	-0.40000

Як бачите, $u'_x(1+0)=-0.4$.

Фізичний зміст має сила стиснення пружини. Вона протидіє натягу струни, який прикладений зліва і справа від точки з'єднання. Тому реакція R пружини дорівнює $R=T\cdot\left(u'_x(1-0)+\left|u'_x(1+0)\right|\right)$. Обчислимо її, змінивши вираз у вузлі **Point Evaluation 1** і, враховуючи, що T=1.

Derived Values - Point Evaluation 1:

Expression – **Expression** = down(ux)-up(ux)

LMB: Evaluate – Table 1

Gradient of u, x component, Point: 2	down(ux), Point: 2	up(ux), Point: 2	down(ux)-up(ux), Point: 2
6.6613E-16	0.40000	-0.40000	0.80000

Збережіть модель в файл `StringWithSpring_03.mph`.

Трохи ускладнимо задачу, додавши в попередню модель ще одну пружину в точці x=0.5.

Приклад 2. Загрузіть файл `StringWithSpring_01.mph`. В ньому вже побудована значна частина моделі.

Додайте в геометрію нову точку x=0.5, створивши її в уже існуючому вузлі **Point 1**.

Geometry 1 – Point 1:

Point – **x** =1,0.5 (*в існуючий вузол додаємо точку x*=0.5)

Додайте параметр k2, який буде представляти жорсткість другої пружини.

Parameters:

Parameters – **Name, Expression** =k2,12 (*третій рядок таблиці*)

В фізику **Poisson's Equation (poeq)** додайте два вузли типу Flux/Source. Перший – для точки x=1, другий – для точки x=0.5.

Physics -> **Boundary - Boundaries** – Flux/Source

Poisson's Equation (poeq) - Flux/Source 1:

Boundary Selection – **Selection,** < > = Manual,3 (*точка x*=1)

Boundary Flux/Source – **g** = -k*u (*реакція першої пружини*)

Physics -> **Boundary - Boundaries** – Flux/Source

Poisson's Equation (poeq) - Flux/Source 2:

Boundary Selection – **Selection,** < > = Manual,2 (*точка x*=0.5)

Boundary Flux/Source – **g** = -k2*u (*реакція другої пружини*)

Додайте в компонент дві змінні R1 і R2, які будуть містити вертикальні реакції кінців струни.

Definitions -> **Variables** – Local Variables

Definitions - Variables 1:

Geometric Entity Selection – **Geometric entity level** = Boundary
– **Selection** = Manual,1 (т. x=0)

Variables – **Name, Expression** = R1,-ux (*введення в таблиці*)

Definitions -> **Variables** – Local Variables

Definitions - Variables 2:

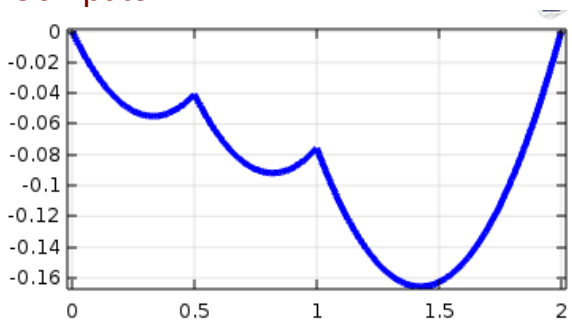
Geometric Entity Selection – **Geometric entity level** = Boundary

– **Selection** = `Manual,4` (т. x=2)

`Variables` – **Name, Expression** = `R2,ux` (*введення в таблиці)*

Обчисліть модель і отримайте криву, яка зображатиме форму струни.

`Home` **`->`** **`Study`** - Compute



Надрукуйте силу реакції кінців струни.

`Results` **`->`** **`Derived Values`** – Point Evaluation

Results - Derived Values - Point Evaluation 1:

`Selection` – **Selection** = `Manual, 1` (*лівий кінець струни*)

`Expression` – **Expression** = `R1`

LMB: Evaluate

Буде створена таблиця **Table 1** з однією коміркою і результатом обчислення `R1=0.33140`.

`Results ->` **`Derived Values`** – Point Evaluation

Results - Derived Values - Point Evaluation 2:

`Selection` – **Selection** = `Manual, 4` (*правий кінець струни*)

`Expression` – **Expression** = `R2`

LMB: Evaluate – Table 1

В таблицю **Table 1** буде додано результат обчислення `R2=0.57558`.

Надрукуйте суму всіх точкових реакцій.

`Results ->` **`Derived Values`** – GlobalEvaluation

Results - Derived Values - Global Evaluation 1:

`Expression` – **Expression** = `at0(0,R1)+at0(2,R2)+`
`k*abs(at0(1,u))+k2*abs(at0(0.5,u))`

LMB: Evaluate – Table 1

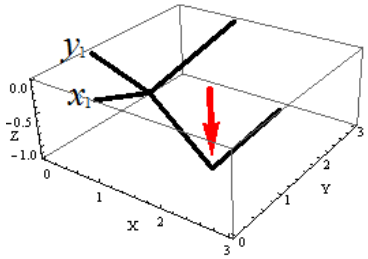
Остаточний вид таблиці показано на наступному рисунку.

R1, Point: 1	R2, Point: 4	at0(0,R1)+at0(2,R2)+k*abs(at0(1,u))+k2*abs(at0(0.5,u))
0.33140	0.57558	2.0000

Зверніть увагу, що в полі **Expression** вузла **Global Evaluation 1** для отримання сил реакцій ми використали оператор `at0` вибору значень величин в точках (використання доданків `R1+R2` без цього оператора призводить до помилки). Таким чином сума реакцій точок закріплення разом з силами реакцій пружин співпадає (з точністю до знака) з сумарною зовнішньою силою $\int_0^2 q(x)\,d\,x = -2$, яка діє на струну зверху.

5.4.7. Статичний прогин двох перетинних струн.

Розглянемо механічну систему, яка складається з двох натягнутих закріплених на кінцях перетинних під прямим кутом струн/ниток L_x і L_y довжиною L. Нитки в точці перетину (x_1, y_1) з'єднані. В деякій точці x_2 до струни L_x прикладена зосереджена сила f. Під її дією обидві нитки набувають вигляду ламаних. Потрібно знайти їх форму.



Очевидно, що на першу струну L_x діють дві зосереджені сили: зовнішня f і зосереджена сила a з боку іншою нитки. Силу, що діє на нитку L_y, позначимо через b. Очевидно, що $a+b=0$. Якщо знати сили a і b, то можна, розв'язавши відповідні ЗДР, визначити прогин кожної із струн. Для пошуку величин a і b існує додаткова умова рівності прогину ниток в точці перетину.

Покладемо довжини ниток однаковими L=3, для зосередженої сили покладемо $x_2=2$, f=−2, і точку перетину струн візьмемо $x_1=1, y_1=1$. Незважаючи на те, що нитки перпендикулярні, ми можемо розглянути їх рівняння на спільній розрахунковій області – відрізку [0,3], і вважати, що в спільній точці x=1 їх прогини однакові. Якщо позначити функцію прогинів нитки L_x через u, а нитки L_y – через v, то для обох функцій матимемо рівняння Пуассона (ф. (1) параграф 5.2.3) і в точці перетину матимемо двобічну умову/рівняння $u=v$.

Двобічність означає, що коли в умову входять дві величини, то зміна значення однієї спричиняє зміну значення іншої і навпаки. Можливість інтерпретувати умову як двобічну є у вузлів типу `Constraint` (зв'язок). В них умова задається у вигляді $0=R(x,u,u'_x,...)$ і, якщо в ПВ вузла **Constraint** встановлена опція **Apply reaction terms on** = `All physics (symmetric)`, то умова трактується як двобічна. В цьому параграфі ми продемонструємо використання такої двобічної умови `Constraint`, використовуючи в моделі два рівняння Пуассона, які діють на спільному відрізку, і розв'язки яких пов'язані в одній точці.

Розв'язання. Створіть одновимірну модель з двома «фізиками» – рівняннями Пуассона.

File->New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs – Poisson's Equation (poeq) ->
Classical PDEs – Poisson's Equation (poeq2) + Variables = v -> Stationary

При створенні моделі ми відразу додали два рівняння.

Untitled.mph (root): `Unit System` – <> = None

Геометрія моделі складається з відрізка [0, 3] і точок x=1, x=2. Створіть їх.

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1) :

Interval – **Left endpoint, Right endpoint** = 0,3

Geometry **-> Primitives->** Point

Geometry 1 – Point 1:

Point – **x** =1,2

LMB > Build All Objects

На створеній геометрії генеруйте СкЕ сітку зі значенням за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Обидва рівняння Пуассона застосовуватимуться на всьому відрізку і матимуть однаковий вигляд $\nabla\cdot(-c\nabla u)=f$ з c=-1 і f=0, але в першій фізиці діятиме точкове джерело сили f. Налаштуйте гілки обох рівнянь.

Poisson's Equation (poeq):

Domain Selection – **Selection**=Manual,1,2,3 (*відрізок* [0,3])

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = 0

Для першого рівняння створіть точкове джерело.

Physics -> **Boundaries** – Flux/Source

Poisson's Equation (poeq) - Flux/Source 1:

Boundary Sekection – **Selection** = Manual,3 (*точка x*=2)

Boundary Flux/Source – **g** =2

Boundary Absorption/Impedance Term – **q** =0

Введіть дані другого рівняння.

Poisson's Equation 2 (poeq2):

Domain Selection – **Selection**=Manual,1,2,3 (*відрізок* [0,3])

Poisson's Equation 2 (poeq2) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = 0

Задайте нульові граничні умови в обох фізиках.

Poisson's Equation (poeq):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1,4 (*лівий і правий кінці*)

Dirichlet Boundary Condition – **r** = 0

Poisson's Equation 2 (poeq2):

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Poisson's Equation 2 (poeq2) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1,4 (*лівий і правий кінці*)

Dirichlet Boundary Condition – **r** = 0

Додайте в гілку першого рівняння Пуассона вузол типу Constraint і налаштуйте його.

Poisson's Equation (poeq):

Physics -> **Boundary** – **Boundaries** – Constraint

Poisson's Equation (poeq) – Constraint 1:

Boundary Selection – **Selection** = Manual, 2 (*точка x*=1)

Boundary Constraint – **R** = v-u

Constraint Settings – **Apply reaction terms on** = All physics (symmetric)

Остання опція налаштовує зв'язок (Constraint) діяти симетрично, тобто застосовуватися до всіх залежних змінних, які беруть участь у виразі *R*. Це і є двобічність умови.

Зауваження. Двобічну умову (Constraint) можна розмістити в будь-якій з двох фізик (або навіть в обох), але з налаштуванням All physics (symmetric).

Виконайте обчислення.

Home **->** **Study** - Compute

Створиться дві графічні гілки. Перша «фізика» побудує графік розв'язку u, а друга – графік розв'язку v.

Нам відомий «аналітичний» розв'язок.

$$U(x) = \frac{1}{12}\left(-21 + 5\,x - 3\,|x-1|\right) + \,|x-2|$$

$$V(y) = \frac{1}{12}\left(-3 - y + 3\,|y-1|\right)$$

Зауваження. При перевірці, що останні функції задовольняють рівнянням прогину ниток типу $u''_{xx} = -\sum_{i=1}^{n-1} f_i \cdot \delta(x - x_i)$ (див. ф. (1) параграф 5.2.3), потрібно враховувати, що $|x - x_i|' = 2 \cdot H(x - x_i) - 1$ і $H'(x - x_i) = \delta(x - x_i)$, де $H(x)$ – функція Хевисайда, $\delta(x)$ – функція Дірака.

Для порівняння побудуйте графіки цих функцій в відповідних графічних групах. В гілку **1D Plot Group 1** додайте графік функції $U(x)$.

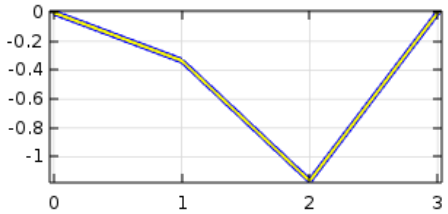
Results - 1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 2 :

y-Axis Data – **Expression** = (-21+5*x-3*abs(x-1))/12+abs(x-2)

Змініть самостійно стиль ламаної, щоб криві відрізнялися.

LMB > Plot



В гілку **1D Plot Group 2** додайте графік функції $V(x)$. Зверніть увагу на ім'я незалежної змінної – *x*.

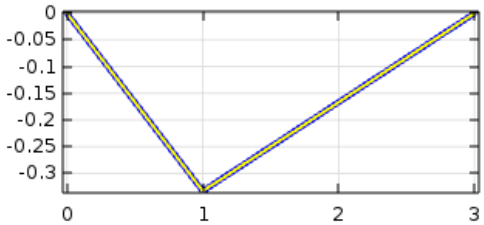
Results - 1D Plot Group 2 - Line Graph 1: RMB > Duplicate

1D Plot Group 2 - Line Graph 2 :

y-Axis Data – **Expression** = (-3-x+3*abs(x-1))/12

Налаштуйте самостійно стилі кривих, щоб графіки відрізнялися.

LMB > Plot



Тепер бажано побудувати тривимірне зображення механічної системи. Зробити це відразу неможливо, бо немає підходящих джерел даних. Треба мати двовимірну геометрію з двома перехресними відрізками. Отже створіть в моделі новий двовимірний компонент, в якому не буде розв'язуватися ніяке рівняння, але його геометрія використовуватиметься для подальших графічних побудов.

`Home` -> Model -> Add Component -> 2D

В ДМ буде створена гілка другого компонента **Component 2 (comp2)**. В його геометрії створіть два перехресних відрізка, які представлятимуть нитки в недеформованому стані. Відрізки можна приблизно накреслити, використовуючи команду Geometry – Draw – Line, а потім в створених вузлах типу `Bézier Polygon` скорегувати координати початкових і кінцевих точок відрізків.

Component 2 – Geometry 2:> `Geometry`-> **`Draw`** – Line

Накресліть відрізок приблизно бажаної форми і налаштуйте його параметри.

Bézier Polygon 1 (b1):

`Polygon Segments` – **Segment 1(linear):** (*виділити рядок*)
– **Control points <x>,<y>=** `0,1`
– **Control points <x>,<y>=** `3,1`

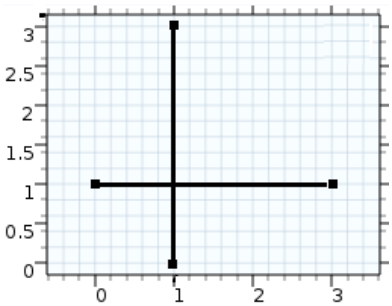
`Geometry` -> Draw – Line

Накресліть ще один відрізок перпендикулярний до першого і налаштуйте його параметри.

Bézier Polygon 2 (b2):

`Polygon Segments` – **Segment 1(linear):** (*виділити рядок*)
– **Control points <x>,<y>=** `1,0`
– **Control points <x>,<y>=** `1,3`

LMB > Build All Objects



Виконайте обчислення моделі.

Study 1 : LMB > Compute

Тепер можна побудувати джерела даних, які спиратимуться на ці відрізки.

Results – Data Sets - Study 1/Solution 1: RMB > Duplicate

Data Sets - Study 1/Solution 1 (2):

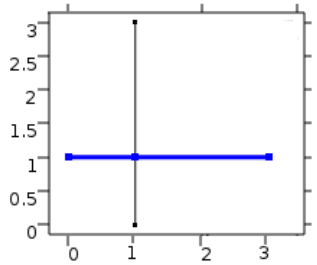
Solution – **Component**= Component (comp2)

Results – Data Sets: RMB > Edge 2D

Results – Data Sets - Edge 2D 1:

Data – **Data set** = Study 1/Solution 1(2)

Selection – **Selection** = Manual,1,4 (*горизонтальний відрізок*)

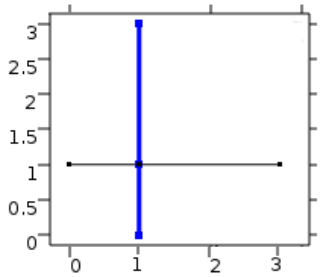


Results – Data Sets: RMB > Edge 2D

Results – Data Sets - Edge 2D 2:

Data – **Data set** = Study 1/Solution 1(2)

Selection – **Selection** = Manual,2,3 (*вертикальний відрізок*)



На горизонтальному і вертикальному відрізках другого компонента можна було б будувати графіки розв'язків *u* і *v*. Але вони (розв'язки *u* і *v*) створені в першому компоненті і їх значення недосяжні в другому компонента. Тому в першому компоненті ще треба створити оператори зв'язку, за допомогою яких можна передати значення розв'язків в другий компонент. Зокрема для цього можна застосувати General Extrusion оператори, які встановлюють відповідність координат геометрії першого компонента з просторовими координатами другого.

Component 1 :> Definitions > Component Couplings - General Extrusion

Component 1 - Definitions - General Extrusion 1 (genext1):

Source Selection – **Selection, <>** = Manual,1,2,3 (*відрізок* [0,3])

Destination Map – **x-expression** = x (*координата в 2-му компоненті*)

Тут встановлено, що подовжня координата відрізка [0,3] першого компонента відповідає координаті *x* в геометрії іншого компонента.

Розв'язок *v* буде потрібно використовувати на вертикальному відрізку другого компонента, вздовж якого змінюється координата y. Отже нам потрібен ще один General Extrusion оператор, який встановлює відповідність подовжньої координати відрізка [0,3] і координати y другого компонента.

Component 1 :> Definitions > Component Couplings - General Extrusion

Component 1 - Definitions - General Extrusion 2 (genext2):

Source Selection – **Selection, <>** = Manual,1,2,3 (*відрізок* [0,3])

Destination Map – **x-expression** = y (*координата в 2-му компоненті*)

Щоб оператори стали доступними, оновіть розв'язок.

Study 1: RMB > Update Solution

Тепер можна будувати графік деформованої механічної системи.

`Results` -> 2D Plot Group

2D Plot Group 3:

`Plot Settings` – **Plot data set edges** = `False`

RMB > Line

2D Plot Group 3 - Line 1:

`Data` – **Data set** = `Edge 2D 1`

`Expression` – **Expression** = `comp1.genext1(comp1.u)`

`Coloring and style` – **Line type** = `Tube`

– **Tube radius expression** = `0.03`

– **Radius scale factor** = `True, 1`

– **Coloring, Color** = `Uniform,Black`

2D Plot Group 3 - Line 1: RMB > Height Expression

2D Plot Group 3 - Line 1 - Height Expression 1:

`Axis` – **Scale factor** = `True, 1`

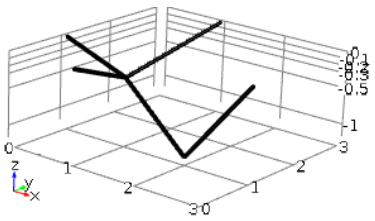
2D Plot Group 3 - Line 1: RMB > Duplicate

2D Plot Group 3 - Line 2:

`Data` – **Data set** = `Edge 2D 2`

`Expression` – **Expression** = `comp1.genext2(comp1.v)`

LMB > Plot



5.4.8. Стаціонарна хімічна реакція.

Швидкість реакції, де дві речовини X і Y реагуючи, створюють речовину Z $\left(X+Y\xrightarrow{k}Z\right)$, пропорційна добутку концентрацій речовин $\tilde{X}$ і $\tilde{Y}$

$$\frac{d\tilde{Z}}{dt}=k\,\tilde{X}\,\tilde{Y},$$

де k – коефіцієнт пропорційності, $\tilde{Z}$ - концентрація речовини Z. Якщо в реакції n молекул речовини X взаємодіють з однією молекулою Y, то швидкість зміни концентрації Z пропорційна $\tilde{X}^n\tilde{Y}$.

Нехай в деякому хімічному реакторі перетворення йдуть за наступною схемою

$$A\xrightarrow{k_1}X,\ B+X\xrightarrow{k_2}Y,\ A+Y\xrightarrow{k_3}X+D,\ X\xrightarrow{k_4}E$$

Концентрації речовин A і B в реакторі підтримуються незмінними, і деяким чином видаляються речовини D і E. Для зручності позначимо концентрації

речовин тими ж літерами. Тоді із закону діючих мас випливає наступна система рівнянь:

$$\begin{aligned} \frac{\partial X}{\partial t} &= k_1 A - k_2 B \cdot X + k_3 A \cdot Y - k_4 X + D_1 X''_{xx} \\ \frac{\partial Y}{\partial t} &= k_2 B \cdot X - k_3 A \cdot Y + D_2 Y''_{xx} \end{aligned} . \qquad (1)$$

Концентрація реагентів X і Y може різнитися в різних точках, тому в рівняння входять члени $D_1 X''_{xx}$ і $D_2 Y''_{xx}$, які враховують дифузію вздовж координати *x*.

Будемо шукати розв'язки системи (1), які не змінюються з часом. Тоді похідні по часу дорівнюватимуть нулю, і ми отримаємо наступну систему диференціальних рівнянь.

$$\begin{aligned} D_1 X''_{xx} + k_1 A - (k_2 B + k_4) \cdot X + k_3 A \cdot Y = 0 \\ D_2 Y''_{xx} + k_2 B \cdot X - k_3 A \cdot Y = 0 \end{aligned} \qquad (2)$$

Приклад. Нехай $0 \le x \le 1$ і відомі концентрації речовин X і Y на межах реактора, наприклад, $X(0) = 0.6$, $Y(0) = 0.2$, $X(1) = 0.2$, $Y(1) = 0.4$. Тоді ми матимемо крайову задачу для системи двох диференціальних рівнянь (2).

Для модельної задачі покладемо $A = 0.2$, $B = 0.5$, $k_1 = 0.3$, $k_2 = 1$, $k_3 = 0.7$, $k_4 = 0.1$ і $D_1 = 0.002$, $D_2 = 0.008$.

Р о з в ' я з о к . За допомогою майстра створіть 1D модель задачі.

File–>New–>Model Wizard–>1D–>

Mathematics– PDE Intefaces – Coefficient Form PDE (c) **->**

Number of dependent variables: = 2 +

Dependent variables: =[X,Y] **->** Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Створіть таблицю параметрів.

Global Definitions: RMB > Parameters

Задайте параметри відповідно до наступної таблиці.

Parameters

Name	Expression	Value
A	0.2	0.2
B	0.5	0.5
D1	2E-3	0.002
D2	8e-3	0.008
k1	0.3	0.3
k2	1	1
k3	0.7	0.7
k4	0.1	0.1

Геометрія моделі складається з відрізка [0,1].

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1) : LMB > Build All Objects

На створеному відрізку побудуйте СкЕ сітку.

Mesh 1:

Mesh Settings –**Element size** = Extra Fine

LMB > Build All

В дереві моделі виділіть вузол **Coefficient Form PDE 1**. Справа відкриється його панель. В розділі **Equation** цієї панелі наведено рівняння

$$e_a \frac{\partial^2 \boldsymbol{u}}{\partial t^2} + d_a \frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \left(-c \nabla \boldsymbol{u} - \alpha \boldsymbol{u} + \gamma\right) + \beta \cdot \nabla \boldsymbol{u} + a \boldsymbol{u} = f \ ,$$

де $\boldsymbol{u} = (X, Y)^T$ – шукана вектор–функція і $\nabla = \partial / \partial x$. Коефіцієнти системи задаються в полях нижньої частини ПВ.

Coefficient Form PDE 1:

Diffusion Coefficient – **c** = $\begin{matrix} D_1 & 0 \\ 0 & D_2 \end{matrix}$

Absorption Coefficient – **a** = $\begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix}$

Source Term – **f** = $\begin{matrix} k1 * A - (k2 * B + k4) * X + k3 * A * Y \\ k2 * B * X - k3 * A * Y \end{matrix}$

Для інших коефіцієнтів введіть $e_a = d_a = \alpha = \beta = \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix}$, $\gamma = \begin{matrix} 0 \\ 0 \end{matrix}$.

Задайте граничні умови.

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1 (т. x=0)

Dirichlet Boundary Condition – $\mathbf{r_1}, \mathbf{r_2}$ = 0.6,0.2

Physics -> **Boundary - Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,2 (т. x=1)

Dirichlet Boundary Condition – $\mathbf{r_1}, \mathbf{r_2}$ = 0.2,0.4

Виконайте обчислення.

Study 1 : LMB > Compute

В результаті буде побудовано графік розв'язку X. Налаштуйте його стиль і легенду.

1D Plot Group 1:

Axis –**Manual axis limits** = True

Axis –**y minimum, y maximum** = 0, 1 (*два поля*)

1D Plot Group 1 - Line Graph 1:

Coloring and style – **Color, Width** = Blue, 3

Legends – **Show legends** = True

– **Legends, <>** = Manual,X

В поточну графічну групу додайте графік розв'язку Y.

1D Plot Group 1 - Line Graph 1: RMB > Duplicate

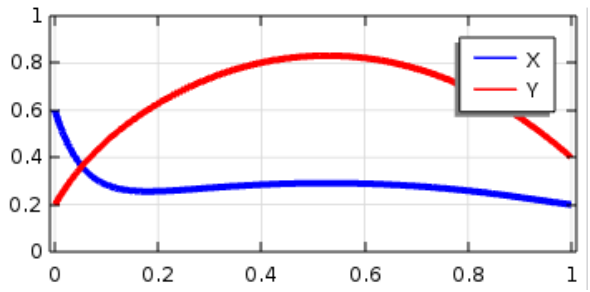
1D Plot Group 1 - Line Graph 2:

y-Axis Data – **Expression** = Y

Coloring and style – **Color, Width** = Red, 3

– **Legends, <>** = Manual,Y

LMB > Plot



5.4.9. Електричне поле двошарового сферичного конденсатора.

Знайдемо потенціал електростатичного поля сферичного конденсатора, заповненого неоднорідним діелектриком з діелектричною сталою

$$\varepsilon = \begin{cases} \varepsilon_1 \,, a < r < c \\ \varepsilon_2 \,, c < r < b \end{cases}.$$

Вважатимемо, що заряди всередині діелектрика відсутні, а на поверхні заданий електростатичний потенціал $u(a)=1, u(b)=0$. Потрібно розв'язати крайову задачу

$$\begin{aligned} -\nabla\cdot(\varepsilon_1 \nabla u_1) = 0 \quad \text{при} \quad a < r < c, \\ -\nabla\cdot(\varepsilon_2 \nabla u_2) = 0 \quad \text{при} \quad c < r < b, \end{aligned} \tag{1}$$

де u_1 і u_2 задовольняють при $r = a$ і $r = b$ граничним умовам

$$u_1(a)=1, u_2(b)=0,$$

і при $r = c$ умовам спряження (неперервність потенціалу u і вектора електричної індукції $\boldsymbol{D} = -\varepsilon \operatorname{grad} u$)

$$u_1(c) = u_2(c), \; \varepsilon_1 \frac{\partial u_1}{\partial r}(c) = \varepsilon_2 \frac{\partial u_2}{\partial r}(c). \tag{2}$$

Аналітичний розв'язок цієї задачі відомий [2]

$$u_1 = 1 + A\cdot\left(\frac{1}{r} - \frac{1}{a}\right), u_2 = \frac{\varepsilon_1}{\varepsilon_2} A\cdot\left(\frac{1}{r} - \frac{1}{b}\right), \text{ де } A = \frac{1}{\frac{1}{a} - \frac{1}{c} - \frac{\varepsilon_1}{\varepsilon_2}\left(\frac{1}{b} - \frac{1}{c}\right)}. \tag{3}$$

Ми порівняємо з ним розв'язок, отриманий системою.

Приклад. Для модельної задачі покладемо $a = 0.01$, $c = 0.02$, $b = 0.03$, $\varepsilon_1 = 4, \varepsilon_2 = 16$.

Р о з в ' я з о к . За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Poisson's Equation (poeq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Створіть таблицю параметрів.

Global Definitions: RMB > Parameters

Задайте параметри відповідно до наступної таблиці.

Parameters

Name	Expression	Value
a	0.01	0.01
c	0.02	0.02
b	0.03	0.03
e1	4	4
e2	16	16
A	1/(1/a-1/c-e1/e2*(1/b-1/c))	0.018462

В ПВ вузла **Component 1** змініть назву поздовжньої координати на *r*.

Component 1:

Spatial coordinates – **First** = r

General

Unit system:

Same as global system

Spatial coordinates

First (x)	Second (y)	Third (z)
r	y	z

Геометрія моделі складається з відрізка [*a*,*b*].

Geometry -> **Primitives ->** Interval

Geometry 1 - Interval 1 (i1) :

Interval – **Number of intervals** = Many

– **Points** = a,c,b

LMB > Build All Objects

На створеному відрізку побудуйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** =Finer

LMB> Build All

Задайте коефіцієнти рівняння. Перевірте, що в ПВ гілки **Poisson's Equation (poeq)** в секції вибору **Domain Selection** стоять індикатори 1,2 обох створених нами відрізків.

Зауваження. В гілці фізики **Poisson's Equation (poeq)** ми вказуємо обидва відрізка [*a*,*c*] і [*c*,*b*]. І спочатку у дочірньому вузлі **Poisson's Equation 1**, який представлятиме перше рівняння (1), зоною дії будуть ці два відрізка. Але подальше додавання ще одного дочірнього вузла для другого рівняння (1) з зоною дії на відрізку [*c*,*b*] видалить цей відрізок з зони дії першого рівняння.

Нагадаємо, що ми розв'язуємо сферично-симетричні рівняння (1) виду $-\varepsilon_i \Delta u_i = 0$. Записуючи лапласіан в сферичних координатах і враховуючи сферичну симетрію, отримуємо $-\frac{\varepsilon_i}{r^2}\frac{\partial}{\partial r}\left(r^2 \frac{\partial u}{\partial r}\right) = 0$. Для подальшого буде зручно переписати ці рівняння у вигляді $\frac{\partial}{\partial r}\left(-\varepsilon_i r^2 \frac{\partial u}{\partial r}\right) = 0$.

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = r^2*e1

Source Term – **f** =0

В гілці **Poisson's Equation (poeq)** створіть вузол другого рівняння Пуансона і налаштуйте його параметри.

Physics -> **Domains** -> Classical PDEs - Poisson's Equation

Poisson's Equation (poeq) - Poisson's Equation 2:

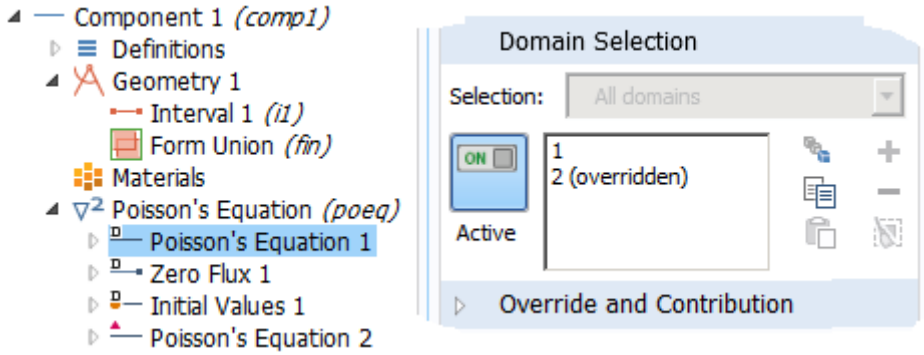
Domain Selection – **Selection,** <> = Manual, 2 (*відрізок* [*c*,*b*])

Diffusion Coefficient – **c** = r^2*e2

Source Term – **f** = 0

Зауваження. У вузлі **Poisson's Equation 2** ім'я розв'язку автоматично співпадає з ім'ям, заданим у вузлі батьківської «фізики» **Poisson's Equation (poeq)**, тобто використовується ідентифікатор *u*.

Відкрийте ПВ вузла **Poisson's Equation 1**. Його розділ **Domain Selection** повинен мати наступний вигляд.



Тут видно, що відрізок з індикатором 2 не входить до зони дії першого рівняння.

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** < > = Manual, 1 (*точка x=a*)

Dirichlet Boundary Condition – **r** = 1

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection,** < > = Manual,3 (*точка x=b*)

Dirichlet Boundary Condition – **r** = 0

Умови неперервності (2) електростатичного поля виконуватимуться автоматично (на межі розділу діелектриків електричні заряди відсутні).

Виконайте обчислення.

Home -> **Study** - Compute

Створюється гілка **1D Plot Group 1** з графіком розв'язку. Перевіримо його, побудувавши в цій же групі графік функцій (3). Нагадаємо, що заздалегідь в таблиці параметрів було створено константу A, яка використовується в цих формулах.

1D Plot Group 1 - Line Graph 1:

Coloring and style – **Color, Width** = Blue, 4

1D Plot Group 1 **-> Add Plot** – Line Graph

1D Plot Group 1 - Line Graph 2:

Selection – < > = 1 (*відрізок* [*a*,*c*])

y-Axis Data – **Expression** = 1+A*(1/r-1/a)

x-Axis Data – **Parameter, Expression** = Expression,r

Coloring and style – **Color, Width** = Yellow, 2

RMB > Duplicate

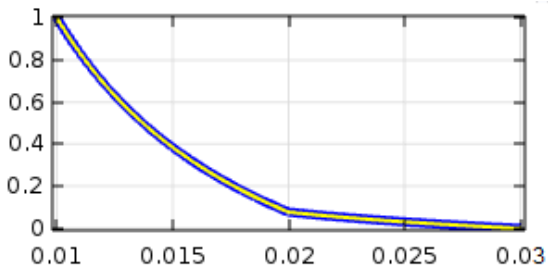
1D Plot Group 1 - Line Graph 3:

Selection – < > = 2 (*відрізок* [*c*,*b*])

Відрізок з індикатором 1 видаліть.

y-Axis Data – **Expression** = e1/e2*A*(1/r-1/b)

LMB > Plot



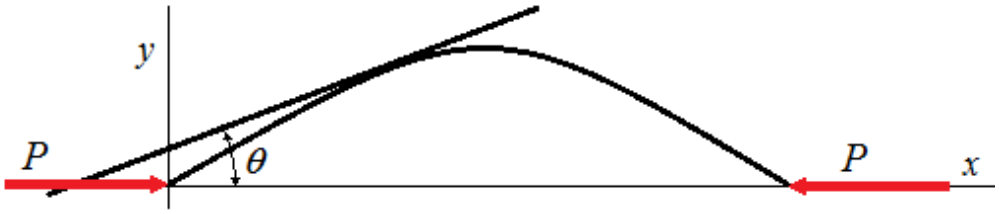
Вправа. В поточному файлі побудуйте тривимірний поверхневий графік електричного поля *u* по перерізу сферичного конденсатора.

Вказівка. Створити тривимірне джерело даних і оператор зв'язку.

5.4.10. Еластіка Ейлера.

Візьмемо тонку сталеву проволоку довжиною *S*, яка повинна легко гнутися без помітної залишкової деформації. Якщо її покласти на стіл вздовж прямої і трохи стиснути на кінцях, то вона прийме криволінійну форму. Плоску криву, форма якої відповідає осі такої проволоки, називають еластікою Ейлера.

Диференціальне рівняння згину проволоки/стрижня має вигляд [3] $E\,J\,\kappa = -P\,y$, де *E*, *J* – модуль пружності матеріалу проволоки і момент інерції її перерізу, κ – кривина зігнутої осі, *P* – стискаюча сила, *x*, *y* – координати точок проволоки на площині стола.



Нехай координата *x* спрямована вздовж лінії з'єднання кінців проволоки. Тоді кривина κ зігнутої осі дорівнює $d\,\vartheta / d\,s$, де ϑ зображений на рисунку кут між дотичною до зігнутої осі і віссю X (ϑ називають кутом нахилу пружної лінії), *s* – довжина дуги кривої. Якщо покласти $\lambda^2 = P / E\,J$, продиференціювати по *s* рівняння згину і врахувати, що $d\,y / d\,s = \sin\theta$, то отримаємо співвідношення

$$\frac{d^2\theta}{d\,s^2} = -\lambda^2 \sin\theta\,. \qquad (1)$$

До рівняння (1) слід додати крайові умови, наприклад у вигляді $\theta(0)=\theta_1$, $\theta(S)=\theta_2$. Після розв'язання крайової задачі (1) форма еластіки Ейлера визначається наступними параметричними рівняннями

$$x(s)=\int_0^s \cos\theta(\xi)\,d\,\xi\ ,\ y(s)=\int_0^s \sin\theta(\xi)\,d\,\xi\,. \qquad (2)$$

Зауваження. Врахування геометричної нелінійності проволоки без поперечного навантаження призвело до співвідношення (1), що збігається по формі з рівнянням коливань нелінійного маятника $\theta''_{tt}+\omega^2\sin\theta=0$ (див. п. 4.4.2).

Приклад. Для модельної задачі покладемо $\lambda=1, S=2\pi$.

Р о з в ' я з о к . За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Poisson's Equation (poeq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Геометрія моделі складатиметься з відрізка $[0,2\pi]$.

`Geometry`->**`Primitives`**-> Interval

Geometry 1 - Interval 1 (i1):

`Interval` – **Left endpoint, Right endpoint** =`0, 2*pi`

LMB > Build All Objects

В ПВ вузла **Component 1** змініть назву поздовжньої координати на *s*.

Component 1:

`Spatial coordinates` – **First** = `s`

На створеному відрізку побудуйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – **Element Size** = `Extremely Fine`

LMB> Build All

Задайте параметри рівняння.

Poisson's Equation (poeq) - Poisson's Equation 1:

`Diffusion Coefficient` – **c** =`1`

`Source Term` – **f** = `sin(u)`

Задайте граничні умови.

`Physics` -> **`Boundary`** – **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

`Boundary Selection` – **Selection** = `1` (т. x=0)

`Dirichlet Boundary Condition` – **r** = `pi`

`Physics` -> **`Boundary`** – **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

`Boundary Selection` – **Selection** = `2` (т. $x=2\pi$)

`Dirichlet Boundary Condition` – **r** = `-pi`

Налаштуйте систему на розв’язання сильно нелінійної задачі. Необхідний для цього вузол в дереві моделі спочатку не відображається. Тому виконайте команду

`Study` –> Solver - Show Default Solver

У вузлі **Study 1** з'явиться кілька дочірніх вузлів і можна буде виконати інструкції

Study 1 – Solver Configurations – Solution 1 –
Stationary Solver 1 - Fully Coupled 1:

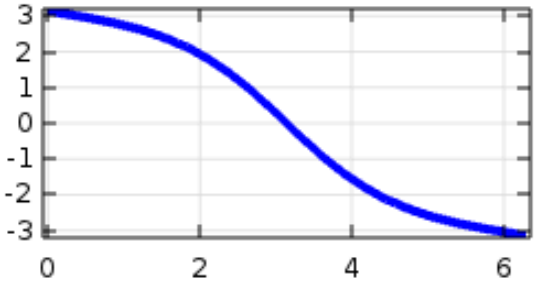
`Method and Termination` – **Nonlinear method** = `Automatic highly nonlinear (Newton)`

`Method and Termination` – **Maximum number of iterations** = `300`

Виконайте обчислення.

`Home` **`-> Study`** - Compute

Створюється графік розв’язку *u*, який зображає залежність кута θ нахилу пружної лінії від параметра *s* - довжини дуги вздовж осі проволоки.



Щоб побудувати форму проволоки, спочатку створіть в моделі дві змінні, які будуть обчислюватися за формулами (2). Але перед цим потрібно створити оператор інтегрування по відрізку $[0, 2\pi]$.

`Definitions` **`-> Coupling - Component Coupling`** – Integration

Definitions - Integration 1 (intop1):

`Source Selection` – **Geometric entity level** = `Domain`
– **Selection -** < > = `1` (*відрізок* $[0, 2\pi]$)

Ім'я оператора інтегрування `intop1` в полі **Operator name** залиште без змін. Нагадаємо, що оператор `intop1(expr)` інтегрує будь-який вираз `expr`, заданий на зоні дії оператора, по цій зоні (в нашому прикладі по відрізку $[0, 2\pi]$).

Тепер створіть змінні X і Y.

`Definitions` **`->Variables`** – Local Variables

Definitions – Variables 1:

`Geometric Entity Selection` – **Geometric entity level** = `Entire model`

`Variables` – **Name, Expression** = `X, intop1(cos(u)*(s<=dest(s)))`
– **Name, Expression** = `Y, intop1(sin(u)*(s<=dest(s)))`

Variables

Name	Expression
X	intop1(cos(u)*(s<=dest(s)))
Y	intop1(sin(u)*(s<=dest(s)))

Використовуючи створений оператор інтегрування `intop1` ми записали вирази для змінних X і Y, які обчислюються за формулами (2). Але спочатку було

модифіковано інтеграли по відрізку $[0, s]$ зі змінною верхньою межею в інтеграл по відрізку $[0, 2\pi]$

$$x(s) = \int_0^s \cos\theta(\xi)\, d\,\xi = \int_0^{2\pi} \cos\theta(\xi) \cdot (\xi \le s)\, d\,\xi .$$

Тут додатковий множник $(\xi \le s)$ дорівнює 1, коли він істинний, або 0 в іншому випадку. Щоб в операторі `intop1(...)`, що реалізує інтегрування по відрізку $[0, 2\pi]$, розрізнити поздовжню координату s і змінну інтегрування ξ (обидві змінюються вздовж відрізка $[0, 2\pi]$), зовнішню координату s було замінено оператором `dest(s)`. В результаті, останній інтеграл набув вигляду `intop1(cos(u)*(s<=dest(s)))`. Аналогічно було записано другий інтеграл (2).

Щоб оператор і змінні стали доступними, оновіть обчислення.

Study 1: RMB > Update Solution

Тепер можна перейти до креслення форми проволоки.

`Results` **`-> Plot Group`** – 1D Plot Group

`1D Plot Group 2` -> **`Add Plot`** – Line Graph

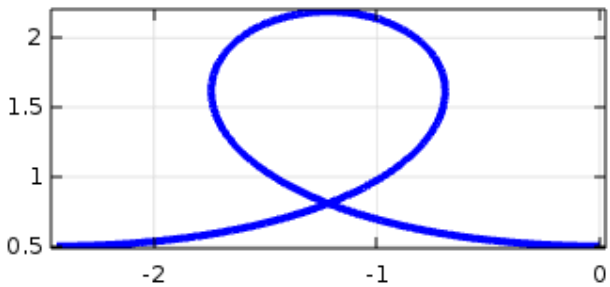
1D Plot Group 2 - Line Graph 1:

`Selection` – < > = 1

`y-Axis Data` – **Expression** = Y

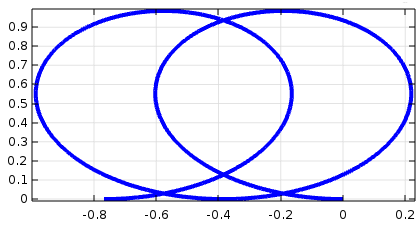
`x-Axis Data` – **Parameter, Expression** = Expression, X

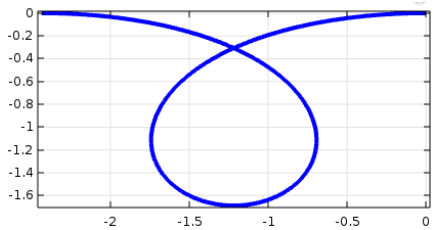
LMB > Plot



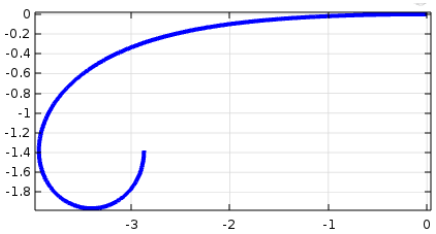
Зверніть увагу на те, що початкова точка кривої s=0 має координати x=0, y=0, тобто розташована на попередньому графіку справа.

Змінюючи у вузлах **Dirichlet Boundary Condition** граничні значення $\theta(0) = \theta_1$, $\theta(S) = \theta_2$ кута нахилу пружної лінії, можна отримати різні форми еластіки Ейлера. На наступному рисунку наведено кілька форм кривої при різних значеннях θ_1 і θ_2. Побудуйте їх самостійно.

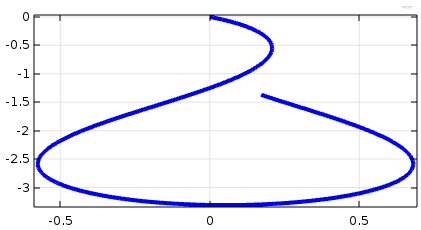




$\theta_1 = \pi$, $\theta_2 = -3\pi$



$\theta_1 = -\pi$, $\theta_2 = \pi$



$\theta_1 = -\pi$, $\theta_2 = \pi/2$

$\theta_1 = -\pi/4$, $\theta_2 = 2\pi/3$

5.4.11. Рівняння Шредінгера для гармонічного осцилятора.

Гармонійним осцилятором називають частинку, яка здійснює одновимірний рух під дією сили $F = -k\,x$, де x – відхилення частинки від положення рівноваги. Рівняння Шредінгера для гармонічного осцилятора має вигляд [4]

$$\frac{h^2}{2\mu}\frac{d^2\psi}{d\,x^2} + (E - U)\,\psi = 0,$$

де $U = \dfrac{\mu\,\omega_0^2}{2}x^2$ – потенціальна енергія частинки, E – її загальна енергія, ω_0 - циклічна частота осцилятора, h – стала Планка, μ - маса частинки. Тут E відіграє роль власного значення, яке підлягає визначенню при додатковій умові нормування власних функцій ψ

$$\int_{-\infty}^{+\infty} |\,\psi\,|^2 d\,x = 1.$$

Вводячи позначення $\lambda = 2E/h\,\omega_0$, $\xi = x\sqrt{\mu\,\omega_0/h}$, після деяких перетворень отримуємо рівняння

$$\frac{\partial^2\psi}{\partial\,\xi^2} + \left(\lambda - \xi^2\right)\psi = 0 \qquad (1)$$

з умовою нормування

$$\int_{-\infty}^{+\infty} |\,\psi\,|^2 d\,\xi = \sqrt{\frac{\mu\,\omega_0}{h}}. \qquad (2)$$

Розв’язком цієї задачі є власні функції

$$\psi_n(\xi) = \sqrt[4]{\frac{\mu\,\omega_0}{h}}\,\frac{e^{-\xi^2/2}\,H_n(\xi)}{\sqrt{2^n\,n!\sqrt{\pi}}}, \qquad (3)$$

які відповідають власним значенням $\lambda_n = 2n + 1$ (n=0,1,2,...). Тут $H_n(x)$ - поліноми Чебишева – Ерміта, які можна обчислювати за формулою

$$H_n(x) = (-1)^n e^{x^2}\frac{d^n}{d\,x^n}e^{-x^2}.$$

Ось значення перших чотирьох поліномів:

$$H_0(x) = 1,\ H_1(x) = 2\,x,\ H_2(x) = 4\,x^2 - 2,\ H_3(x) = 8\,x^3 - 12\,x. \qquad (4)$$

Ми зможемо порівняти функції ψ_n з власними значеннями і функціями Ψ_n, які відшукає система.

При пошуку в COMSOL Multiphysics власних функцій спочатку обирається часозалежний інтерфейс, наприклад, `Wave Equation` або `Heat Equation`, в якому застосовується `Eigenfrequency` або `Eigenvalue` тип дослідження. В залежності від типу дослідження, система робить в рівнянні заміну розв'язку $U(x,y,z,t)=u(x,y,z)\cdot e^{i\omega t}$ або $U(x,y,z,t)=u(x,y,z)\cdot e^{-\lambda t}$, і отримує нове рівняння, призначене для пошуку власних чисел і функцій. Наприклад, хвильове рівняння

$$e_a \frac{\partial^2 U}{\partial t^2} + \nabla\cdot(-c\nabla U) = f\cdot e^{-\lambda t}$$

при виборі `Eigenvalue` типу дослідження після заміни $U = u\cdot e^{-\lambda t}$ перетворюється на $\lambda^2 e_a u + \nabla\cdot(-c\nabla u) = f$, яке є початковим в COMSOL при пошуку власних коливань фізичних систем. Рівняння теплопровідності

$$d_a \frac{\partial U}{\partial t} + \nabla\cdot(-c\nabla U) = f\cdot e^{-\lambda t}$$

перетворюється на $-\lambda d_a u + \nabla\cdot(-c\nabla u) = f$. Останнє схоже на рівняння (1). Тому при побудові моделі ми виберемо `Classical PDEs - Heat Equation` інтерфейс. В п. 3.5 ми вже розв'язували задачу на власні коливання струни. Тут ми частково повторимо той розв'язок з деяким змінами.

Приклад. Для модельної задачі покладемо $\mu\omega_0/h = 1$. Знайдемо перші 6 власних значень і функцій рівняння (1).

Р о з в ' я з а н н я . За допомогою майстра створіть 1D модель задачі.

File -> New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs - Heat Equation (hteq) ->
Custom Studies – Eigenvalue

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

В рівнянні гармонічного осцилятора вважається, що незалежна координата змінюється від $-\infty$ до $+\infty$. Ми не можемо це реалізувати, але в якості розрахункової області можемо обрати достатньо довгий інтервал. Нехай це буде інтервал [-10,10].

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1):

`Interval` – **Left endpoint, Right endpoint** =`-10, 10`

LMB > Build All Objects

На створеному відрізку згенеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – **Element Size** = `Extremely Fine`

LMB> Build All

Задайте параметри рівняння. В ПВ вузла **Wave Equation (waeq) - Wave Equation 1** в секції вибору повинна стояти одиниця (індикатор відрізка). В розділі **Equation** наведено рівняння

$$-\lambda d_a u + \nabla \cdot (-c \nabla u) = f, \quad \nabla = \frac{\partial}{\partial x}.$$

Щоб воно набуло вигляду (1) потрібно ввести наступні значення параметрів

Wave Equation (waeq) - Wave Equation 1:

Diffusion Coefficient – **c** = 1

Source Term – **f** = -x^2*u

Absorption Coefficient – $\mathbf{d_a}$ = 1

Цим задається рівняння $-\lambda u - u''_{xx} = -x^2 u$.

Умова нормування (2) гарантуватиме швидке спадання функції ψ, тому граничні значення на далеких границях несуттєві і ми залишимо їх заданими за замовчуванням.

Відкрийте ПВ вузла **Step 1: Eigenvalue**. В полі **Desired number of eigenvalues** вказується бажана для визначення кількість власних чисел. Залиште в ньому значення 6. Поле **Search for eigenvalues around** вказує число, в околі якого будуть обчислюватися власні значення. Залиште в ньому значення 0.

Перш, ніж розв'язувати задачу, призначте спосіб нормування власних функцій, які, як відомо, визначаються з точністю до сталого множника. В системі власні функції підбираються так, щоб їх норма дорівнювала одиниці. Спосіб нормування задається в одному з дочірніх вузлів гілки **Study 1**. Щоб ці вузли відобразити (якщо в ДМ їх ще немає), виконайте команду

Study -> Solver - Show Default Solver

Тепер оберіть тип норми.

Study 1 – Solver Configurations – Solution 1 - Eigenvalue Solver 1:

Output – **Scaling of eigenvectors** = RMS (*значення за замовчуванням*)

Цей вибір означатиме, що в якості норми власних функцій Ψ застосовуватиметься середньоквадратичне (RMS – root mean square) значення, обчислюване за формулою $\|\Psi\| = \sqrt{\int_{x_1}^{x_2} |\Psi|^2 dx / (x_2 - x_1)}$, яке буде дорівнювати одиниці.

Виконайте обчислення.

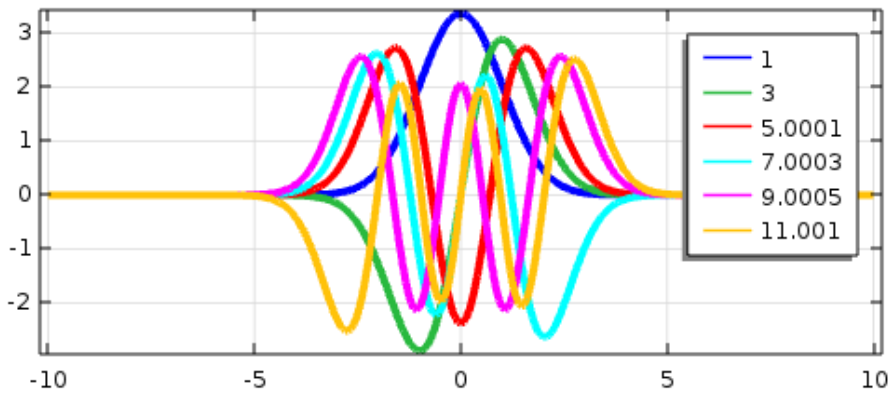
Study 1 : LMB > Compute

В графічній зоні вузла **Results - 1D Plot Group 1 – Line Graph 1** отримайте графіки власних функцій $\Psi_n(x)$. Включіть відображення легенди.

1D Plot Group 1 - Line Graph 1:

Legend – **Show legends** = True

LMB > Plot



Надрукуйте значення перших шести власних чисел (хоча їх вже показано на легенді).

Results – Derived Values: RMB > Global Evaluation

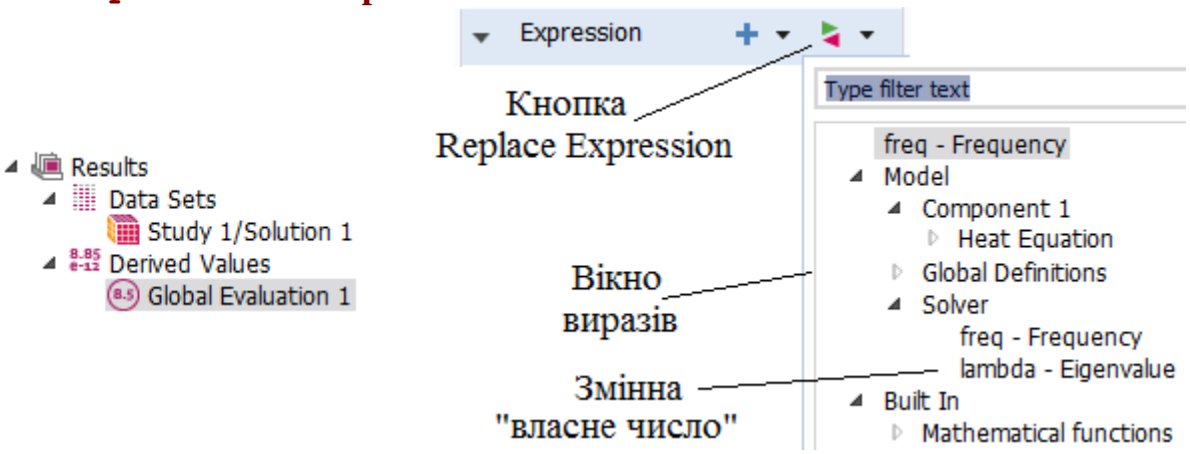
Global Evaluation 1:

`Data` – Data set = `Study 1/Solution 1`

`Data` – Eigenvalue selection = `All`

Щоб дізнатися (і ввести) ім'я змінної, яка представляє власні числа, в правій частині заголовка розділу **`Expression`** натисніть кнопку Replace expression . У вікні готових величин, яке з'явиться, розкрийте дерево змінних готових до вставки, знайдіть і виберіть подвійним клацанням змінну `lambda`.

`Expression` – Expression = `lambda`



LMB > Evaluate

Eigenvalue	Eigenvalue
1.0000	1.0000
3.0000	3.0000
5.0001	5.0001
7.0003	7.0003
9.0005	9.0005
11.001	11.001

Значення змінної `lambda` майже співпадають з точними значеннями 1,3,5,7,9,11, отриманими з формули $\lambda_n = 2n+1$ при n=0,1,...,5. Для подальшої перевірки побудуйте графіки кількох перших власних функцій $\psi_n(x)$. Але спершу треба розібратися з їх нормою.

Опція нормування RMS, встановлювана у вузлі **Eigenvalue Solver 1**, означає обчислення власних функцій Ψ_n з одиничним значенням норми, тобто інтеграла $\sqrt{\int_{x_1}^{x_2} |\Psi_n|^2 dx/(x_2-x_1)}$, де x_1, x_2 координати кінців відрізка розрахункової області. При побудові функцій ψ_n вважається, що норма обчислюється за формулою (2). Отже функції Ψ_n, обчислювані системою, і ψ_n, обчислювані за формулою (3), відрізняються чисельним множником. Враховуючи, що ми обрали $\mu\omega_0/h=1$, і порівнюючи формули

$$\sqrt{\int_{x_1}^{x_2} |\Psi_n|^2 dx/(x_2-x_1)}=1 \quad , \qquad \int_{x_1}^{x_2} |\psi_n|^2 dx=1,$$

бачимо, що повинно бути $\Psi_n=\sqrt{x_2-x_1}\,\psi_n$. Таким чином, для порівняння графіків власних функцій Ψ_n і ψ_n потрібно враховувати множник $\alpha=\sqrt{x_2-x_1}$. В нашому прикладі $x_1=-10$, $x_2=10$.

Побудуйте графіки власних функцій Ψ_0 і $\alpha\psi_0$ (n=0, $\alpha=\sqrt{20}$).

Results -> **Plot Group** – 1D Plot Group

1D Plot Group 2:

Data – **Eigenvalue selection** = From list

Data – **Eigenvalues** = 1

1D Plot Group 2 -> **Add Plot** – Line Graph

1D Plot Group 2 - Line Graph 1:

Selection – < > = 1

y-Axis Data – **Expression** = u

x-Axis Data – **Parameter, Expression** = Expression, x

Coloring and style – **Color, Width** = Red, 4

Line Graph 1: RMB > Duplicate

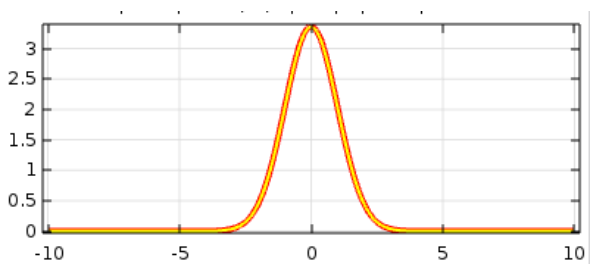
1D Plot Group 2 - Line Graph 2:

y-Axis Data– **Expression** =exp(-x^2/2)/sqrt(sqrt(pi))*sqrt(20)

Остання формула отримана з (3) при n=0 з додаванням множника α.

Coloring and style – **Color, Width** = Yellow, 2

LMB > Plot



Аналогічно побудуйте графіки власних функцій Ψ_1 і $\alpha\psi_1$ (n=1).

1D Plot Group 2: RMB > Duplicate

1D Plot Group 3:

Data – **Eigenvalues** = 3

1D Plot Group 3 - Line Graph 2:

y-Axis Data – **Expression** =

```
exp(-x^2/2)*2*x/sqrt(2)/sqrt(sqrt(pi))*sqrt(20)
```

Оскільки n=1, то в (3) для полінома Чебишева - Ерміта застосована друга формула (4) $H_1(x) = 2x$.

LMB > Plot (*наступний рисунок ліворуч*)

Побудуйте графіки власних функцій Ψ_2 і $\alpha\psi_2$ (n=2), враховуючи в (3), що $H_2(x) = 4x^2 - 2$.

1D Plot Group 3: RMB > Duplicate

1D Plot Group 4:

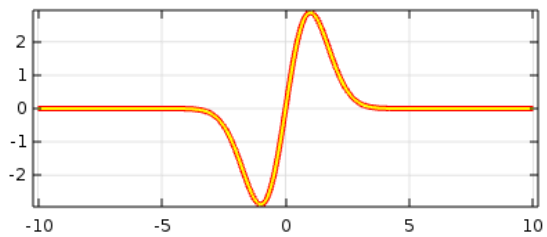
Data – **Eigenvalues** = 5.0001

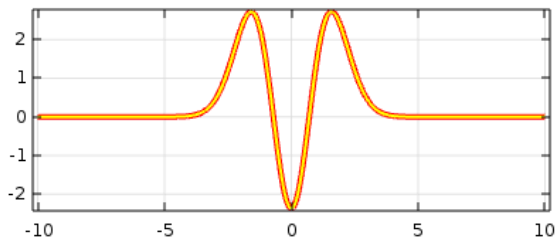
1D Plot Group 4 - Line Graph 2:

y-Axis Data – **Expression** =

```
exp(-x^2/2)*(4*x^2-2)/2/sqrt(2)/sqrt(sqrt(pi))*sqrt(20)
```

LMB > Plot (*наступний рисунок праворуч*)





В рівнянні Шредінгера безпосередній фізичний зміст має не сама функція ψ, а $|\psi|^2$, що тлумачиться як густина імовірності перебування частинки в околі елементарного об'єму $dx\,dy\,dz$ точки (x, y, z). Ви можете самостійно побудувати графіки функцій $|\Psi_n|^2$.

5.4.12. Дифузія крізь напівнепроникливу перегородку.

Замкнутий циліндричний сосуд з непроникливими стінками складається з двох циліндричних сосудів, кожний з яких заповнений однорідним середовищем з розподіленою по них речовиною. Концентрація цієї речовини в обох складових судини і властивості середовища в одній і іншій судині різні. Частинки речовини розпадаються зі швидкістю пропорційній концентрації. На лівому і правому торцях судини концентрація речовини підтримується сталою. Дослідити стаціонарну дифузію згаданої речовини в складеному циліндрі, вважаючи, що його частини з'єднані напівнепроникливою перегородкою, і дифузія речовини крізь неї відбувається за законом $-D\dfrac{\partial u}{\partial n} = \alpha\,(u - u_{ext})$, де u – концентрація речовини біля перегородки, α – коефіцієнт проникливості перегородки, u_{ext} - концентрація з іншого боку, D – коефіцієнт дифузії.

Спрямовуючи вісь x вздовж осі циліндрів і розміщуючи початок координат в місці їх з'єднання, отримуємо крайову задачу [2]

$$D_1 \frac{\partial^2 u_1}{\partial x^2} - b_1 u_1 = 0, \; -L_1 < x < 0,$$

$$D_2 \frac{\partial^2 u_2}{\partial x^2} - b_2 u_1 = 0, \; 0 < x < L_2,$$

$$u_1(-L_1) = u_1^0, \quad u_2(L_2) = u_2^0,$$
$$-D_1 u'_{1x}(0) = \alpha \, [u_1(0) - u_2(0)],$$
$$-D_2 u'_{2x}(0) = \alpha \, [u_1(0) - u_2(0)].$$

Тут $u_i(x)$– концентрація речовини в лівому і правому циліндрах $(i = 1, 2)$, D_i – коефіцієнти дифузії, b_i – коефіцієнти розпаду, α – коефіцієнт проникливості перегородки.

Приклад. Для модельної задачі покласти $D_1 = 1, D_2 = 4$, $b_1 = 0.5, b_2 = 0.75$, $\alpha = 0.75$ і $u_1(0) = 1$, $u_2(0) = 3$, $L_1 = 1, L_2 = 1$.

Ми продемонструємо два способи дослідження моделі. В першому способі в спільному компоненті буде створено дві фізики, в яких використовуватимуться однакові імена шуканих величин, пара (pair) і оператори зв'язку. В другому - дві фізики в одному компоненті з різними іменами шуканих величин, без використання пар і операторів зв'язку.

Розв'язок 1. За допомогою майстра створіть 1D модель задачі.

File –> New –> Model Wizard –> 1D –> Mathematics -
Classical PDEs - Convection-Diffusion Equation (cdeq) –> Stationary

Зауваження. Рівняння теплопровідності має схожий з рівнянням дифузії вигляд і, в принципі, можна було б використати `Classical PDEs - Heat Equation` інтерфейс.

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Щоб на рисунках розрахункової області можна було спостерігати індикатори елементів геометрії, встановіть опцію їх відображення у вузлі **View 1**.

Component 1 – Definitions - View 1:

`View` – **Show geometry labels** = `True`
– **Show edge direction arrows** = `True`

Остання опція включає режим відображення напрямку зростання параметра/координати вздовж відрізка.

В гілці **Global Definitions** згенеруйте вузол, призначений для створення параметрів.

Global Definitions: RMB > Parameters

Задайте параметри відповідно до наступної таблиці.

Parameters

Name	Expression	Value
D1	1	1
D2	4	4
b1	0.5	0.5
b2	0.75	0.75
u1	1	1
u2	3	3
a	0.75	0.75

Збережіть таблицю параметрів в текстовий файл. Для цього достатньо натиснути на кнопку Save to File, розташовану під таблицею,

Global Definitions - Parameters: LMB > Save to File

і ввести ім'я файла, наприклад, 5_4_12_Parameters.txt.

Створіть розрахункову область моделі, яка складається з двох пристикованих відрізків, і поєднайте їх в збірку (assembly).

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1) :

`Interval` – **Left endpoint, Right endpoint** = `-1,0`

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 2 (i2) :

`Interval` – **Left endpoint, Right endpoint** = `0,1`

LMB > Build All Objects

Geometry 1 - Form Union:

`Form Union/Assembly` – **Action** = `Form an assembly`

`Form Union/Assembly` – **Create pairs** =`True` (*значення за замовчуванням*)

Остання інструкція вказує системі створювати пари (за замовчуванням типу `Identity Pair`).

LMB > Build All

Зверніть увагу, що назва вузла **Form Union** в ДМ змінилася на **Form Assembly**. Крім того, точка x=0 відрізка [-1,0] отримала індикатор з номером 2. Ця ж точка, але як частина відрізка [0,1], отримала індикатор з номером 3.

Генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – **Element Size** =`Extra fine`

LMB> Build All

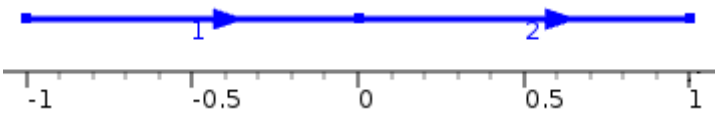
В нашій задачі є дві зони (дві судини), в яких діятимуть різні рівняння дифузії. Для визначення першого рівняння вузол **Convection-Diffusion Equation 1** вже створено. В гілці **Convection-Diffusion Equation (cdeq)** створіть ще один вузол для задання параметрів другого рівняння.

`Physics->` **`Domains ->`** Classical PDEs - Convection-Diffusion Equation

Визначить зони дії обох рівнянь.

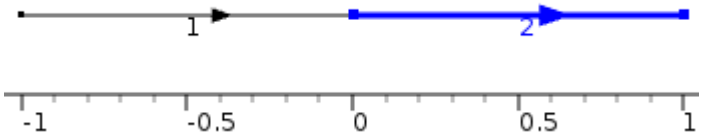
Convection-Diffusion Equation (cdeq):

`Domain Selection` – **Selection, <>** = `All domais,1,2`

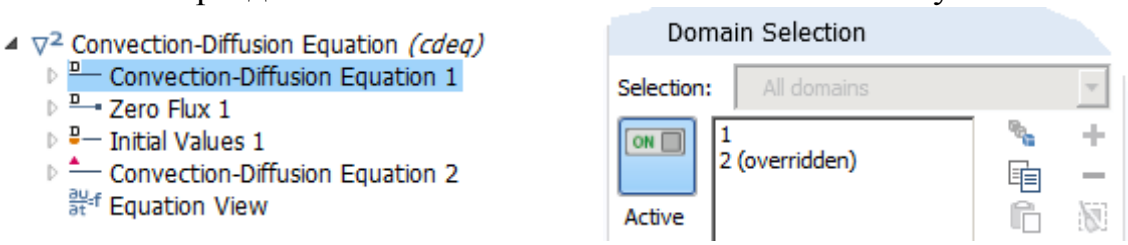


Convection-Diffusion Equation (cdeq) - Convection-Diffusion Equation 2:

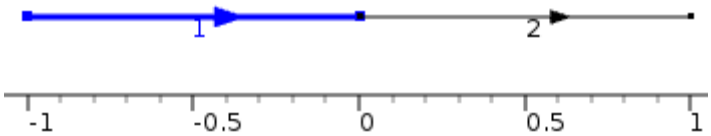
Domain Selection – Selection, <> = Manual, 2



Задання області дії другого рівняння автоматично звузило зону дії першого рівняння. Щоб це перевірити відкрийте ПВ вузла **Convection-Diffusion Equation 1**. Його розділ **Domain Selection** повинен мати наступний вигляд.



В секції **Domain Selection** видно, що відрізок з індикатором 2 не входить до зони дії першого рівняння, і в графічній зоні системи виділено відрізок 1.



Налаштуйте вузли **Convection-Diffusion Equation 1** і **Convection-Diffusion Equation 2**. В розділах **Equation** цих вузлів наводяться рівняння конвекції – дифузії в наступному вигляді

$$d_a \frac{\partial u}{\partial t} + \nabla \cdot (-c \nabla u) + \beta \cdot \nabla u = f .$$

Задайте їх параметри.

Convection-Diffusion Equation 1:

Diffusion Coefficient – c = D1

Diffusion Coefficient – β = 0

Source Term – f = -b1*u

Damping or Mass Coefficient – d_a = 0

Цим задається рівняння $D_1 u''_{1xx} - b_1 u_1 = 0$ на відрізку [-1,0].

Convection-Diffusion Equation 2:

Diffusion Coefficient – c = D2

Diffusion Coefficient – β = 0

Source Term – f = -b2*u

Damping or Mass Coefficient – d_a = 0

Цим визначається рівняння $D_2 u''_{2xx} - b_2 u_2 = 0$ на відрізку [0,1].

Задайте ліву і праву граничні умови.

Physics **-> Boundary - Boundaries ->** Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** < > = Manual, 1 (*точка* x=-1)

Dirichlet Boundary Condition – **r** = u1

Physics **-> Boundary - Boundaries ->** Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection,** < > = Manual,4 (*точка* x=1)

Dirichlet Boundary Condition – **r** = u2

Залишилося пов'язати розв'язки рівнянь на внутрішній границі x=0. Для цього нам знадобляться два оператора тотожного відображення Identity Mapping. Вони використовуватимуться для передавання значень розв'язків з точки 2 в точку 3 (номери індикаторів однієї і тієї ж точки) і навпаки.

Definitions->**Coupling - Component Coupling - Identity Mapping**

Definitions - Identity Mapping 1 (idmap1):

Source Selection – **Geometric entity level** = Boundary

– **Selection -** < > =Manual, 2 (т. x=0 *відрізка* [-1,0])

Створіть оператор відповідності в зворотну сторону.

Definitions->**Coupling - Component Coupling - Identity Mapping**

Definitions - Identity Mapping 2 (idmap2):

Source Selection – **Geometric entity level** = Boundary

– **Selection -** < > =Manual, 3 (т. x=0 *відрізка* [0,1])

Зауваження. Коли ви підводите покажчик миші до точки x=0 і клацаєте, то скоріш за все в секцію вибору вставлятиметься індикатор 2 цієї точки. Щоб вставити індикатор 3, підведіть мишу до точки, трохи крутніть колесико, і лише потім клацайте мишею по точці. Кожне невелике прокручування колесика буде зміщувати фокус миші по черзі між індикаторами 2 і 3.

Задайте першу «внутрішню» граничну умову $-D_1 u'_{1x}(0) = \alpha\,[u_1(0) - u_2(0)]$.

Convection-Diffusion Equation (cdeq):

Physics -> **Boundary – Boundaries** – Flux/Source

Convection-Diffusion Equation (cdeq) - Flux/Source 1:

Boundary Selection –**Selection -** < > = Manual,2 (т. x=0 *відрізка* [-1,0])

Boundary Flux/Source – **g** = -a*(u-idmap2(u))

Оскільки точка з індикатором 2 належить відрізку [-1,0], то величина u відноситься до цього відрізка і є значенням шуканої функції $u_1(x)$ в точці $x = 0$. Для отримання значення функції $u_2(x)$ в точці $x = 0$ ми використали оператор тотожної відповідності idmap2(u).

Задайте другу «внутрішню» граничну умову.

Physics -> **Boundary – Boundaries** – Flux/Source

Convection-Diffusion Equation (cdeq) - Flux/Source 2:

Boundary Selection –**Selection -** < > = 3 (т. x=0 *відрізка* [0,1])

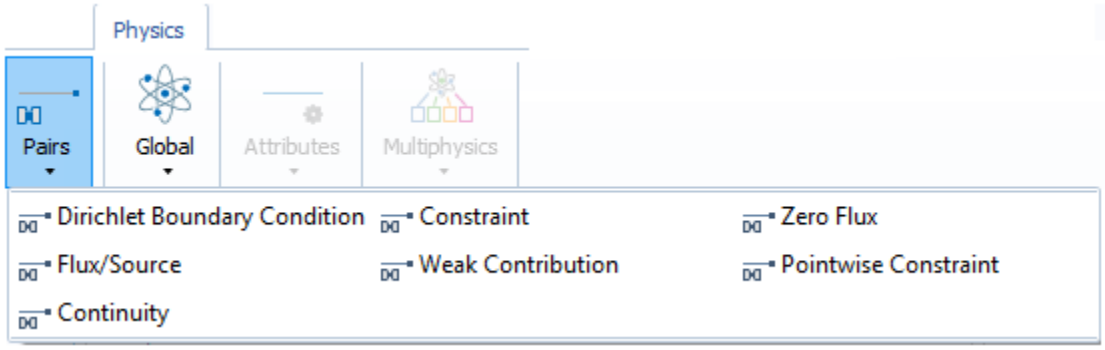
Використовуйте прокручування колеса миші для переміщення фокуса по черзі між індикаторами 2 і 3.

Boundary Flux/Source – **g** = a*(idmap1(u)-u)

Точка з індикатором 3 належить відрізку [0,1] і нормаль $\boldsymbol{n}$ до межі цього відрізка в точці x=0 дорівнює $\boldsymbol{n}=-\boldsymbol{i}$, де $\boldsymbol{i}$ одиничний орт вздовж осі x. Крім того, $\nabla u=\boldsymbol{i}u'_x$. Тому ліва частина умови $-\boldsymbol{n}\cdot(-c\nabla u)=g-qu$ вузла **Flux/Source 2** набуває вигляду $-\boldsymbol{n}\cdot(-c\nabla u)=-(-\boldsymbol{i})\cdot(-c\boldsymbol{i}u'_x)=-cu'_x$. Отже, щоб умова $-\boldsymbol{n}\cdot(-c\nabla u)=g-qu$ набула вигляду $-D_2u'_{2x}(0)=\alpha\,[u_1(0)-u_2(0)]$ потрібно покласти, q=0 (у нас $c=D_2$), в виразі для `g` в якості u_2 записати величину u, а для значення $u_1(0)$ використати оператор `idmap1(u)` .

Зауваження. Граничні умови типу `Flux/Source` (і інші) можуть задаватися командою з випадної кнопки – списку Pairs панелі `Physics`.

`Physics` –> Boundary – Pairs –> Flux/Source

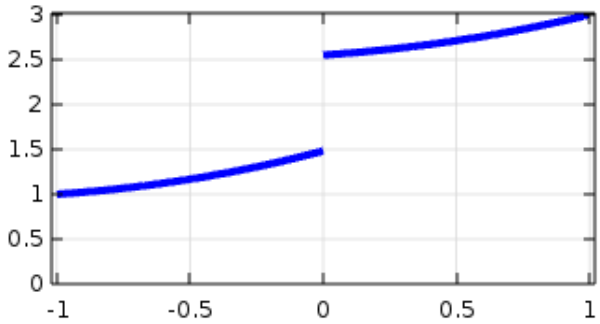


Але вони будуть відноситися до всієї розділювальної межі, а не до різних її сторін. Так умова у вузлі `Flux/Source` на парі (Pairs – Flux/Source) буде діяти як точкове джерело зовнішнього впливу (наприклад, як точкова зовнішня сила в задачі про прогин струни під дією зосереджених сил), прикладене в одній точці. Розв'язок залишиться неперервним. До того ж, два вузла `Flux/Source` на спільній парі будуть підсумовувати свої зусилля, і розглядатимуться як один вузол з сумарним значенням. В поточній задачі потрібно задавати різні умови на різних боках розділювальної точки. Тому ми створили два вузла типу `Flux/Source` з випадної кнопки-списку Boundary – Boundaries, задаючи в точці x=0 по різні її боки дві окремі граничні умови.

Виконайте обчислення моделі.

`Home` –> Study - Compute

В графічній зоні вузла **Results - 1D Plot Group 1 – Line Graph 1** буде побудовано наступний графік розв'язку.



Розв'язок 2. Цей розв'язок частково повторює попередній. За допомогою майстра створіть 1D модель задачі з двома фізиками.

File -> New -> Model Wizard -> 1D -> Mathematics -
Classical PDEs - Convection-Diffusion Equation (cdeq) ->
Classical PDEs - Convection-Diffusion Equation (cdeq) + Variables = w ->
Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

В гілці **Global Definitions** згенеруйте вузол, призначений для задання параметрів.

Global Definitions: RMB > Parameters

Значення параметрів візьміть з файла 5_4_12_Parameters.txt, створеного в першому розв'язку. Для цього натисніть на кнопку Load from File, розташовану під таблицею,

Global Definitions - Parameters: LMB > Load from File

і введіть ім'я файла 5_4_12_Parameters.txt.

Створіть геометрію моделі з двох відрізків.

Geometry -> **Primitives** -> Interval

Geometry 1 - Interval 1 (i1) :

Interval – **Number of intervals** = Many

Interval – **Points** = -1,0,1

LMB > Build All Objects

Для об'єднання відрізків тут використовуватиметься операція Union, обирана за замовчуванням.

Генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** =Extra fine

LMB> Build All

Налаштуйте зони дії і задайте коефіцієнти рівнянь обох фізик.

Convection-Diffusion Equation (cdeq):

Domain Selection – **Selection,** <> = Manual,1 (*відрізок* [-1,0])

Convection-Diffusion Equation (cdeq) - Convection-Diffusion Equation 1:

Diffusion Coefficient – **c** = D1

Diffusion Coefficient – **β** = 0

Source Term – **f** = -b1*u

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Convection-Diffusion Equation 2 (cdeq2):

Domain Selection – **Selection,** <> = Manual,2 (*відрізок* [0,1])

Convection-Diffusion Equation 2 (cdeq2) - Convection-Diffusion Equation 1:

Diffusion Coefficient – **c** = D2

Diffusion Coefficient – **β** = 0

Source Term – **f** = -b2*w

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Задайте ліву і праву граничні умови.

Convection-Diffusion Equation (cdeq):

Physics -> **Boundary - Boundaries** -> Dirichlet Boundary Condition

Convection-Diffusion Equation (cdeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** < > = Manual, 1 (*точка* x=−1)

Dirichlet Boundary Condition – **r** = u1

Convection-Diffusion Equation 2 (cdeq2):

Physics -> **Boundary - Boundaries ->** Dirichlet Boundary Condition

Convection-Diffusion Equation 2 (cdeq2) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** < > = Manual,3 (*точка* x=1)

Dirichlet Boundary Condition – **r** = u2

Здайте першу «внутрішню» граничну умову в першій фізиці.

Convection-Diffusion Equation (cdeq):

Physics -> **Boundary – Boundaries** – Flux/Source

Convection-Diffusion Equation (cdeq) - Flux/Source 1:

Boundary Selection –**Selection -** < > = Manual,2 (т. x=0)

Boundary Flux/Source – **g** = -a*(u-w)

Цим задається умова $D_1 u'_x(0) = -\alpha\,[u(0) - w(0)]$.

Другу «внутрішню» граничну умову задайте в другій фізиці.

Convection-Diffusion Equation 2 (cdeq2):

Physics -> **Boundary – Boundaries** – Flux/Source

Convection-Diffusion Equation (cdeq) - Flux/Source 1:

Boundary Selection –**Selection -** < > = Manual,2 (т. x=0)

Boundary Flux/Source – **g** = a*(u-w)

Враховуючи, що $\boldsymbol{n} = -\boldsymbol{i}$, цим задається умова $-D_2\, w'_x(0) = \alpha\,[u(0) - w(0)]$.

Виконайте обчислення моделі.

Home -> Study - Compute

Після їх завершення в гілці Results створяться дві графічні групи, в яких будуть побудовані графіки розв'язків (кожна фізика будує власний розв'язок). Об'єднайте їх, побудувавши спільний графік.

Results -> **Plot Group** – 1D Plot Group

1D Plot Group 3 -> **Add Plot** – Line Graph

1D Plot Group 3 - Line Graph 1:

Selection – **Selection -** < > = 1 (*відрізок* [-1,0])

y-Axis Data – **Expression** = u

x-Axis Data – **Parameter, Expression** = Expression, x

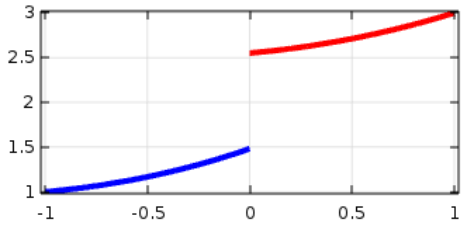
1D Plot Group 3 - Line Graph 1: RMB > Duplicate

1D Plot Group 3 - Line Graph 2:

Selection – **Selection -** < > = 2 (*відрізок* [0,1])

y-Axis Data – **Expression** = w

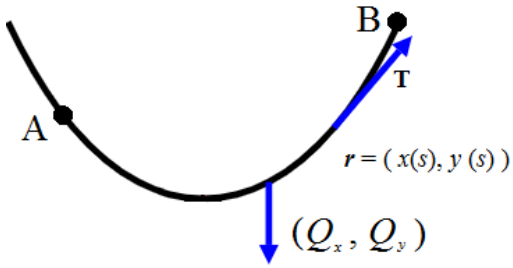
Налаштуйте стилі оформлення кривих і виконайте команду Plot. В результаті буде побудований той же графік, що в першому розв'язку.



5.4.13. Ланцюгова лінія.

Ланцюговою лінією називається плоска крива, форма якої відповідає закріпленій на кінцях однорідній гнучкій нерозтяжній нитці, що провисає під дією сили тяжіння.

Розглянемо класичну задачу про провисання нерозтяжної нитки. На площині XY між точками $A(x_0, y_0)$ і $B(x_1, y_1)$ протягнута (провисає) нитка довжини L.



Якщо нитка і всі прикладені до неї сили знаходяться в площині XY, то рівняння рівноваги мають вигляд

$$\frac{d}{ds}\left(T(s)\frac{dx}{ds}\right)+Q_x(s)=0 \qquad \text{(сума проекцій сил на горизонтальну вісь)}$$

$$\frac{d}{ds}\left(T(s)\frac{dy}{ds}\right)+Q_y(s)=0 \qquad \text{(сума проекцій сил на вертикальну вісь)} \qquad (1)$$

$$\left(\frac{dx}{ds}\right)^2+\left(\frac{dy}{ds}\right)^2=1, \qquad \text{(умова, що } s \text{ натуральний параметр)}$$

де s - натуральний параметр кривої $x(s), y(s)$, по якій провисає нитка, $T(s)$ - натяг нитки, спрямований вздовж дотичної до її траєкторії, $\boldsymbol{Q}=(Q_x, Q_y)$ - вектор розподілених зовнішніх сил (сила, віднесена до одиниці довжини нитки).

У випадку дії лише сили тяжіння горизонтальної складової зовнішніх сил немає і тому $Q_x \equiv 0$. Вертикальна складова дорівнює вазі одиниці довжини нитки і тому вона стала, тобто $Q_y(s)=-q=Const$ (вісь OY спрямована вгору). Тоді з першого рівняння (1) випливає, що $T\frac{dx}{ds}=H(=Const)$ або $T(s)=H\frac{ds}{dx}$.

Підставляючи це в друге рівняння (1), отримуємо $\frac{d}{ds}\left(\frac{ds}{dx}\frac{dy}{ds}\right)=\frac{q}{H}$ або

$\frac{d}{ds}\frac{dy}{dx}=a$, де $a=q/H$. Тоді $\frac{d}{dx}\left(\frac{dy}{dx}\right)\frac{dx}{ds}=a$. З третього рівняння (1) маємо $ds=\sqrt{1+y_x'^2}\,dx$ і останнє співвідношення набуває вигляду $y''_{xx}=a\sqrt{1+y_x'^2}$. Сюди не входить шукана функція $y(x)$, тому порядок рівняння можна понизити і воно розв'язується. Але ми цього робити не станемо, бо нам потрібно дослідити отриману крайову задачу

$$y''_{xx}=a\sqrt{1+y_x'^2},\ y(x_0)=y_0,\ y(x_1)=y_1 \qquad (2)$$

з умовою

$$\int_{x_0}^{x_1}\sqrt{1+y_x'^2}\,dx=L, \qquad (3)$$

щоб продемонструвати деякі можливості COMSOL Multiphysics.

Розв'язок 1. За допомогою майстра створіть безрозмірну 1D модель задачі

File -> New -> Model Wizard -> 1D -> Mathematics - PDE Interfaces - General Form PDE (g) -> Stationary

Untitled.mph (root): `Unit System` – <> = None

Задайте параметри моделі.

Global Definitions : RMB > Parameters

Заповніть таблицю параметрів так, як показано на рисунку.

Parameters

Name	Expression	Value
dltx	1	1
x0	0	0
x1	x0+dltx	1
y0	0	0
y1	1	1
L	2	2

Геометрія моделі складається з відрізка $[x_0, x_1]$.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1) :

`Interval` – Left endpoint, Right endpoint = `x0,x1`

LMB > Build All Objects

На створеній геометрії генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – Element Size = `Extra fine`

LMB> Build All

Для зручності створіть в моделі додаткову змінну, яка обчислюватиметься за формулою $u_1=\sqrt{1+u_x'^2}$.

Definitions: RMB > Variables

Definitions – Variables 1:

`Geometric Entity Selection`–Geometric entity level = `Entire model`

`Variables` – Name, Expression = `u1, sqrt(1+ux^2)`

Створіть оператор інтегрування по відрізку $[x_0, x_1]$.

Definitions -> **Coupling - Component Coupling** – Integration

Component 1 - Definitions - Integration 1 (intop1):

Source Selection – **Geometric entity level** = Domain

– **Selection -** < > = 1 (*відрізок* [0,1])

Ім'я оператора intop1 в полі **Operator name** залиште без змін.

Задайте диференціальне рівняння. Перевірте , що в ПВ вузла **General Form PDE (g) - General Form PDE 1** в секції вибору **Domain Selection** стоїть одиниця. В розділі **Equation** наведено рівняння в загальній (генеральній) формі

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \Gamma = f \ .$$

Параметри рівняння задайте в полях нижньої частини цієї ПВ.

General Form PDE (g) - General Form PDE 1:

Conservative Flux – **Γ** = ux

Source Term – **f** = u1*a

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Mass Coefficient – $\mathbf{e_a}$ = 0

Цим визначається рівняння $u''_{xx} = a\sqrt{1+u'^2_x}$.

Задайте граничні умови.

Physics -> **Boundary** – **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** < > = Manual,1 (*лівий кінець відрізка*)

Dirichlet Boundary Condition – **r** = y0

Physics -> **Boundary** – **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection,** < > = Manual,2 (*правий кінець відрізка*)

Dirichlet Boundary Condition – **r** = y1

Щоб нитка мала задану довжину потрібно підібрати параметр *a*. Для цього в моделі створіть додаткове рівняння.

Physics -> **Global** -> **Global** – Global Equations

General Form PDE (g) - Global Equations 1:

Global Equations – **Name** = a

– **f(u,ut,utt,t)** = L-intop1(u1)

– **Initial value** = 1

Розуміти рівняння $L - \int_{x_0}^{x_1} \sqrt{1+u'^2_x}\, d\,x = 0$ потрібно наступним чином. Задача буде розв'язуватися відносно невідомої *a* одночасно з рівнянням у вузлі **General Form PDE 1** доки вираз в комірці **f(u,ut,utt,t)** не буде дорівнювати нулю, тобто доки довжина кривої $u(x)$ не стане заданою.

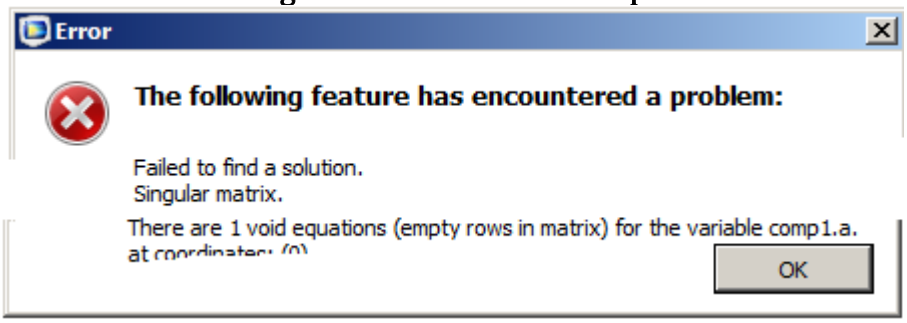
Таким чином, в нашій задачі створено дві шукані величини: u та a. При чому u – це полева змінна (функція від *x*), а *a* – скаляр.

Виконайте обчислення.

`Home -> Study` - Compute

Скоріш за все ви отримаєте повідомлення

The following feature has encounted a problem: ....



Помилка зумовлена звичайною проблемою чисельного алгоритму, що використовується при розв'язанні методом множників Лагранжа нелінійних задач, які містять глобальні або розподілені обмеження. Матриця жорсткості скінченних елементів може виявитися виродженою для деяких пробних розв'язків. В цьому випадку необхідно задати гарне початкове наближення. За замовчуванням використовувалося нульове початкове наближення, тобто $u_0(x)=0$. Змінимо його на параболу, яка проходить крізь кінцеві точки.

Initial Values 1:

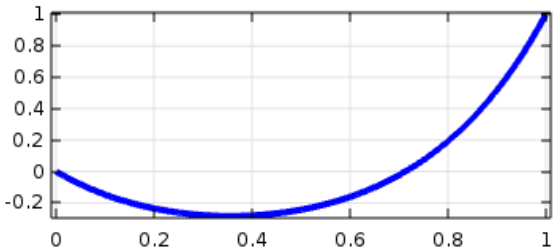
`Initial Values` – `Initial Value for u:` – `u` =

`y0+(y1-y0)/(x1-x0)*(x-x0)+(x-x0)*(x-x1)`

Знову виконайте обчислення

`Home -> Study` - Compute

і в гілці **1D Plot Group 1** отримайте графік ланцюгової кривої, що проходить крізь задані точки.



Надрукуйте значення параметра *a*, а також довжину кривої. Для параметра *a* в гілці **Derived Values** вже створено вузол **State variable a**.

Results - Derived Values - State variable a:

LMB > Evaluate – New Table

Для обчислення довжини ланцюгової лінії створіть вузол `Global Evaluation`.

`Results -> Derived Values` – GlobalEvaluation

Results - Derived Values - Global Evaluation 2:

`Expression` – Expression = `intop1(u1)`

LMB: Evaluate – Table 1

State variable a	intop1(u1)
3.8266	2.0000

Як бачите, довжина кривої дорівнює бажаному значенню `2.0000`.

У розглянутої задачі існує формульний розв'язок [5]

$$y = \alpha \cdot \mathrm{ch}\frac{x-\beta}{\alpha} - \lambda, \qquad (4)$$

де параметри α, β, λ знаходяться з системи алгебраїчних рівнянь

$$y_0 = \alpha \cdot \mathrm{ch}\frac{x_0-\beta}{\alpha} - \lambda, \quad y_1 = \alpha \cdot \mathrm{ch}\frac{x_1-\beta}{\alpha} - \lambda, \; \alpha \cdot \left(\mathrm{sh}\frac{x_1-\beta}{\alpha} - \mathrm{sh}\frac{x_0-\beta}{\alpha}\right) = L. \qquad (5)$$

Порівняємо його з розв'язком, отриманим методом СкЕ. Для цього розв'яжемо систему (5) в COMSOL і в гілці **1D Plot Group 1** добудуємо графік функції (4).

Додайте в компонент «фізику» з `Global ODEs and DAEs` інтерфейсом.

Component 1: RMB > Add Physics = `Add Physics` -> Mathematics – ODE and DAE Interfaces – Global ODEs and DAEs (ge) -> X

В ПВ створеного вузла **Global Equations 1** заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

`Global Equations` – Name = `alph`

- **f(u, ut, utt, t)** =`y0-alph*cosh((x0-bet)/alph)+lmbd`
- **Initial value** =`1`

`Global Equations` – Name = `bet`

- **f(u, ut, utt, t)** =`y1-alph*cosh((x1-bet)/alph)+lmbd`
- **Initial value** =`0`

`Global Equations` – Name = `lmbd`

- **f(u, ut, utt, t)** = `alph*(sinh((x1-bet)/alph)-sinh((x0-bet)/alph))-L`
- **Initial value** =`0`

Global Equations

$f(u,u_t,u_{tt},t) = 0, \; u(t_0) = u_0, \; u_t(t_0) = u_{t0}$

Name	f(u,ut,utt,t)	Initial value	Initial
alph	y0-alph*cosh((x0-bet)/alph)+lmbd	1	0
bet	y1-alph*cosh((x1-bet)/alph)+lmbd	0	0
lmbd	alph*(sinh((x1-bet)/alph)-sinh((x0-bet)/alph))-L	0	0

Заново обчисліть модель. Розв'язок першої фізики не зміниться, але одночасно буде розв'язана система рівнянь в гілці **Global ODEs and DAEs(ge)**. Знайдені з системи параметри можна використати в формулі (4) для побудови графіка.

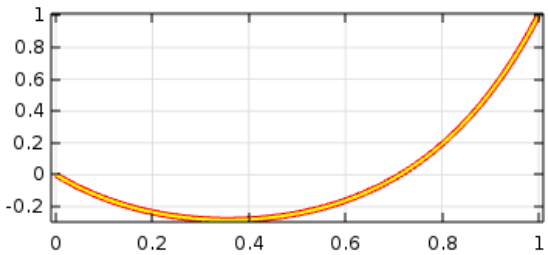
1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 2 :

`y-Axis Data` – Expression = `alph*cosh((x-bet)/alph)-lmbd`

Стиль кривої налаштуйте на свій розсуд.

LMB > Plot



Як бачите, криві співпадають. Збережіть модель в файл, але не закривайте його.

У розглянутого способу дослідження моделі є недолік – потрібно гарно підібрати початкове наближення. Це не завжди можливо. Одним з поширених способів подолання такої проблеми є використання розв'язку задачі без обмеження в якості початкового наближення для задачі з обмеженням. В COMSOL Multiphysics це можна зробити, створивши два кроки в рамках одного дослідження і прибравши обмеження на першому кроці. Розв'язок, отриманий на першому кроці, можна використати в якості входових даних для другого кроку.

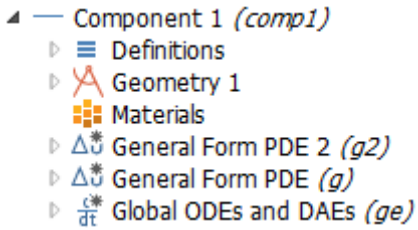
Розв'язок 2. План побудови моделі буде полягати в наступному. Створюється дві фізики (третя «контрольна» фізика `Global ODEs and DAEs(ge)` залишиться з попереднього прикладу). В першій фізиці розв'язуватиметься задача без обмежень $u''_{xx} = a\sqrt{1+u'^2_x}$ при деякому (добре вгаданому) a. В другій – буде розв'язуватися та ж задача але вже з підбором значення a так, щоб довжина кривої була заданою, тобто щоб виконувалося додаткове рівняння $\int_{x_0}^{x_1} \sqrt{1+u'^2_x}\, dx = L$. При цьому початкове наближення для функції $u(x)$ обиратиметься з розв'язку першої задачі.

В нашому випадку друга фізика вже побудована. В ній потрібно буде лише скорегувати початкове значення, взявши його зі спрощеного розв'язку.

Відкрийте файл останньої моделі і додайте в нього ще одну `General Form PDE` фізику.

Component 1: RMB > Add Physics = `Add Physics ->`
Mathematics - PDE Interfaces - General Form PDE (g) **->** X

В ній буде розв'язуватися спрощена задача (2) без умови (3). Оскільки ця задача розв'язуватиметься першою, перемістіть її гілку вище гілки **General Form PDE (g)**. Для цього захопіть створений вузол **General Form PDE 2 (g2)** лівою кнопкою миші (натисніть кнопку і не відпускайте) і перетягніть його вгору. В процесі пересування позиція вузла в дереві позначатиметься червоним відрізком. В результаті гілки фізик розташуються в порядку, зображеному на наступному рисунку.



Розв'язку допоміжної задачі дайте ім'я *w*.

General Form PDE 2 (g2):

`Dependent Variables` – Dependent wariables = `w`

Додайте в компонент допоміжну змінну *w*1.

Component 1 - Definitions – Variables 1:

`Variables` – Name, Expression = `w1, sqrt(1+wx^2)`

Налаштуйте рівняння створеної фізики.

General Form PDE 2 (g2) - General Form PDE 1:

Conservative Flux – **Γ** = wx

Source Term – **f** = w1*a0

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Mass Coefficient – $\mathbf{e_a}$ = 0

Для цієї задачі в рівнянні (2) використано фіксоване значення параметра $a = a_0$, яке задайте в таблиці параметрів.

Global Definitions - Parameters:

Parameters – **Name, Expression** = a0 ,3

В результаті ми матимемо рівняння $u''_{xx} = a_0\sqrt{1+u'^2_x}$.

Задайте граничні умови.

Physics -> **Boundary** – **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** < > = Manual,1 (*лівий кінець відрізка*)

Dirichlet Boundary Condition – **r** = y0

Physics -> **Boundary** – **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection,** < > = Manual,2 (*правий кінець відрізка*)

Dirichlet Boundary Condition – **r** = y1

Таким чином, в гілці **General Form PDE 2 (g2)** розв'язуватиметься задача

$$u''_{xx} = a_0\sqrt{1+u'^2_x},\ u(x_0) = y_0,\ u(x_1) = y_1.$$

Скорегуйте налаштування фізики **General Form PDE (g)**. В ній потрібно вказати, що в якості початкового значення використовуватиметься розв'язок першої задачі, тобто величина *w*.

General Form PDE (g) - Initial Values 1:

Initial Values – **Initial Value for u:** – **u** = w

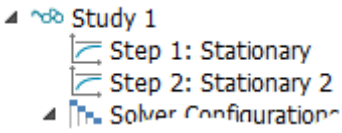
Останнім ми сказали, що початкові значення для u співпадатимуть з величиною *w* (але щоб так було цього замало, бо не вказано звідкіля береться *w*).

Інші зміни в цю фізику не вносимуться.

Залишилось налаштувати гілку **Study 1**, в якій потрібно створити два кроки дослідження.

Study 1: RMB > Study Steps - Stationary – Stationary

Як наслідок, гілка дослідження **Study 1** матиме наступний вигляд.

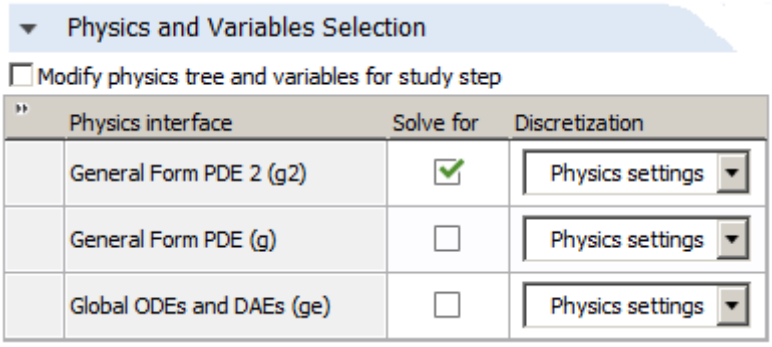


В кожному з вузлів-кроків потрібно вказати фізику, яку він обчислюватиме. Другому кроку потрібно також вказати, що його початкове наближення братиметься з першого кроку.

Step 1: Stationary:

Physics and Variables Selection (таблиця)-

General Form PDE 2 (g2) – Solve for = `True`
General Form PDE (g) – Solve for = `False`
Global ODEs and DAEs (ge) – Solve for = `False`

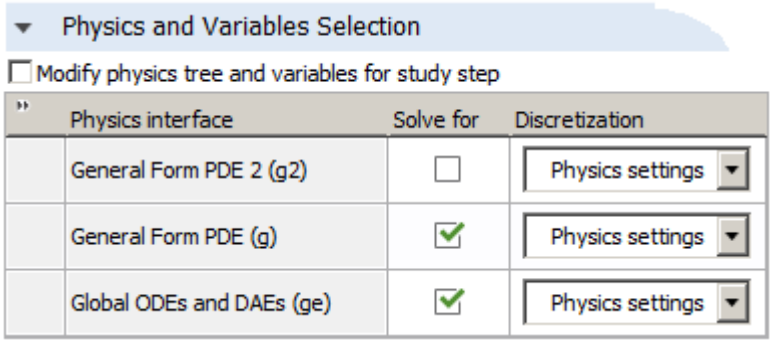


Останні налаштування означають, що на першому кроці розв'язуватися буде лише перша фізика **General Form PDE 2 (g2)** .

Step 2: Stationary 2:

`Physics and Variables Selection (таблиця)-`

General Form PDE 2 (g2) – Solve for = `False`
General Form PDE (g) – Solve for = `True`
Global ODEs and DAEs (ge) – Solve for = `True`



Задані налаштування вказують, що на другому кроці розв'язуватися будуть фізики **General Form PDE (g)** і **Global ODEs and DAEs (ge)**.

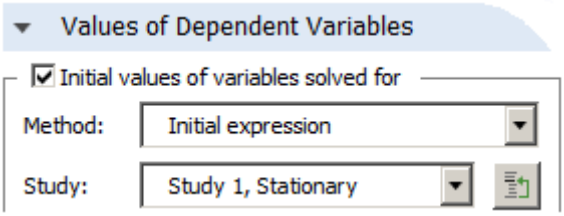
Щоб початкове наближення вибиралося по результатам роботи першого кроку, в ПВ другого кроку треба встановити кілька прапорців.

`Values of Dependent Variables –`

– **Initial values of variables solved for** = `True`
– **Method** = `Initial expression`

Обране в списку **Method** значення `Initial expression` визначає, що дані попереднього кроку будуть використовуватися для початкових наближень поточного кроку. В списку **Study** визначається з якого дослідження ці дані будуть обиратися.

– **Study** = `Study 1: Stationary`



Нагадаємо, що у вузлі **General Form PDE (g) - Initial Values 1** для початкового наближення розв'язку *u* вказано ім'я *w*. Налаштування розділу **`Values of Dependent Variables`** визначають, що це ім'я буде братися з розв'язку першого кроку.

Виконайте обчислення

`Home` **`-> Study`** - Compute

і отримайте збіжні графіки наближеного і «аналітичного» розв'язків.

Цікаво подивитися на графік початкового наближення другого кроку. Для цього в гілці **1D Plot Group 1** добудуйте графік величини *w*, а також налаштуйте легенди.

1D Plot Group 1 :

`Legend` – **Show legends, Position** = `True, Upper left`

1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 3 :

`y-Axis Data` – **Expression** = `w`

`Legends` – **Show legends, Legends, <>** = `True, Manual,w`

Стиль кривої налаштуйте на свій розсуд.

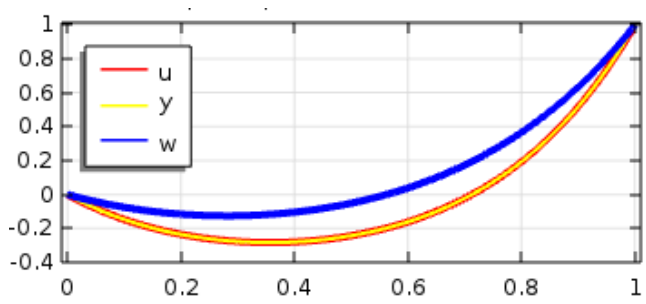
1D Plot Group 1 - Line Graph 1 :

`Legends` – **Show legends, Legends, <>** = `True, Manual,u`

1D Plot Group 1 - Line Graph 2 :

`Legends` – **Show legends, Legends, <>** = `True, Manual,y`

LMB > Plot



5.4.14. Скін ефект.

Розглянемо нескінченний однорідний циліндричний дріт, по якому тече змінний струм. Повний струм $I = I_0 e^{i\omega t}$, що проходить крізь переріз дроту, відомий. Потрібно дослідити розподіл густини $\boldsymbol{j}$ електричного струму по перерізу дроту. Таку двовимірну задачу ми вже розв'язували в п.3.11. Тут ми розглянемо її як одновимірну.

Враховуючи матеріальні рівняння $\boldsymbol{B} = \mu\mu_0 \boldsymbol{H}$, $\boldsymbol{D} = \varepsilon\varepsilon_0 \boldsymbol{E}$ і закон Ома $\boldsymbol{j} = \sigma \boldsymbol{E}$, а також нехтуючи струмом зміщення $\partial \boldsymbol{D}/\partial t$ в порівнянні зі струмом провідності $\boldsymbol{j}$, запишемо рівняння Максвела: теорему о циркуляції магнітного поля $\nabla \times \boldsymbol{H} = \boldsymbol{j} + \partial \boldsymbol{D}/\partial t$ у вигляді

$$\nabla \times \boldsymbol{H} = \sigma \boldsymbol{E},$$

і закон індукції Фарадея $\nabla \times \boldsymbol{E} = -\partial \boldsymbol{B}/\partial t$ у вигляді

$$\nabla \times \boldsymbol{E} = -\mu\,\mu_0\,\partial\,\boldsymbol{H}/\partial t\,.$$

Вважаючи процес стаціонарним, тобто залежним від часу по закону $e^{i\omega t}$, після скорочення на множник $e^{i\omega t}$, ці рівняння набувають вигляду [4]

$$\nabla \times \boldsymbol{H} = \sigma\,\boldsymbol{E}\,, \tag{1}$$

$$\nabla \times \boldsymbol{E} = -i\,\omega\,\mu\,\mu_0\,\boldsymbol{H}\,, \tag{2}$$

де застосовано однакові позначення для векторних амплітуд $\boldsymbol{H}(x,y,z)$, $\boldsymbol{E}(x,y,z)$ і векторних полів $\boldsymbol{H}(x,y,z,t) = \boldsymbol{H}(x,y,z)\,e^{i\omega t}$, $\boldsymbol{E}(x,y,z,t) = \boldsymbol{E}(x,y,z)\,e^{i\omega t}$.

Введемо циліндричну систему координат (r,φ,z) так, щоб вісь z співпадала з віссю дроту.

Нагадаємо формулу для обчислення ротора вектор-функції $\boldsymbol{U} = (U_r, U_\varphi, U_z)$ в циліндричних координатах

$$\nabla \times \boldsymbol{U} \equiv \mathrm{rot}\,\boldsymbol{U} = \boldsymbol{e}_r\left(\frac{1}{r}\frac{\partial U_z}{\partial \varphi} - \frac{\partial U_\varphi}{\partial z}\right) + \boldsymbol{e}_\varphi\left(\frac{\partial U_r}{\partial z} - \frac{\partial U_z}{\partial r}\right) + \boldsymbol{e}_z\left(\frac{1}{r}\frac{\partial r U_\varphi}{\partial r} - \frac{1}{r}\frac{\partial U_r}{\partial \varphi}\right). \tag{3}$$

Через осьову симетрію всі величини залежать лише від радіальної змінної *r*. Оскільки вектор $\boldsymbol{E}$ спрямований уздовж осі *z*, то з (3) в осесиметричному випадку для $\boldsymbol{E} = (0, 0, E_z(r))$ матимемо

$$\nabla \times \boldsymbol{E} = 0 \cdot \boldsymbol{e}_r - \frac{\partial E_z}{\partial r} \cdot \boldsymbol{e}_\varphi + 0 \cdot \boldsymbol{e}_z\,.$$

Для $\boldsymbol{H} = (H_r, H_\varphi, H_z)$ з (2) випливає $H_r = 0$, $H_z = 0$ і

$$\frac{\partial E_z}{\partial r} = i\,\omega\,\mu\,\mu_0\,H_\varphi\,, \tag{4}$$

тобто $\boldsymbol{H} = (0, H_\varphi, 0)$. Тоді з (3) отримуємо

$$\nabla \times \boldsymbol{H} = 0 \cdot \boldsymbol{e}_r + 0 \cdot \boldsymbol{e}_\varphi + \left(\frac{1}{r}\frac{\partial r H_\varphi}{\partial r}\right) \cdot \boldsymbol{e}_z\,,$$

і рівняння (1) переписується у вигляді

$$\frac{1}{r}\frac{\partial r H_\varphi}{\partial r} = \sigma\,E_z\,. \tag{5}$$

Виключаючи звідси H_φ, знаходимо

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial E_z}{\partial r}\right) = i\,\omega\,\mu\,\mu_0\,\sigma\,E_z\,. \tag{6}$$

Це є комплексним рівнянням, яке описує розповсюдження плоских гармонічних електромагнітних хвиль в провіднику. Воно співпадає з рівнянням (5) п. 3.11, якщо там нехтувати доданком $\varepsilon\,\varepsilon_0\omega^2$ в другому члені (для провідників $\sigma >> \omega\,\varepsilon\,\varepsilon_0$).

Виведемо граничну умову на поверхні дроту при $r = R$. Для цього скористаємося тим, що нам відомий повний струм I_0, протікаючий по дроту.

Запишемо рівняння Максвела (теорему про циркуляцію магнітного поля) в інтегральному вигляді [4]

$$\oint_C H_s d\,s = I_0\,,$$

де C - контур, що охоплює дріт, H_s - тангенціальна складова вектора $\boldsymbol{H}$ на C.

Якщо в якості такого контуру обрати коло $r = R$, то матимемо $\int_0^{2\pi} H_\varphi(R)\,d\,\varphi = \frac{1}{R} I_0$, або $H_\varphi(R) = \frac{1}{2\pi R} I_0$. Використовуючи співвідношення (4), знаходимо граничну умову

$$\left.\frac{\partial E_z}{\partial r}\right|_{r=R} = \frac{i\,\omega\,\mu\,\mu_0}{2\pi R} I_0\,. \qquad (7)$$

Отже ми маємо диференціальне рівняння (6) з граничною умовою (7), і умовою осьової симетрії при $r = 0$. Зверніть увагу на те, що рівняння (6) і умова (7) є комплексними (містять уявну одиницю i). Але COMSOL Multiphysics вільно працює і з комплексними величинами.

Після розв'язання крайової задачі (6), (7) відносно E_z можна обчислити густину струму $\boldsymbol{j} = \sigma\,\boldsymbol{E}$.

Приклад. Матеріалом дроту вважатимемо мідь. Електропровідність міді $\sigma = 57 \cdot 10^6$ S/m, магнітна проникність $\mu\,\mu_0 = 4\pi 10^{-7}$ *гн / м*. В прикладі оберемо $R = 0.001$ *м*, $I_0 = 1$A, $f = 5000$ *герц* (частота струму) і $\omega = 2\pi\,f$.

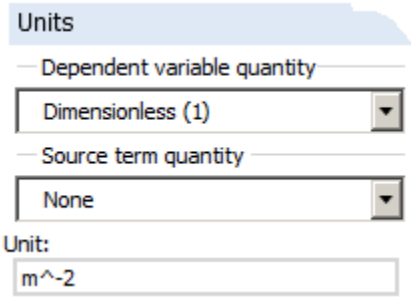
Р о з в ' я з а н н я. Створіть одновимірну осесиметричну модель задачі.

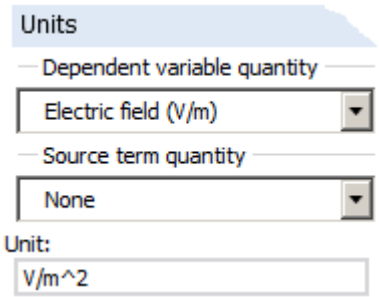
File->New->Model Wizard->1D Axisymmetric->
Mathematics – PDE Interfaces – Coefficient From PDE(c)

Після вибору типу рівняння можна задати одиниці вимірювання. Спочатку у вікні `Review Physics Interface` в розділі **`Units`** вказано, що розв'язок безрозмірний (наступний рисунок ліворуч). Оскільки задача буде розв'язуватися відносно величини u - напруженості електричного поля, яка вимірюється в одиницях *вольт/метр*, то оберемо цю величину (`Electric field`) в випадному списку **Dependent variable quantity**. В випадному списку **Source term quantity** обираються одиниці вимірювання лівої і правої частини досліджуваного рівняння. Якщо їх там немає (одиниць вимірювання), то можна вказати `None` і задати явно одиниці в полі **Unit**, яке розташовано під списком.

Review Physics Interface:

`Units` – **Dependent variable quantity** = `Electric field (V/m)`
– **Source term quantity** = `None`
– **Unit** = `V/m^2` (*наступний рисунок праворуч*)





Натисніть кнопку Study і виберіть тип дослідження `Stationary`.

При побудові моделі нам знадобляться числове значення радіуса дроту R, величина I_0 загального струму, частота f і кутова частота ω струму, магнітна проникності $\mu\mu_0$ і електропровідності σ міді. Задайте їх в таблиці параметрів.

Global Definitions : RMB > Parameters

Заповніть таблицю параметрів так, як показано на рисунку.

Parameters

Name	Expression	Value
R	0.001[m]	0.001 m
f	5000[Hz]	5000 Hz
w	2*pi*f	31416 Hz
mu	4*pi*1E-7[H/m]	1.2566E-6 H/m
sgm	57E6[S/m]	5.7E7 S/m
I0	1.0[A]	1 A

Тут параметр `mu` позначає добуток $\mu_0\,\mu$.

Побудуйте розрахункову область, яка складається з відрізка $[0, R]$ (він представлятиме радіус кола).

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1) :

`Interval` – **Left endpoint, Right endpoint** = `0,R`

LMB > Build All Objects

На створеній геометрії генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – **Element Size** = `Finer`

LMB> Build All

Налаштуйте рівняння. Спочатку перепишемо (6) у вигляді

$$\frac{\partial}{\partial r}\left(r\frac{\partial E_z}{\partial r}\right)=i\,\omega\,\mu\,\mu_0\,\sigma\,r\,E_z.$$

В ПВ вузла **Coefficient Form PDE (c) - CoefficientFromPDE 1** в розділі **`Equation`** наведено загальний вигляд коефіцієнтної форми ДРЧП.

$$e_a\frac{\partial^2 u}{\partial t^2}+d_a\frac{\partial u}{\partial t}+\nabla\cdot\left(-c\nabla u-\alpha u+\gamma\right)+\beta\cdot\nabla u+\mathrm{a}\,u=f\,,\ \nabla=\frac{\partial}{\partial r}.$$

Щоб це рівняння набуло бажаного вигляду потрібно ввести наступні значення коефіцієнтів.

Coefficient Form PDE (c) - CoefficientFromPDE 1:

`Diffusion Coefficient` – **c** = `r`

`Absorption Coefficient` – **a** = `i*w*mu*sgm*r` (`i` – *уявна одиниця*)

Source Term – **f** = 0

Інші коефіцієнти задайте нульовими, тобто $e_a = 0$, $d_a = 0$, $\alpha = 0, \beta = 0$, $\gamma = 0$. Цим задається рівняння (6) у вигляді $\left(-r u_r'\right)_r' + i\omega\mu\mu_0\sigma r u = 0$, і величина u буде репрезентувати амплітуду електричного поля E_z.

Гранична умова в точці $r = 0$ не задається, бо на початку координат вона непотрібна через осесиметричність. Задайте граничну умову (7) на поверхні дроту.

Physics -> **Boundary - Boundaries** – Flux/Source

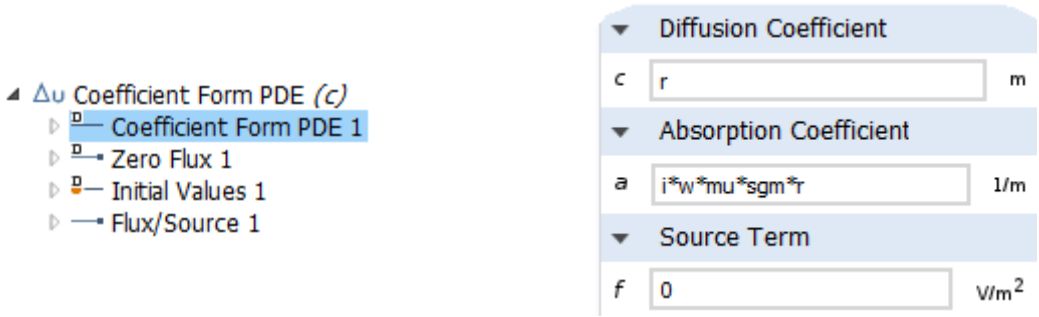
Coefficient Form PDE (c) - Flux/Source 1:

Boundary Selection – **Selection,** < > = Manual,2 (*точка* r=R)

Boundary Flux/Source – **g** = i*w*mu*I0/(2*pi)

Цим задається умова $r u_r'\big|_{r=R} = i\omega\mu\mu_0 I_0 / 2\cdot\pi$, яка співпадає з (7).

Зауваження. У вузлах **Coefficient Form PDE 1** і **Flux/Source 1** праворуч від полів коефіцієнтів вказуються розмірності.



Система автоматично перевіряє одиниці вимірювання виразів в цих полях. Якщо введений вираз має іншу розмірність, то формула в полі стає помаранчевою.

Тепер обчисліть модель

Home -> **Study** – Compute

Треба пам'ятати, що рівняння (6) і умова (7) є комплексними. Тому нас цікавить лише дійсна частина розв'язку. Але система COMSOL Multiphysics завжди будує графік дійсної частини величин. Отже побудована в гілці **1D Plot Group 1** крива є графіком величини real(u). Нас цікавить розподіл густини струму по перерізу дроту, яка обчислюється за формулою $j = \sigma\cdot\mathrm{real}(u)$. Тому неважко цей графік модернізувати.

1D Plot Group 1 - Line Graph 1 :

y-Axis Data – **Expression** = real(u)*sgm

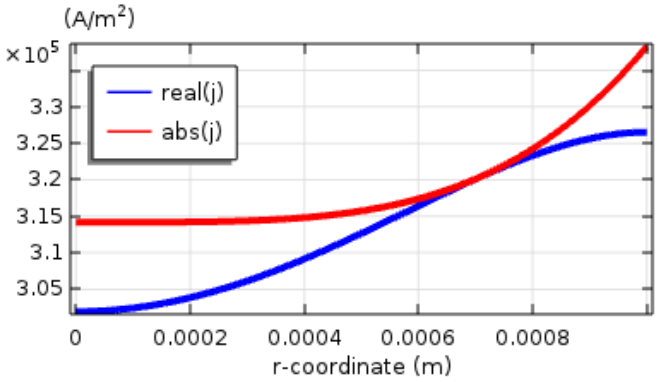
Додайте сюди графік модуля густини струму, і самостійно налаштуйте стиль кривих та легенди графіків.

1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 2 :

y-Axis Data – **Expression** = abs(u)*sgm

LMB > Plot



Оскільки створювалась `1D Axisymmetric` модель, то було автоматично побудовано поверхневий графік напруженості по перерізу дроту. Для нього в гілці **Data Sets** створено джерело даних **Revolution 1D 1**. Скорегуйте його налаштування.

Results - Data Sets - Revolution 1D 1:

`Revolution Layers` – **Start angle, Revolution angle** = `0, 360`

Тепер перебудуйте поверхневий графік так, щоб він зображав густину струму.

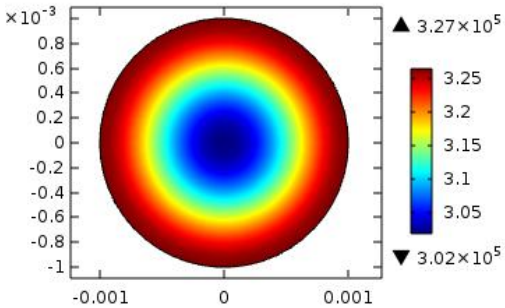
2D Plot Group 2:

`Color Legend` – **Show maximum and minimum values** = `True`

2D Plot Group 2 - Surface 1:

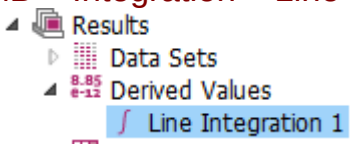
`Expression` – **Expression** = `real(u)*sgm`

LMB > Plot



Залишилося перевірити, що сумарний струм дорівнює 1А. Для цього потрібно проінтегрувати густину струму по перерізу дроту.

Results – Derived Values: RMB > Integration – Line Integration



Derived Values - Line Integration 1:

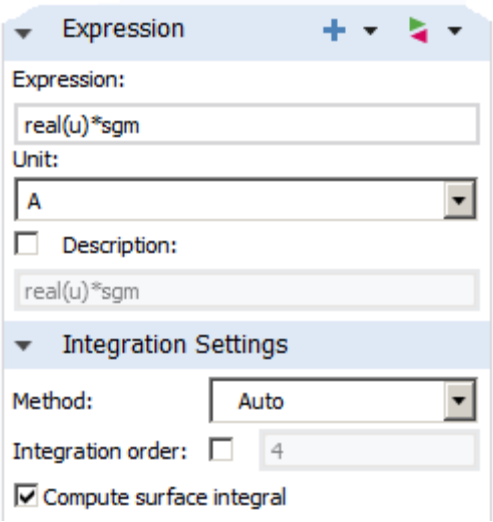
`Selection` – < > = `1` (*відрізок* [0,R])

Введіть вираз, який буде інтегруватися.

`Expression` – **Expression** = `real(u)*sgm`

Щоб інтегрування виконувалося по площі обертання відрізка (а не по відрізку) потрібно встановити наступний прапорець.

`Integration Settings` – **Compute surface integral**= `True`



LMB > Evaluate – New Table

real(u)*sgm (A)
1.00000

Надруковане значення представляє інтеграл $\iint_S \sigma \operatorname{Re}(u)\, d\, s$, тобто амплітуду загального струму через переріз дроту.

5.4.15. Осесиметричний нагрів і деформація дроту.

Модель, яку ми збираємося створити, буде аналізувати мідний циліндричній дріт, по якому протікає постійний електричний струм. Струм, розподілений рівномірно по перерізу дроту, і призводить до нагрівання, яке викликає температурну деформацію. Задача моделювання – обчислити цю деформацію.

Процес нагрівання описується рівнянням теплопровідності, а температурні деформації – рівняннями теорії термопружності. Оскільки ми розглядатимемо осесиметричну стаціонарну задачу, то ці рівняння виродяться в звичайні диференціальні рівняння. Після їх розв'язання ми отримаємо величини температури і радіального зміщення відповідно.

Введемо циліндричну систему координат (r, φ, z) так, щоб вісь z співпадала з віссю дроту. Нехай густина струму $\boldsymbol{j}$ є сталою $\boldsymbol{j} = \boldsymbol{k} \cdot j_z$. Тоді потужність q джерел тепла, яке виділяється при проходженні струму, дорівнює $q = \boldsymbol{j} \cdot \boldsymbol{E} = j_z^2 / \sigma$. Якщо повний струм I_0 відомий, то $j_z = I_0 / \pi R^2$ і $q = I_0^2 / \left(\sigma\, \pi^2 R^4\right)$, де σ - електропровідність матеріалу дроту.

Для дослідження тепловиділення потрібно розв'язати рівняння теплопровідності $c \rho T_t' = \lambda \Delta T + q$, де λ – коефіцієнт теплопровідності, c – питома теплоємність, ρ – густина речовини. Зважаючи на стаціонарність і осьову симетрію, рівняння набуває вигляду ЗДР

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) = -\frac{q}{\lambda} \quad \text{або} \quad \frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) = -\frac{q \cdot r}{\lambda}. \qquad (1)$$

Вважатимемо, що на поверхні дроту виконується теплообмін з навколишнім середовищем по закону Ньютона $\lambda \cdot \operatorname{grad} T = \gamma\,(T_0 - T)$, де γ - коефіцієнт

теплообміну, T_0 - температура навколишнього середовища. В одновимірному випадку ця умова набуває виду

$$\lambda \cdot T_r'(R) = \gamma\ (T_0 - T(R)).$$

Тепловиділення призводить до температурних деформацій. Якщо позначити компоненту переміщення в радіальному напрямку через *u*, то вона у осесиметричному випадку задовольнятиме рівнянню [6]

$$\frac{d^2u}{d\,r^2} + \frac{1}{r}\frac{d\,u}{d\,r} - \frac{u}{r^2} = (1+\nu)\alpha\frac{d\,T}{d\,r}, \tag{2}$$

де ν - коефіцієнт Пуассона речовини, α - коефіцієнт теплового розширення.

Переміщення *u* в центрі дроту повинно дорівнювати нулю $u(0) = 0$.

У випадку симетричного розподілу температури відносно центра радіальна і кутова деформації дорівнюють

$$\varepsilon_r = d\,u / d\,r,\ \varepsilon_\varphi = u / r, \tag{3}$$

а радіальне напруження обчислюється за формулою [6]

$$\sigma_r = \frac{E}{1-\nu^2}[\varepsilon_r + \nu\,\varepsilon_\varphi - (1+\nu)\alpha(T - T_0)] \tag{4}$$

Тут через *E* позначений модуль пружності речовини, а $T - T_0$ представляє температуру, відлічувану від деякого початкового незмінного по всьому тілу значення, яке ми узяли співпадаючим з температурою навколишнього середовища T_0.

На поверхні дроту при $r = R$ радіальне напруження σ_r повинно дорівнювати нулю. Отже матимемо

$$\frac{d\,u}{d\,r} + \nu\frac{u}{r} - (1+\nu)\alpha(T - T_0)\big|_{r\,=\,R} = 0. \tag{5}$$

Це є другою граничною умовою для рівняння (2).

Таким чином, для дослідження теплового розширення дроту треба розв'язати крайові задачі:

1. Рівняння (1)

$$\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right) = -\frac{q\,r}{\lambda},\ \text{де}\ \ q = I_0^2 / \left(\sigma\ \pi^2\ R^4\right) \tag{1}$$

з граничними умовами

$$T_r'(0) = 0 \quad \text{і} \quad \lambda \cdot T_r'(R) = \gamma\ (T_0 - T(R)). \tag{6}$$

2. Рівняння (2), яке запишемо у вигляді

$$\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) - \frac{u}{r} = (1+\nu)\alpha\frac{d\,T}{d\,r}\cdot r \tag{2}$$

з граничними умовами (умову (5) домножаємо на *r*)

$$u(0) = 0\ \ \text{і}\ \ r\frac{d\,u}{d\,r} + \nu\,u - (1+\nu)\alpha(T - T_0)\cdot r\,\big|_{r\,=\,R} = 0. \tag{7}$$

Задачі пов'язані, бо у вільному члені рівняння (2) і правій граничній умові (7) використовується розв'язок задачі (1), (6). Але розв'язок другої задачі не

впливає на розв'язок першої. Отже їх можна розв'язувати послідовно. Для цього в COMSOL Multiphysics у вузлі Study потрібно створити два кроки дослідження.

Приклад. Будемо вважати, що дріт мідний. В прикладі оберемо $R = 0.001\,м$, $I_0 = 100\,\text{A}$. Електропровідність міді $\sigma = 57 \cdot 10^6\,\text{S/m}$, коефіцієнт теплопровідності $\lambda = 401\,вт/м \cdot К$, коефіцієнт теплового розширення $\alpha = 1.65 \cdot 10^{-5}\,K^{-1}$. Коефіцієнт теплообміну вважатимемо незалежним від температури. Зазвичай він змінюється в діапазоні від 10 до $100\,вт/м^2 \cdot К$. Візьмемо його $\gamma = 100\,вт/м^2 \cdot К$. Коефіцієнт Пуансона міді $\nu = 0.33$. Початкове значення, від якого будемо відлічувати *T*, покладемо співпадаючим з температурою навколишнього середовища $T_0 = 273\,K$.

Розв'язання. За допомогою майстра створіть одновимірну осесиметричну модель з двома рівняннями.

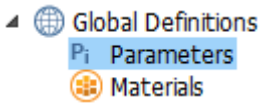
File –> New –> Model Wizard –> 1D Axisymmetric –> Mathematics - PDE Interfaces – Coefficient From PDE(c) + Variables = T –> PDE Interfaces - General Form PDE (g) –> Stationary

При генеруванні шаблона моделі ми відразу створили дві фізики/рівняння. Одиниці вимірювання шуканих величин *T* і *u* визначимо пізніше.

В гілці **Global Definitions** створіть вузол параметрів.

Global Definitions: RMB > Parameters

Задайте параметри відповідно до наступної таблиці.



Parameters

Name	Expression	Value
R	0.001[m]	0.001 m
sgm	57E6[S/m]	5.7E7 S/m
I0	100[A]	100 A
lmd	401[W/m/K]	401 W/(m·K)
alph	1.65E-5[K^-1]	1.65E-5 1/K
gam	100[W/m^2/K]	100 W/(m²·K)
T0	273[K]	273 K
nu	0.33	0.33
q	I0^2/(sgm*pi^2*R^4)	1.7776E7 W/m³
E	100000[MPa]	1E11 Pa

Переріз провідника утворюється обертанням відрізка [0,*R*] навколо осі Z. Створіть цей відрізок.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1) :

Interval – **Left endpoint, Right endpoint** = 0,R

LMB > Build All Objects

Генеруйте СкЕ сітку. Для нашої задачі сітка Extremely fine буде занадто великою. Тому побудуємо скінченно-елементну сітку з достатньо дрібними елементами, задавши їх максимальний розмір.

Mesh 1:

Mesh Settings – **Sequence type** = User-controlled mesh

Mesh 1 - Size:

Element Size – **Custom** = True

Element Size Parameters – **Maximum element size** = 1E-6

LMB> Build All

Налаштуйте першу фізику, але спочатку для неї задайте одиниці вимірювання.

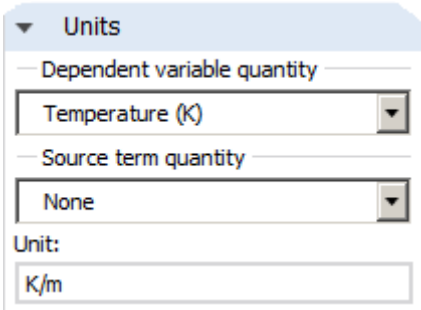
Coefficient Form PDE (c):

Units – **Dependent variable quantity** =Temperature (K)

– **Source term quantity** =None

– **Unit** =K/m

В полі **Dependent variable quantity** обирається одиниця вимірювання шуканої величини (в нашому випадку *T*). В полі **Source term quantity** обирається одиниця вимірювання лівої і правої частин рівняння. Якщо обрано None, то одиниця задається явно в полі **Unit**.



У дочірньому вузлі задайте коефіцієнти рівняння.

Coefficient Form PDE (c) - Coefficient Form PDE 1:

Diffusion Coefficient – **c** = r

Absorption Coefficient – **a** = 0

Source Term – **f** = q*r/lmd

Решту коефіцієнти покладіть нулями: e_a=0, d_a=0, α=0, β=0, γ=0. Цим визначається рівняння (1).

Задайте граничні умови першої фізики. Якщо другу умову (6) помножити на *R*, то вона набуде вигляду $R \cdot T_r'(R) = (\gamma R / \lambda)(T_0 - T(R))$. Обмеження, яке встановлює вузол Flux/Source, на правому кінці $r = R$ має вигляд $cT_r' = g - qT$. У нас $c = r = R$, тобто $R \cdot T_r'(R) = g - qT$. Тому потрібно задавати $g = \gamma R\, T_0 / \lambda$ і $q = \gamma R / \lambda$.

Physics -> **Boundary - Boundaries** – Flux/Source

Coefficient Form PDE (c) - Flux/Source 1:

Boundary Selection – **Selection,** < > = Manual,2 (*точка x=R*)

Boundary Flux/Source – **g** = gam*R*T0/lmd

Boundary Absorption/Impedance Term – **q** = gam*R/lmd

Граничну умову в точці $r = 0$ у осесиметричному випадку задавати непотрібно.

У вузлі **Initial Values 1** задайте початкове значення температури. В стаціонарній задачі воно розглядається як початкове наближення.

Coefficient Form PDE (c) - Initial Values 1:

Initial Values – **Initial value for T** = T0

Налаштуйте гілку другої фізики. Спочатку задайте одиниці вимірювання шуканої величини *u* і лівої та правої частин рівняння.

General Form PDE (g):

Units – **Dependent variable quantity** =Displacement field(m)

– **Source term quantity** =Dimensionless(1)

Тепер задайте коефіцієнти рівняння термопружності (2).

General Form PDE (g) - General Form PDE 1:

Conservative Flux – $\mathbf{\Gamma}$ = r*ur

Source Term – **f** = u/r+(1+nu)*alph*r*Tr

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Mass Coefficient – $\mathbf{e_a}$ = 0

Цим задається рівняння (2).

Задайте граничні умови другої задачі.

Physics -> **Boundary - Boundaries** – Flux/Source

Coefficient Form PDE (c) - Flux/Source 1:

Boundary Selection – **Selection,** < > = Manual,2 (*точка* x=R)

Boundary Flux/Source – **g** = -(1+nu)*alph*r*(T-T0)

Boundary Absorption/Impedance Term – **q** = -nu

Оскільки умова Flux/Source має вид $-\boldsymbol{n}\cdot\boldsymbol{\Gamma}=g-qu$, а у нас $\boldsymbol{\Gamma}=\boldsymbol{e}_r\cdot r u_r'$ і $\boldsymbol{n}=\boldsymbol{e}_r\big|_{r=R}$, то матимемо $-r u_r'\big|_{r=R}=g-qu\big|_{r=R}$. Щоб цей вираз набув вигляду другої умови (7), потрібно було покласти $g=-(1+\nu)\alpha(T-T_0)\cdot r$ і $q=-\nu$.

Граничну умову в точці $r=0$ у осесиметричному випадку можна не задавати (задавання умови Діріхле $u(0)=0$ розв'язок не поміняє).

Залишилось налаштувати гілку **Study 1**, в якій потрібно створити два кроки дослідження.

Study 1: RMB > Study Steps - Stationary – Stationary

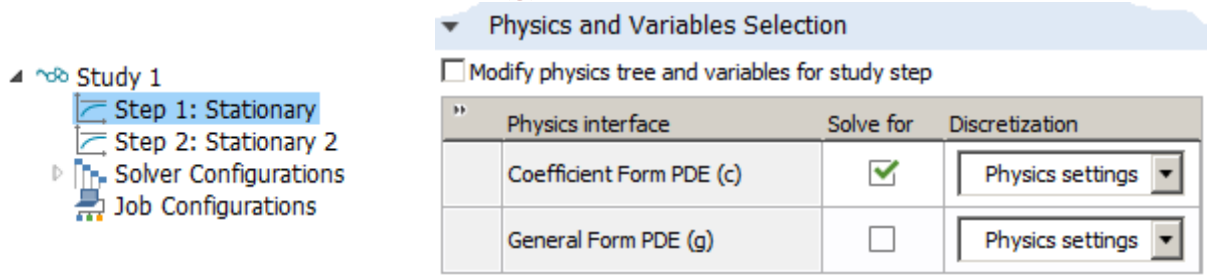
В кожному з вузлів-кроків потрібно вказати фізику, яку він обчислюватиме.

Step 1: Stationary:

Physics and Variables Selection (таблиця)-

Coefficient Form PDE (c) – **Solve for** = True

General Form PDE (g) – **Solve for** = False



Останні налаштування означають, що на першому кроці розв'язуватися буде лише перша фізика **Coefficient Form PDE (c)** .

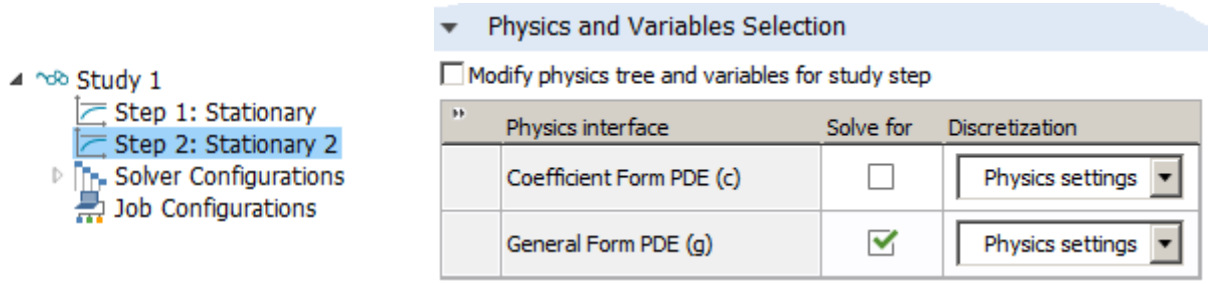
Налаштуйте другий крок дослідження.

Step 2: Stationary 2:

Physics and Variables Selection (таблиця)-

Coefficient Form PDE (c) – **Solve for** = False

General Form PDE (g) – **Solve for** = True

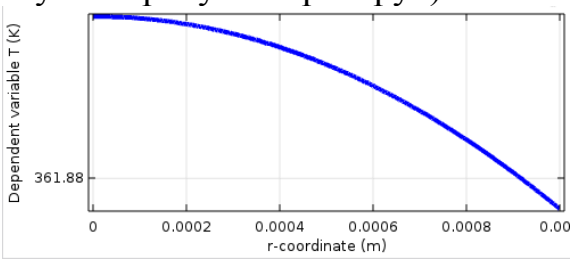


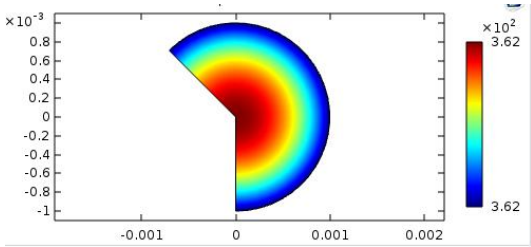
Останнє вказує що на другому кроці розв'язуватися буде фізика **General Form PDE (g)**. При чому шукана величини *T* першого кроку буде автоматично відома на другому. Для цього непотрібно додавати ніяких налаштувань.

Виконайте обчислення

Home **-> Study** - Compute

і отримайте чотири графічні групи. В групі **1D Plot Group 1** побудований графік температури *T* (наступний рисунок ліворуч), а в групі **2D Plot Group 2** зображений поверхневий графік тієї ж температури по перерізу дроту (наступний рисунок праворуч).





Оскільки дріт дуже тонкий, то діапазон температур дуже вузький. Надрукуйте мінімальне і максимальне значення.

Results – Derived Values: RMB > Maximum – Line Maximum

Line Maximum 1:

Selection – < > = 1 (*відрізок* [0,*R*])

Expression – **Expression** = T

LMB > Evaluate

Results – Derived Values: RMB > Minimum – Line Minimum

Line Minimum 1:

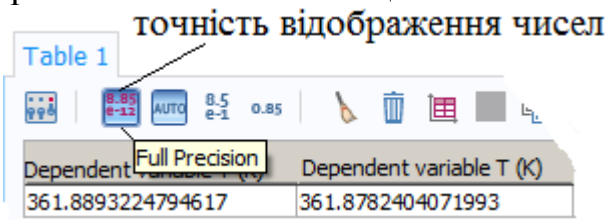
Selection – < > = 1 (*відрізок* [0,*R*])

Expression – **Expression** = T

LMB > Evaluate –Table 1

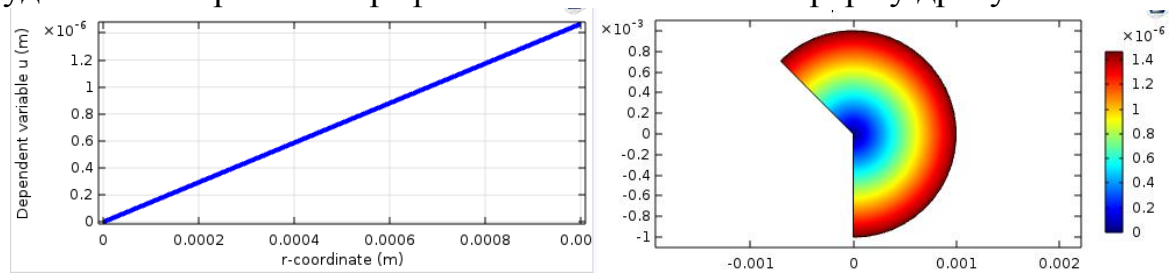
Dependent variable T (K)	Dependent variable T (K)
361.89	361.88

Це той випадок, коли корисно застосувати кнопку Full Precision, щоб побачити більшу точність відображення чисел в таблиці.



Як бачите, мінімум відрізняється від максимуму приблизно на 0.01 градуса.

У вузлі **1D Plot Group 3** побудований графік переміщення *u*. Він майже прямолінійний (наступний рисунок ліворуч). А у вузлі **2D Plot Group 4** побудовано поверхневий графік тієї ж величини по перерізу дроту.



Радіальна температурна напруга обчислюється за формулами (3), (4). Щоб побудувати її графік створимо змінну моделі, яка буде обчислюватися за цими формулами.

Component 1 (comp1) – Definitions: RMB > Variables

Definitions – Variables 1:

`Variables` – **Name** = `Sr`

– **Expression** = `(ur+nu*u/r-(1+nu)*alph*(T-T0))*E/(1-nu^2)`

Щоб змінна стала доступною, оновіть обчислення моделі.

Study 1: RMB > Update Solution

Тепер можна будувати графік величини σ_r.

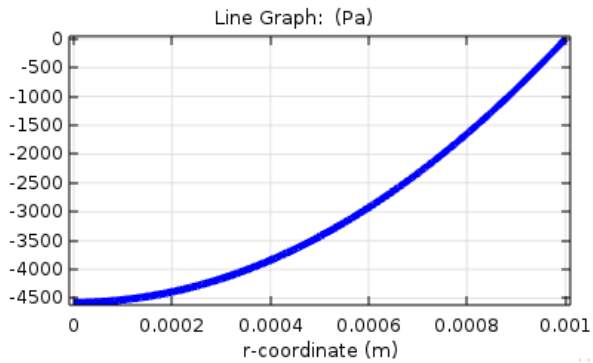
Results - 1D Plot Group 1: RMB > Duplicate

1D Plot Group 5 - Line Graph 1 :

`y-Axis Data` – **Expression** = `Sr`

LMB > Plot

Графік радіальної термонапруги σ_r набуде вигляду.



У рівняння (2) існує аналітичний розв'язок [6]

$$U=(1+\nu)\alpha\frac{1}{r}\int_0^r (T(\rho)-T_0)\rho\, d\rho + C_1 r, \text{ де } C_1=(1-\nu)\frac{\alpha}{R^2}\int_0^R (T(\rho)-T_0)\rho\, d\rho \qquad (8)$$

з якого по формулам (3), (4) обчислюється радіальна температурна напруга

$$\sigma_r=\alpha E\left(\frac{1}{R^2}\int_0^R (T(\rho)-T_0)\rho\, d\rho-\frac{1}{r^2}\int_0^r (T(\rho)-T_0)\rho\, d\rho\right) \qquad (9)$$

Для перевірки побудуємо графіки величин U і σ_r.

Спочатку створіть оператор інтегрування по відрізку [0,*R*].

`Definitions` **`-> Coupling - Component Coupling`** – Integration

Component 1 - Definitions - Integration 1 (intop1):

Source Selection – **Geometric entity level** = Domain
– **Selection, < >** = Manual,1 (*відрізок* [0,*R*])

Ім'я оператора intop1 в полі **Operator name** залиште без змін.

Тепер в таблиці **Variables 1** створіть змінні *C*1, *U* і *SUr*, які обчислюватимуться за формулами. (8) і (9).

Definitions – Variables 1:

Variables – **Name** = C1
– **Expression** = (1-nu)*alph/R^2*intop1((T-T0)*r)

Variables – **Name** = U
–**Expression**=(1+nu)*alph*
intop1((T-T0)*r*(r<=dest(r)))/r+C1*r

Variables – **Name** = SUr
– **Expression** = alph*E*(1/R^2*intop1((T-T0)*r)-
intop1((T-T0)*r*(r<=dest(r)))/r^2)

Variables

Name	Expression	Unit
Sr	(ur+nu*u/r-(1+nu)*alph*(T-T0))*E/(1-nu^2)	Pa
C1	(1-nu)*alph/R^2*intop1((T-T0)*r)	
U	(1+nu)*alph*intop1((T-T0)*r*(r<=dest(r)))/r+C1*r	m
SUr	alph*E*(1/R^2*intop1((T-T0)*r)-intop1((T-T0)*r*(r<=dest(r)))/r^2)	Pa

Знову оновіть обчислення моделі.

Study 1: RMB > Update Solution

Додайте графічні вузли в групи **1D Plot Group 3** і **1D Plot Group 5**, побудувавши в них відповідні графіки.

Results - 1D Plot Group 3 - Line Graph 1: RMB > Duplicate

1D Plot Group 3 - Line Graph 2 :

y-Axis Data – **Expression** = U

Coloring and style – **Color, Width** = Yellow, 2

LMB > Plot (*наступний рисунок ліворуч*)

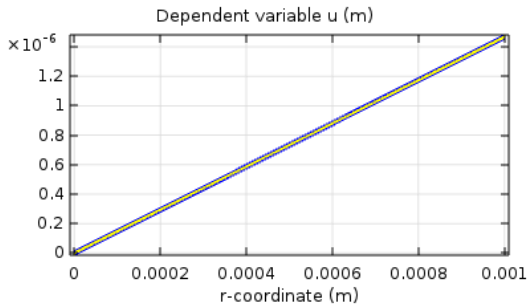
Results - 1D Plot Group 5 - Line Graph 1: RMB > Duplicate

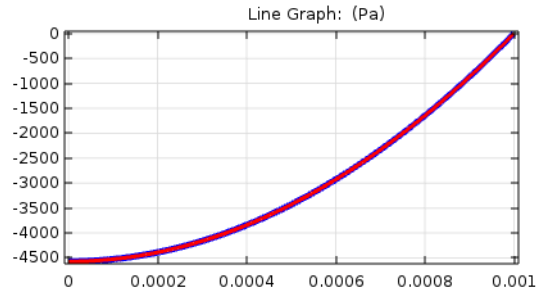
1D Plot Group 5 - Line Graph 2 :

y-Axis Data – **Expression** = SUr

Coloring and style – **Color, Width** = Red, 2

LMB > Plot (*наступний рисунок праворуч*)





Зауваження. Насправді в формулах (8), (9) при $r = 0$ немає особливості. Наприклад, для граничного значення другого члена в (9) маємо

$$\lim_{r\to 0}\frac{1}{r^2}\int_0^r (T(\rho)-T_0)\rho\, d\,\rho=\frac{1}{2}(T(0)-T_0).$$

Тому радіальна напруга σ_r при $r=0$ матиме скінченне значення

$$\sigma_r(0)=\alpha E\left(\frac{1}{R^2}\int_0^R (T(\rho)-T_0)\rho\, d\,\rho-\frac{1}{2}(T(0)-T_0)\right).$$

Можете самостійно обчислити сталу $\sigma_r(0)$ по останній формулі. Однак обчислити змінну `Sr` в точці $r=0$ не вдасться (буде ділення на нуль). Але у вузлі `Global Evaluation` можна обчислити значення в близькій до нуля точці, наприклад, `at1(0.000001,Sr)`. Зробіть це самостійно і порівняйте отримані значення. Вірогідно, для цього вам доведеться створити ще дві змінні моделі для значень $T(0)$ і $\sigma_r(0)$.

Література.

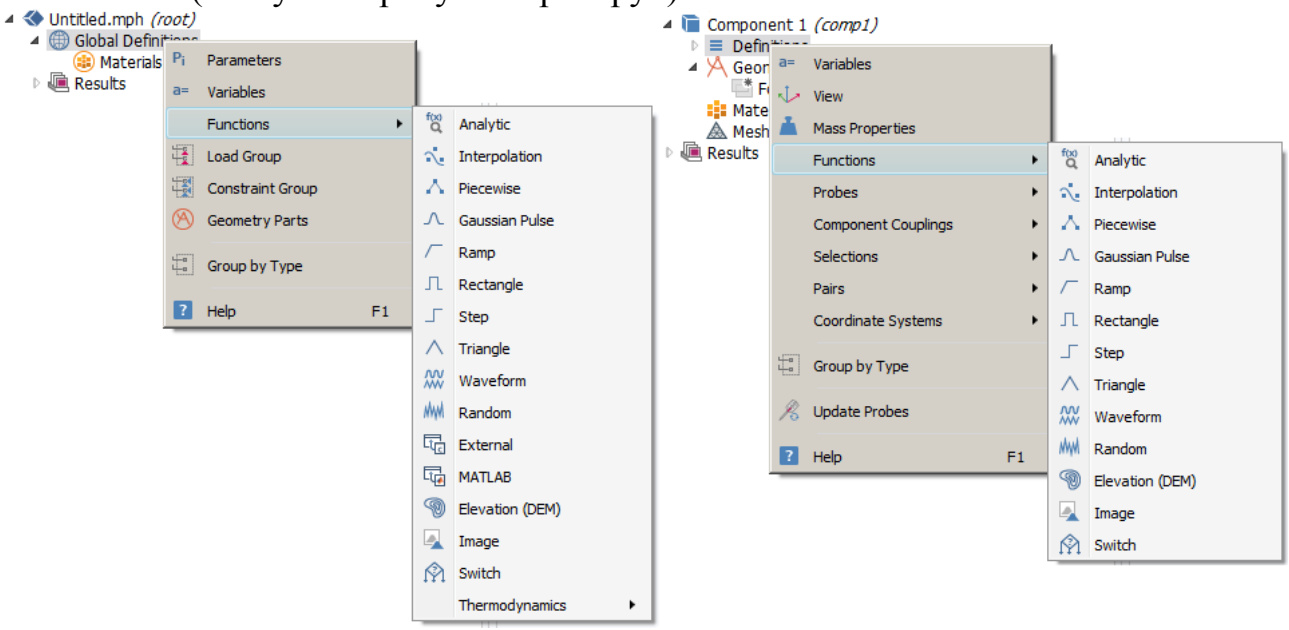
1. Трикоми Ф. Дифференциальные уравнения. Изд. Иностранной лит. М. – 1962
2. Будак Б.М., Самарский А.А., Тихонов А.Н. Сборник задач по математической физике: Учебное пособие. – 3-е изд. – М.: Наука, Главная редакция физико-математической литературы, 1980, 688 стр.
3. Работнов Ю.Н. Механика деформируемого твердого тела. – Учеб. пособие для вузов. – 2-е изд. – М.: Наука. Гл. ред. физ.-мат. лит., 1988, – 712 с.
4. Тихонов А.Н., Самарский А.А. Уравнения математической физики. – М.: Наука, 1977. – 736с.
5. Лаврентьев М.А., Люстерник Л.А. Курс вариационного исчисления. – М.–Л.: Гос. изд. технико-теоретической литературы, 1950. – 296 с.
6. Тимошенко С.П., Гудьер Дж. Теория упругости. – М.: Наука, 1979.
7. Тимошенко С.П. Сопротивление материалов. Т.1 – М.: Наука, Главная редакция физико-математической литературы, 1965, 363 с.

6. Функції.

При створенні математичної моделі COMSOL Multiphysics доводиться застосовувати велику кількість інструментів. Значна частина з них розділена на групи по типам завдань і зосереджена на основних інструментальних стрічках: геометрія (Geometry), матеріали (Materials), фізики (Physics), сітки (Mesh), дослідження (Study), тощо. Окрім цього існують засоби, які використовуються при записі математичних виразів. Ними є функції, оператори і змінні. Вони можуть бути вбудованими, визначеними, або обчисленими (якщо мова йде про змінні). Звичайно, що спочатку засоби користувача повинні бути створені. Інструменти для цього зібрано на стрічці «визначення» (Definitions). В кількох наступних главах мова піде про них, а в поточній – про функції. Вони вже створювалися багато разів, але в більшості випадків це були функції обчислювані по аналітичним формулам. Існують і інші «типи» функцій. Оскільки побудова моделей в COMSOL Multiphysics обходиться без мови програмування, то для створення різнотипних функцій використовуються спеціальні «трафарети».

Функції користувача в COMSOL Multiphysics різняться по рівню доступу і бувають глобальними (створеними у вузлі Global Definitions) і локальними (створеними в компоненті). Різниця між ними полягає лише в способі виклику: «локальні» функції в інших компонентах визиваються по повному імені (ім'я компонента, крапка, ім'я функції), наприклад, `comp1.f(x)`.

Список всіх трафаретів функцій можна побачити, розкривши випадне меню вузлів **Global Definitions** (наступний рисунок ліворуч) або **Component 1 – Definitions** (наступний рисунок праворуч).



Шаблони функцій майже однакові. Тому далі ми говоритимемо про ці функції, ігноруючи їх рівень доступу (глобальний чи локальний).

6.1. Аналітичні функції.

Тут під аналітичними функціями ми розумітимемо вбудовані функції, а також функції обчислювані за формулами.

Перш ніж переходити до обговорення функцій користувача, розглянемо деякі особливості використання вбудованих функцій, таких як $\sin(x)$- синус, $\tan(x)$ - тангенс та інші. Ці функції потребують застосування безрозмірних аргументів і повертають безрозмірний результат. Тому, якщо ви збираєтеся застосувати функцію sin(x), яка приймає аргумент x довжини, вимірюваний в метрах, і повертає зусилля, вимірюване в ньютонах, то в полі виразу ви повинні написати `1[N]*sin(x[1/m])`, або `sin(x[1/m])[N]`.

Ми не будемо давати повний перелік вбудованих функцій, користувач знайде його в довідковій системі. Наведемо лише деякі найчастіше використовувані:

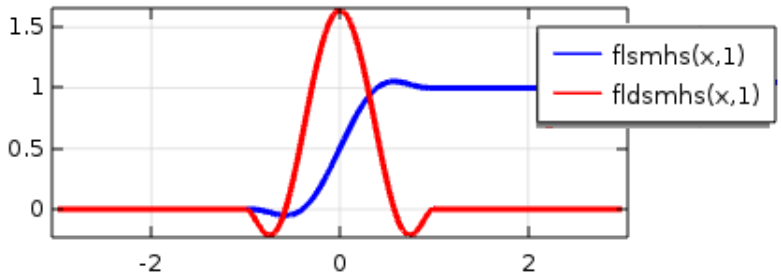
- тригонометричні функції `sin(x)`, `tan(x)`, ... ;
- `exp(x)`- e^x;
- `abs(x)`- абсолютне значення числа (дійсного або комплексного);
- `ceil(x)`- найближче наступне ціле;
- `floor(x)`- найближче попереднє ціле;
- `round(x)`- округлення до найближчого цілого;
- `mod(a,b)`- залишок від ділення (перший аргумент - ділене, другий аргумент – дільник);
- `max(a,b)`, `min(a,b)` – максимум і мінімум з двох аргументів;
- `random(x,y)`- випадкова функція з рівномірним розподіленням навколо 0 і діапазоном 1;
- `randomnormal(x,y)`- випадкова функція з нормальним (гаусовим) розподіленням, з середнім значенням 0 і середньоквадратичним відхиленням 1;
- `gamma(x)`- гамма-функція; може застосовуватися для обчислення факторіалів цілих чисел `n`: `gamma(n+1)=n!`;
- `erf(x)`- функція помилок.

Імена всіх вбудованих функцій (`abs`,`cos`,`tan`, ...) зарезервовані і спроба використати їх для інших цілей може приводити до непередбачених наслідків.

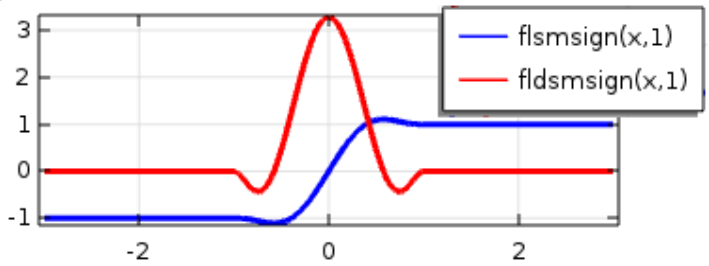
Більшість стандартних функцій не потребують спеціального пояснення, але в системі COMSOL Multiphysics вони можуть мати нетипові для «аналітичної» математики застосування. Деякі приклади будуть наведені далі.

Згладжувальні функції. Коли коефіцієнти рівнянь, наприклад, властивості матеріалу, містять розриви неперервності, то можуть виникати проблеми збіжності. Щоб їх уникнути, розриви замінюються виразами з використанням вбудованих гладких функцій, апроксимуючих функцію Гевісайда (функцію-сходинку) і функцію знака (sign-функцію) схожими гладкими функціями. Ми умовно називатимемо їх згладжувальними.

1. `flsmhs(x,s)` – згладжена функція-сходинка (згладжена Heaviside-функція) з неперервною першою похідною і викидами в перехідній зоні $-s \le x \le s$; `fldsmhs(x,s)` – похідна згладженої функції-сходинки з викидами.

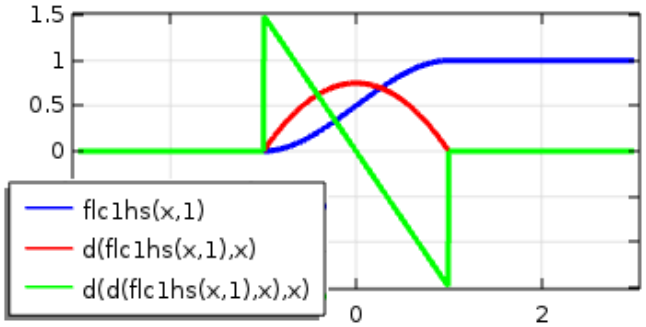


2. `flsmsign(x,s)` – згладжена функція знаку з неперервною першою похідною і викидами в перехідній зоні $-s \le x \le s$; `fldsmsign` – похідна згладженої sign-функції з викидами.



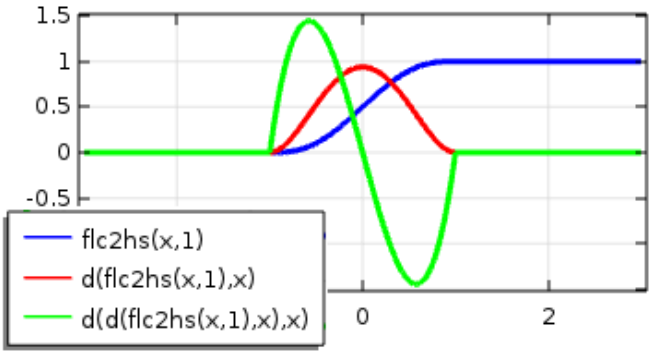
3. `flc1hs(x,s)`– згладжена функція-сходинка з неперервною першою похідною (без викидів) і перехідною зоною $-s \le x \le s$; для неї в довідковій системі наводиться наступний вираз

$$\mathrm{flc1hs}(x,s) = \begin{cases} 0 & , x \le -s \\ 0.5 + 0.75 \cdot x/s - 0.25 \cdot x^2/s^2, & -s < x < s \\ 1 & , x \ge s \end{cases}.$$

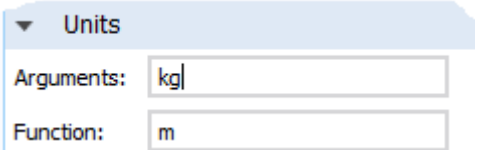


4. `flc2hs(x,s)` – згладжена функція-сходинка з неперервною першою і другою похідними (без викидів) і перехідною зоною $-s \le x \le s$; для неї в довідковій системі наводиться наступний вираз

$$\mathrm{flc2hs}(x,s) = \begin{cases} 0 & , x \le -s \\ 0.5 + 0.9375 \cdot x/s - 0.625 \cdot x^3/s^3 + 0.1875 \cdot x^5/s^5, & -s < x < s \\ 1 & , x \ge s \end{cases}.$$



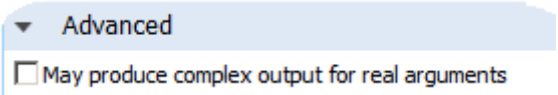
Аналітичні (Analytic) функції користувача. Окрім вбудованих функцій користувач може створювати власні функції, які обчислюються за формулами. При цьому можна визначати фізичні одиниці вимірювання аргументів і значення функції. Вони вказуються в ПВ функції в розділі **`Units`**.



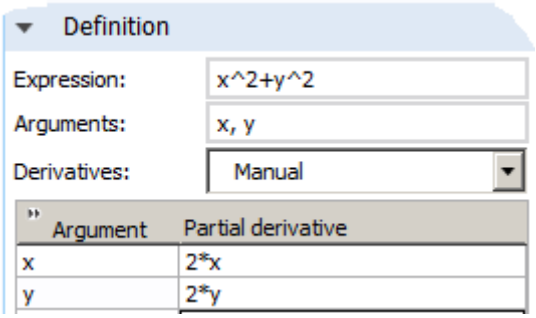
Якщо функція має кілька аргументів, то їм всім призначаються однакові одиниці вимірювання.

Розділ **`Units`**, окрім аналітичних мають шаблони функцій `Interpolation` і `Piecewise`.

Деяким «аналітичним» функціям корисно дозволити повертати комплексні значення навіть при дійсних значеннях аргументів. Це робиться встановлюванням відповідного перемикача в розділі **`Advanced`**.

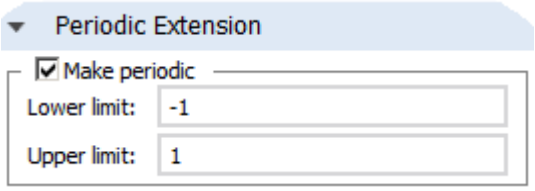


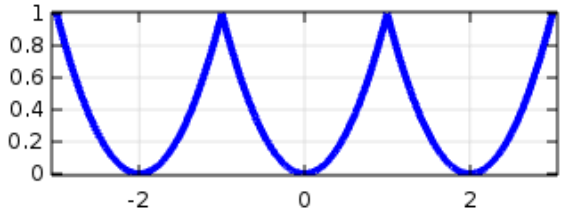
Інколи корисно підказати системі формули для обчислення похідних аналітичної функції. Це потрібно, коли в якості аргументу створюваної функції будуть застосовуватися шукані величини. Коли у випадному списку **`Definition`** – **Derivatives** встановлено значення `Automatic`, то похідні обчислюються системою символьно. Встановлення значення `Manual` потребує ручного задання похідних по всім змінним, інакше вони будуть покладені нулями.



Поле **Derivatives** також буває корисним для External і MATLAB функцій.

Важливе налаштування для аналітичних функцій виконується в розділі **`Periodic Extension`**. Його сенс випливає з назви. Наприклад, налаштування в цьому розділі, які показані на наступному рисунку ліворуч, для аналітичного виразу x^2 призводять до створення функції з графіком, зображеним праворуч.





Існує можливість створювати аналітичні функції більш ніж трьох аргументів, але побудувати графік такої функції не вдасться. Можна залишити вільними не більше трьох аргументів, а іншим задати сталі значення. Тоді графік побудується.

6.2. Кускові функції і приклади їх застосування.

Користувач може створювати власні функції різних типів. Для них існують різні шаблони, які обираються з випадних меню Functions вузлів **Definitions** або на однойменній інструментальній панелі. В наступних прикладах ми опишемо способи створення і особливості застосування деяких кускових функцій.

Приклад 1. Приклади використання шаблонів `Step`, `Rectangle`, `Piecewise` і `Ramp` функцій при побудові графіків поверхонь.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Вимкніть в моделі перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Оскільки в цьому прикладі ми збираємося будувати графіки глобальних виразів, то для джерел даних нам буде достатньо вузлів типу `Grid` і вузол дослідження `Study` не знадобиться.

1. Почнемо розглядання функції-сходинки (step-функції). Вона виконує різкий перехід від одного значення (From) до іншого (To) в якомусь місці або в якийсь момент часу (Location).

Global Definitions: RMB > Functions – Step

Function name = `step1`

`Parameters`– Location, From, To = `0,0,1`

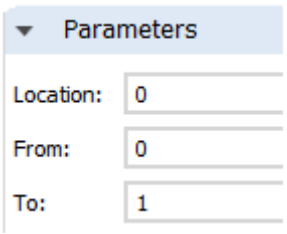
Зніміть перемикач, який примушує функцію бути гладкою.

`Smoothing` – Size of transition zone = `False`

Для перевірки виконайте побудову графіка цієї функції.

LMB> Plot (*наступний рисунок праворуч*)

Сенс параметрів незгладженої функції-сходинки пояснюється на наступному рисунку.



В подальших прикладах ми часто застосовуватиме цю незгладжену функцію при значеннях опцій **Location**=x_0, **From**=0, **To**=1, і позначатимемо її $\mathrm{step}(x-x_0)$.

Функція-сходинка створюється як безрозмірна функція одного аргументу (безрозмірні аргумент і значення). Якщо в якості аргументу використовувати функції двох змінних, то можна будувати поверхні. Наступні інструкції пояснюють сказане.

Створіть двовимірне джерело даних `Grid 2D`.

`Results` -> Data Set - More Data Sets -> Grid 2D

Results - Data Sets - Grid 2D 1 :

`Data` – **Source, Function** = `Function, All`

Вибір в списку **Source** значення `Function` означає, що на множині точок `Grid 2D` буде обчислюватися значення функції. У списку **`Function`** обирається потрібна функція (або `All`, якщо над цим ДД будуватимуться графіки кількох функцій).

`Parameemter Bounds` –

– **First parameter** – **Minimum, Maximum** = `-2,2`

– **Second parameter** – **Minimum, Maximum**= `-2,2`

В розділі **`Resolution`** вказуються кількості точок дискретизації прямокутної області по кожній координаті; залишимо їх незмінними.

Тепер побудуйте графік функції $\mathrm{step1}\left(1-x^2-y^2\right)$ «над» цим ДД.

`Results` -> **`Plot Group`** -> 2D Plot Group

2D Plot Group 1:

`Data` – **Data set** = `Grid 2D 1`

`2D Plot Group 1`-> **`Add Plot`** -> Surface

2D Plot Group 1 – Surface 1:

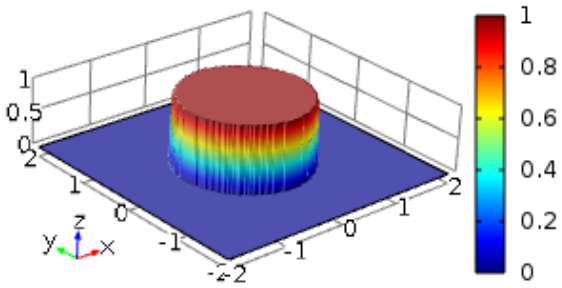
`Expression` – **Expression** = `step1(1-x^2-y^2)`

RMB > Height Expression

2D Plot Group 1 – Surface 1 – Height Expression 1:

`Axis` – **Scale factor** = `True,1`

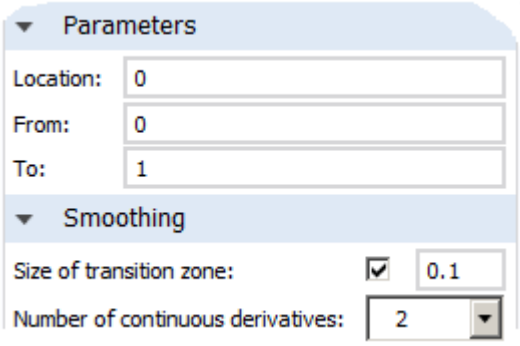
LMB > Plot

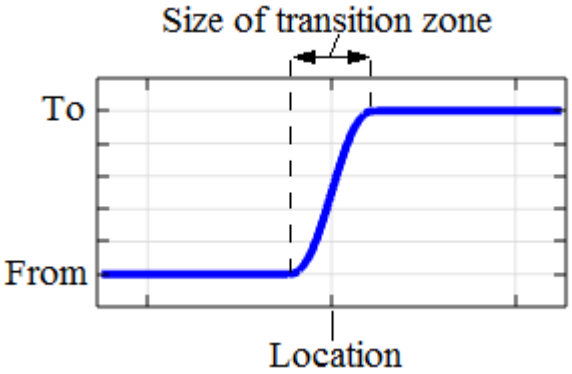


Зауваження. При визначенні незгладженої функції-сходинки (**Size of transition zone** = `False`) її значення в нулі можна задавати по різному. В COMSOL визначено, що $\mathrm{step}(0)=1$.

Зауваження. Незгладжену функцію-сходинку можна задавати логічним виразом $(x>0)$, який в арифметичних обчисленнях інтерпретуватиметься як 0 або 1 в залежності від хибності чи істинності нерівності.

Сенс параметрів згладженої функції-сходинки пояснюється на наступному рисунку. Її корисно застосовувати для узгодження граничних і початкових умов в динамічних задачах.



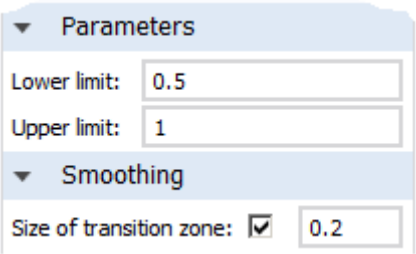


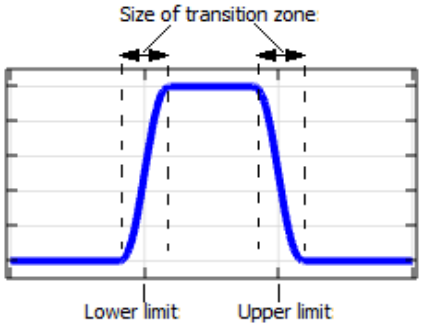
2. Функція-прямокутник (Rectangle) дорівнює 1 в заданому інтервалі і 0 – зовні нього (окрім перехідних зон, яких може не бути). Її можна представити у вигляді різниці двох функцій-сходинок (`Step`) з однаковими значеннями полів **From**=`0`, **To**=`1`, і однаково налаштованою опцією **Smoothing**. Для незгладженої функціїї-прямокутника введемо позначення $\mathrm{rect}(x,a,b)$, і для неї матиме місце співвідношення

$$\mathrm{rect}(x,a,b)=\mathrm{step}(x-a)-\mathrm{step}(x-b)$$

де a,b $(a<b)$ є значеннями опцій **Location** функцій-сходинок і опцій **Lower limit** і **Upper limit** функції `Rectangle`.

Сенс параметрів функції-прямокутника пояснюється наступним рисунком.





Продемонструємо застосування функції-прямокутника, продовживши роботу з останнім файлом. Створіть в поточному файлі функцію-прямокутник.

Global Definitions: RMB > Functions – Rectangle

Function name = `rect1`

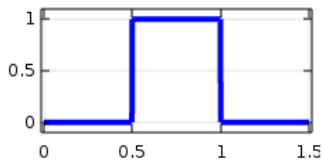
`Parameters`– Lower limit, Upper limit = `0.5, 1`

Зніміть перемикач, який змушує функцію бути гладкою.

`Smoothing` – **Size of transition zone** = `False`

Для перевірки виконайте побудову графіка функції.

LMB> Plot



Ми маємо намір створити розривну функцію двох змінних з графіком у формі «квадратної стінки». Для цього нам знадобиться функція, яка додатна всередині квадрата $-1 \le x \le 1, -1 \le y \le 1$, дорівнює нулю на його межі і від'ємна зовні. Такі функції в п. 2.2.2 ми називали ідентифікаторами області. Там же було наведено кілька прикладів їх побудови. Для зазначеного квадрата ця функція може мати наступний вид

$$W(x,y) = 2 - x^2 - y^2 - \sqrt{\left(1-x^2\right)^2 + \left(1-y^2\right)^2}\,.$$

Створіть її.

Global Definitions: RMB > Functions – Analytic

Function name = `W`

`Definition` – **Expression** = `(2-x^2-y^2-sqrt((1-x^2)^2+(1-y^2)^2))/(2-sqrt(2))`

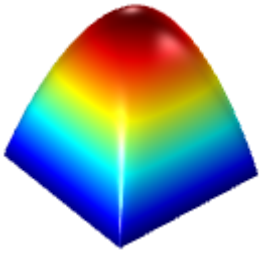
– **Arguments** = `x,y`

Ділення на `(2-sqrt(2))` ми додали для того, щоб функція на початку координат дорівнювала одиниці. Для перевірки побудуйте графік функції.

`Plot Parameters` – **Argument, Low limit, Upper limit** = `x,-1,1`

– **Argument, Low limit, Upper limit** = `y,-1,1`

LMB> Plot



Поза зоною квадрата $-1 \le x \le 1, -1 \le y \le 1$ функція $W(x,y)$ від'ємна.

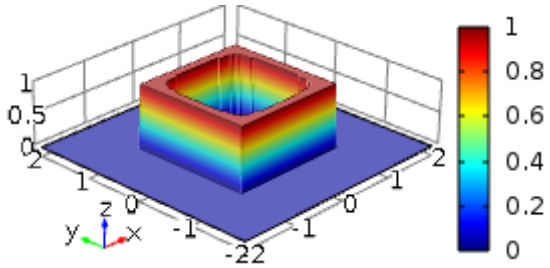
Тепер побудуйте графік функції `rect1(1-W(x,y))`. В тих зонах площини XY, де вираз `1-W(x,y)` менше за `0.5` або більше 1, вона дорівнює нулю.

2D Plot Group 1: RMB > Duplicate

2D Plot Group 2 – Surface 1:

`Expression` – **Expression** = `rect1(1-W(x,y))`

LMB> Plot

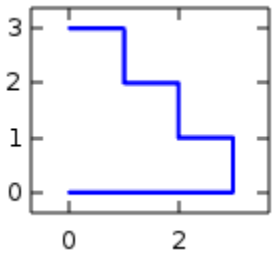


Можна вигадати багато схожих застосувань для функцій `Step` і `Rectangle`.

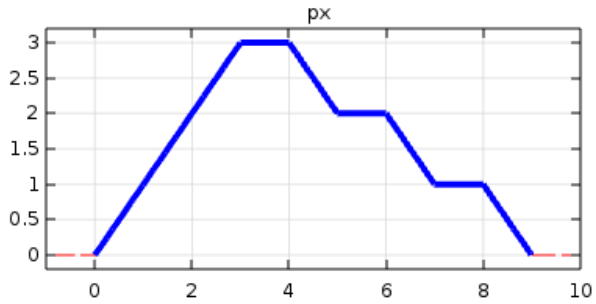
3. Кускові (`Piecewise`) функції в COMSOL створюються у вигляді функцій однієї змінної, і на різних ділянках дійсної осі обчислюються за різними формулами. Підстановка в аргумент функції кількох змінних може перетворити

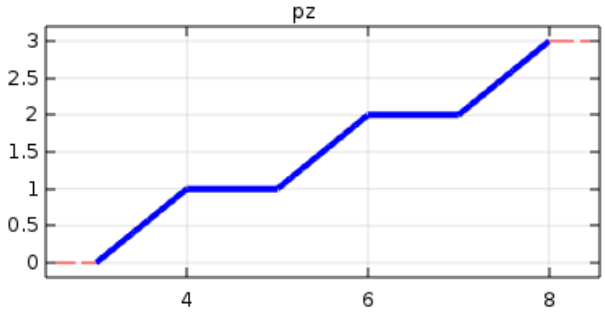
таку суперпозицію на функцію кількох аргументів. Окрім іншого, кускові функції можна використовувати в параметричних рівняннях поверхонь. Для прикладу побудуємо поверхню обертання багатосходинкової ламаної, параметричні рівняння якої створимо за допомогою двох piecewise функцій.

Спочатку створимо параметричні рівняння ламаної, побудованої в площині XZ (див. наступний рисунок).



Якщо параметру кривої (скажімо u) призначити у вузлах ламаної значення 0,3,4,5,6,7,8,9, то графіки координатних функцій $x = \mathrm{px}(u), z = \mathrm{pz}(u)$ можуть мати наступний вид.



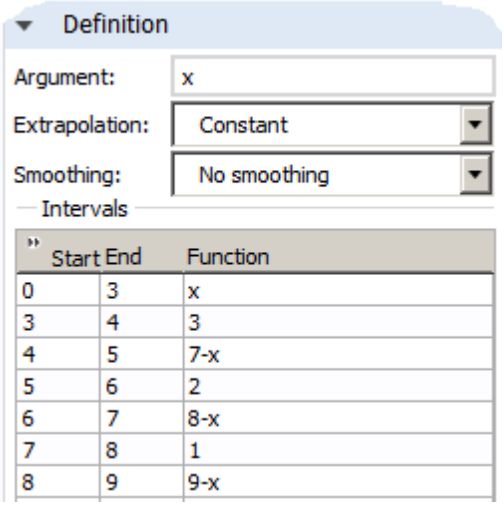


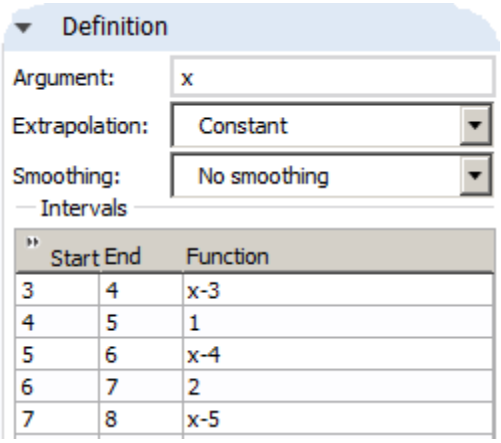
Побудуйте такі кускові функції.

Global Definitions : RMB > Functions – Piecewise

Function name = px

Заповніть розділ **`Definition`** ПВ вузла функції так, як зображено на наступному рисунку ліворуч.





LMB> Plot (*попередній графік ліворуч*)

Global Definitions : RMB > Functions – Piecewise

Function name = pz

Заповніть розділ **`Definition`** ПВ вузла функції так, як зображено на попередньому рисунку праворуч.

LMB> Plot (*попередній графік праворуч*)

Перевірте, що створені функції дозволяють накреслити багатосходинкову ламану. Для побудови ламаної по параметричному рівнянню використайте лінійний графік. Але перед цим для задання діапазону зміни параметра `u` в параметричних рівняннях створіть ДД `Grid 1D`.

`Results` -> Data Set - More Data Sets -> Grid 1D

Data Sets - Grid 1D 2 :

`Data` – **Source, Function** = `Function, All` (*два поля*)

`Parameemter Bounds` – **Name** = `u`

`Parameemter Bounds` – **Minimum, Maximum** = `0,9` (*два поля*)

Створіть групу одновимірних графіків і додайте в неї `Line Graph` вузол.

`Results` -> Plot Group - 1D Plot Group

1D Plot Group 3:

`Data Set` – **Data** = `Grid 1D 2`

RMB > Line Graph

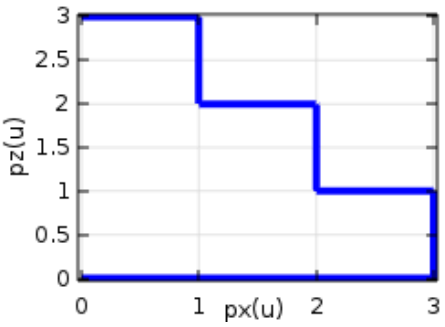
1D Plot Group 3 - Line Graph 1:

`y-Axis Data` – **Expression** =`pz(u)`

`x-Axis Data` – **Parameter, Expression** = `Expression,px(u)`

В розділі **`Coloring and Style`** задайте стиль, колір і товщину лінії.

LMB > Plot



Якщо параметричне рівняння кривої в площині XZ має вид

$$x=\varphi(u),\ z=\psi(u)\ ,\ \ u_0\le u\le u_1,$$

то параметричне рівняння поверхні її обертання навкруги осі Z має вигляд

$$x=\varphi(u)\cos v,\ y=\varphi(u)\sin v,\ z=\psi(u),\ \ u_0\le u\le u_1\ ,0\le v\le 2\pi\ .$$

Використайте ці формули і побудуйте параметричні рівняння та зображення поверхні обертання багатосходинкової ламаної навкруги осі Z.

Для побудови поверхні по параметричним рівнянням потрібно мати `Parameterized Surface` джерело даних. Оскільки точки поверхні будуть розташовані в тривимірному просторі, то додатково знадобиться тривимірне ДД.

`Results` -> Data Set - More Data Sets -> Grid 3D

Results - Data Sets - Grid 3D 3 :

`Data` – **Source, Function** = `Function, All`

`Parameemter Bounds` –

– **First parameter** – **Minimum, Maximum** = `-3,3`

– **Second parameter** – **Minimum, Maximum** = `-3,3`

– **Third parameter** – **Minimum, Maximum** = `0,3`

В розділі **Resolution** вказується кількість точок дискретизації просторової області; в нашому прикладі в полях **x resolution**, **y resolution** і **z resolution** залиште значення за замовчуванням 30.

Тепер побудуйте Parameterized Surface джерело даних.

Results -> Data Set - More Data Sets -> Parameterized Surface

Data Sets - Parameterized Surface 1 :

Data – **Data set** = Grid 3D 3

Parameter – **First parameter** – **Name** = u

– **First parameter** – **Minimum, Maximum** = 0,9

– **Second parameter** – **Name** = v

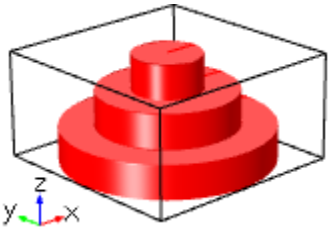
– **Second parameter** – **Minimum, Maximum** = 0,2*pi

Expression – **x** = px(u)*cos(v)

– **y** = px(u)*sin(v)

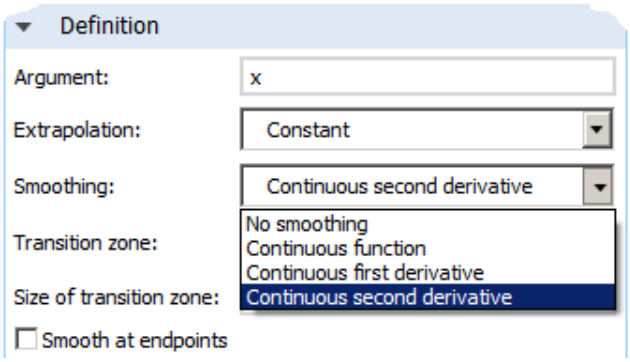
– **z** = pz(u)

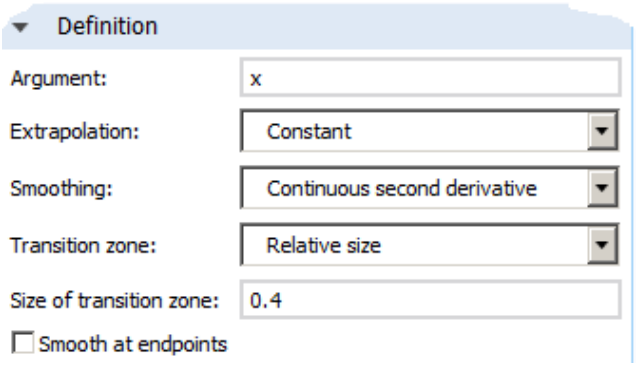
LMB > Plot



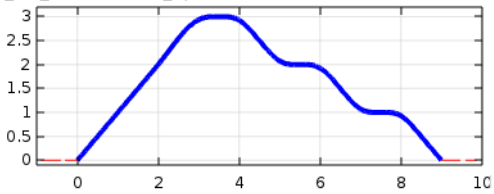
Розфарбовувати поверхню цього ДД в тривимірній графічній групі ми не станемо.

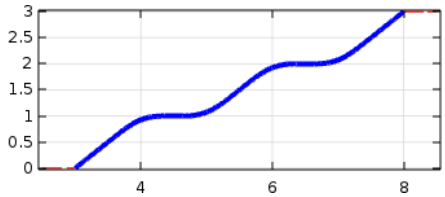
Зауваження. У кускових функцій є опція згладжування. Вона обирається з випадного списку **Smoothing** і може робити функцію неперервною, один або два рази неперервно диференційованою.





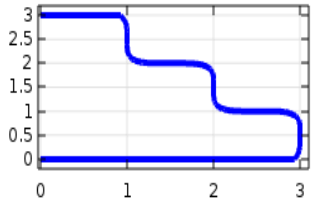
Якщо її застосувати до координатних функцій px і py, то ребра на поверхні будуть заокругленими. Поверніться до налаштувань цих функцій і задайте опції згладжування (попередній рисунок праворуч). Нижче зображені їх графіки при неперервних других похідних і значенні опції **Size of transition zone** =0.4.

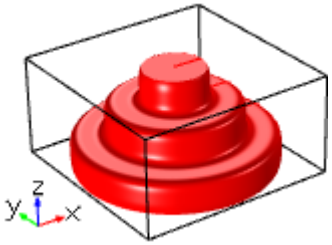




На наступному рисунку ліворуч зображено криву, отриману по параметричним

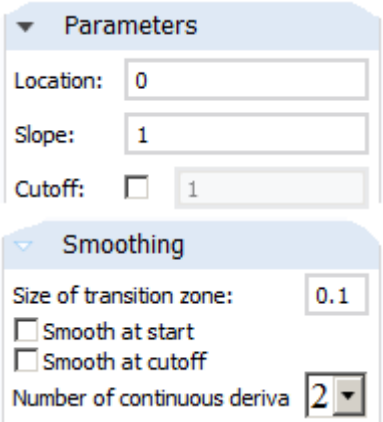
рівнянням $x = px(u)$, $y = py(u)$, а праворуч – поверхню її обертання.



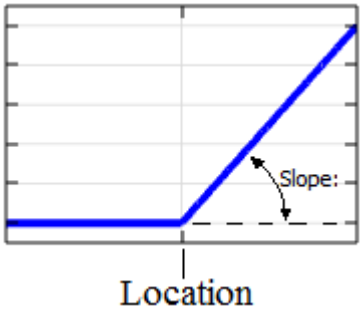


Виконайте ці побудови самостійно.

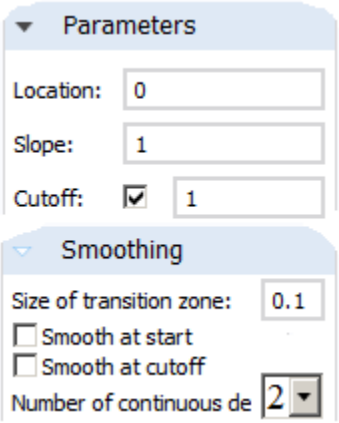
4. Незгладжена функція-нахил (Ramp-функція) може мати два види, сенс параметрів і графіки яких наведені на наступних рисунках.



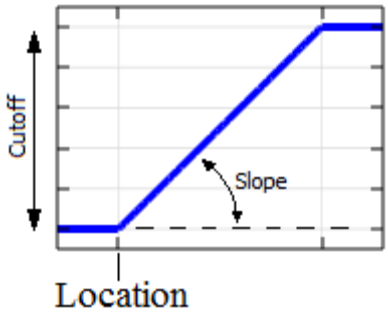
«Одноповерхова» Ramp-функція



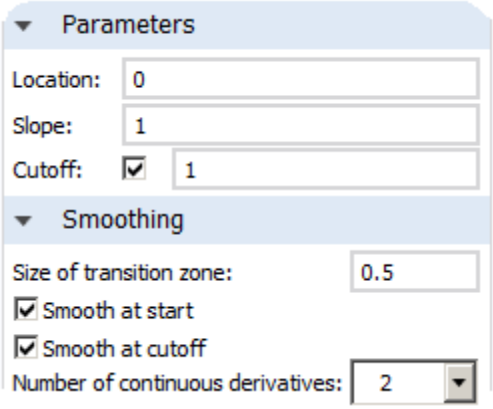
Ми позначатимемо незгладжену «одноповерхову» Ramp-функцію з одиничним нахилом як $\mathrm{ramp}(x - x_0)$, де x_0 значення опції **Location**. Для неї має місце співвідношення $\mathrm{ramp}(x - x_0) = (x - x_0) \cdot \mathrm{step}(x - x_0)$.

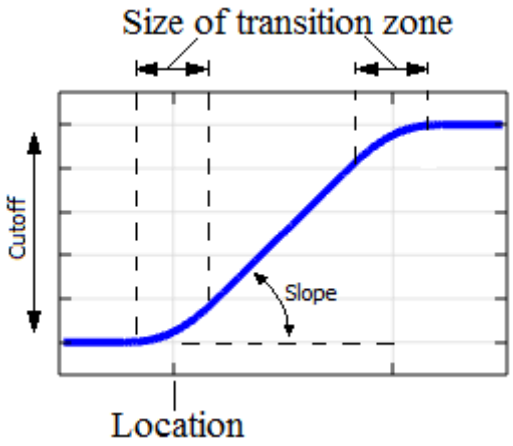


«Двоповерхова» Ramp-функція



Сенс параметрів згладженої «двоповерхової» Ramp-функції пояснюється на наступному рисунку.





Використання цієї функції досить різноманітно і тут ми наведемо лише два приклади.

Функцію `Ramp` можна застосовувати для «обрізання» діапазонів зміни аргументів і значень функцій. Почнемо з простого прикладу побудови поверхні півкулі.

Поверхня півкулі має рівняння $z=\sqrt{1-x^2-y^2}$. Якщо ДД прямокутне, наприклад, $-2\le x\le 2$, $-2\le y\le 2$, то під квадратним коренем можуть бути від'ємні значення. Але застосовуючи незгладжену Ramp-функцію $\mathrm{ramp}(x)$, рівняння поверхні можна записати у вигляді $z=\sqrt{\mathrm{ramp}\left(1-x^2-y^2\right)}$, і під квадратним коренем від'ємні значення не з'являтимуться.

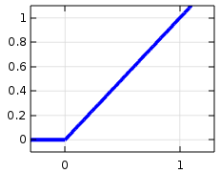
Створіть «одноповерхову» Ramp-функцію з одиничним нахилом.

Global Definitions: RMB > Functions – Ramp

Function name = `rm1`

`Parameters`– Location, Slope, Cutoff = `0,1,False`

Перемикачі згладжування не встановлюйте.



Для перевірки виконайте побудову графіка.

LMB> Plot

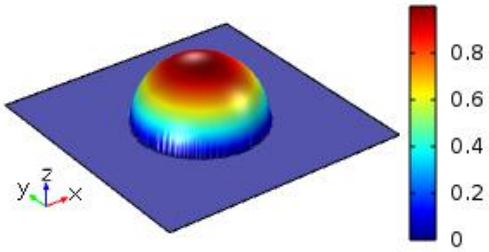
Тепер побудуйте графік поверхні.

2D Plot Group 1: RMB > Duplicate

2D Plot Group 4 – Surface 1:

`Expression` – Expression = `sqrt(rm1(1-x^2-y^2))`

LMB> Plot



Застосування «двоповерхової» Ramp-функції до однієї з координатних функцій параметричної поверхні дозволяє обрізати її відразу двома площинами, перпендикулярними до напрямку координати.

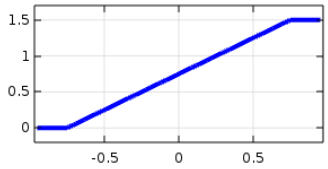
Створіть наступну «двоповерхову» Ramp-функцію.

Global Definitions: RMB > Functions – Ramp

Function name = rm2

Parameters – **Location, Slope** = -0.75,1

– **Cutoff, <>** = True, 1.5



Для перевірки побудуйте графік функції.

LMB> Plot

Для подальшого нам знадобиться ще одне Grid 3D джерело даних в формі куба $[-1,1]\times[-1,1]\times[-1,1]$.

Results -> Data Set - More Data Sets -> Grid 3D

Results - Data Sets - Grid 3D 4 :

Data – **Source, Function** = Function, All

Paramemter Bounds –

– **First parameter** – **Minimum, Maximum** = -1,1

– **Second parameter** – **Minimum, Maximum** = -1,1

– **Third parameter** – **Minimum, Maximum** = -1,1

Тепер створіть і налаштуйте джерело даних Parameterized Surface.

Results -> Data Set - More Data Sets -> Parameterized Surface

Data Sets - Parameterized Surface 2 :

Data – **Data set** = Grid 3D 4

Parameter – **First parameter** – **Name** = u

– **First parameter** – **Minimum, Maximum** = 0, 2*pi

– **Second parameter** – **Name** = v

– **Second parameter** – **Minimum, Maximum**=-pi/2,pi/2

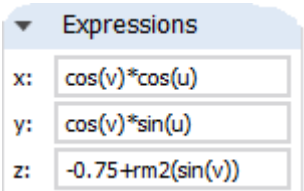
Expression – **x** = cos(v)*cos(u)

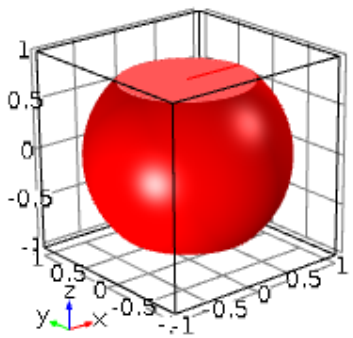
– **y** = cos(v)*sin(u)

– **z** = -0.75+rm2(sin(v))

Тут в параметричному рівнянні кулі виконується обрізання значень координатної функції $z(u,v)$

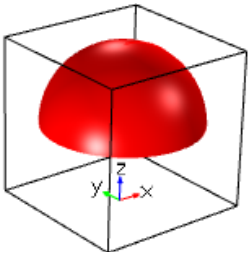
LMB > Plot





Вправа. Побудувати зображення півкулі по її параметричному рівнянню.

Вказівка. В параметричному рівнянні кулі до z координати застосувати «одноповерхову» Ramp-функцію.



Запропоновані перетворення можна виконувати при створенні розрахункової геометрії. Для прикладу побудуйте по параметричному рівнянню «обрізану кулю» по граням куба трохи меншого за кулю. Звичайно, що це можна зробити за допомогою геометричних інструментів COMSOL, але мати параметричне рівняння поверхні такого тіла теж буває корисним.

Щоб мати розрахункову геометрію нам знадобиться додати в модель тривимірний компонент.

Home –> Model –> Add Component – 3D

Розрахункова геометрія компонента буде тілом з заданою параметричною поверхнею.

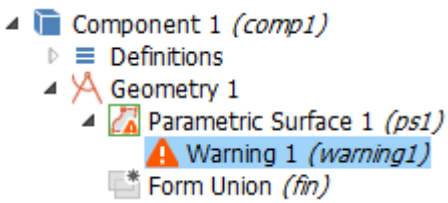
Geometry –> **Primitives – More Primitives** – Parametric Surface

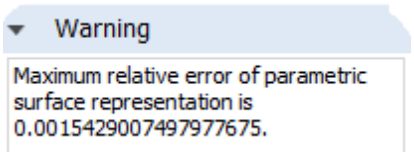
Geometry 1 – Parametric Surface 1 (ps1):

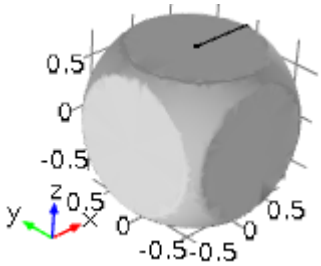
Parameter – **First parameter** – **Name** = u
– **First parameter** – **Minimum, Maximum** = 0,2*pi
Parameter – **Second parameter** – **Name** = v
– **Second parameter** – **Minimum, Maximum**=-pi/2,pi/2
Expression – **x** = -0.75+rm2(cos(v)*cos(u))
– **y** = -0.75+rm2(cos(v)*sin(u))
– **z** = -0.75+rm2(sin(v))
Advanced Settings – **Relative tolerance** = 1E-4
– **Maximum number of knots** = 100

Зверніть увагу на те, що ми зменшили точність «внутрішнього» зображення поверхні (опція **Relative tolerance**=1E-4) і збільшили максимальну кількість вузлів в параметричних рівняннях B-сплайнів, які використовуються системою при зображенні поверхні. Зазвичай кількість вузлових точок в сплайні підбирається так, щоб відносна похибка не перевищувала значення в полі **Relative tolerance**, або доки не досягнуто кількості вузлів, зазначеної в полі **Maximum number of knots**. Для достатньо складних поверхонь, щоб система змогла коректно побудувати поверхню, потрібно керувати значеннями цих опцій.

LMB > Build All Objects







Як бачите, система побудувала розрахункову геометрію, але попередила що при заданій опції **Maximum number of knots**=100 бажана точність не досягнута.

■

В математичній фізиці добре відомий метод періодичного продовження, але його застосування часто наштовхується на труднощі побудови періодичних функцій. Та в COMSOL Multiphysics у аналітичних і Piecewise функцій користувача є опція періодичного продовження. Застосовуючи цю можливість, в деяких задачах вдається уникати складних обчислень. Для прикладу розглянемо одновимірне хвильове рівняння

$$u''_{tt} = a^2 u''_{xx} + f(x,t), \tag{1}$$

де $u(x,t)$ являє зміщення з положення рівноваги точки x одновимірного коливного середовища в момент часу t. Для однозначного визначення розв'язку рівняння (1) необхідно задати початкові умови

$$u(x,0) = \varphi(x),\ u'_t(x,t) = \psi(x) \tag{2}$$

і граничні умови, якщо у коливного об'єкта є межа.

Для необмеженої прямої $-\infty < x < \infty$ $(t \ge 0)$ відомий загальний розв'язок задачі (1), (2) в формі Д'Аламбера [1]

$$u(x,t) = \frac{\varphi(x+at)+\varphi(x-at)}{2} + \frac{1}{2a}\int_{x-at}^{x+at}\psi(\alpha)d\alpha + \frac{1}{2a}\int_0^t \int_{x-a(t-\tau)}^{x+a(t-\tau)} f(\xi,\tau)d\xi\ d\tau\ , \tag{3}$$

при чому формула (3) при будь-яких (достатньо гладких) функціях $\varphi(x), \psi(x)$ дає розв'язок рівняння (1).

Метод періодичного продовження при розв'язанні задачі (1), (2) для скінченного відрізка $0 \le x \le L$ при нульових граничних умовах

$$u(0,t) = 0,\ u(L,t) = 0 \tag{4}$$

полягає в тому, що виконується непарне періодичне продовження функцій $\varphi(x), \psi(x)$ і $f(x,t)$ на всю вісь (з відрізка $x \in [0, L]$ непарно на відрізок $[-L,0]$ і далі на всю вісь з періодом $2L$). Відома лема [1], яка стверджує, що коли функції $\varphi(x), \psi(x), f(x,t)$ є непарними функціями відносно деякої точки x_0, то формула Д'Аламбера (3) дає розв'язок рівняння (1), який в цій точці дорівнює нулю. Тоді підстановка непарно періодично продовжених функцій $\tilde{\varphi}(x), \tilde{\psi}(x)$, $\tilde{f}(x,t)$ в (3) дасть функцію $u(x,t)$, яка є розв'язком рівняння коливань (1), задовольняє на відрізку $[0, L]$ початковим умовам (2), і обертається в нуль на кінцях цього відрізка. Це означає, що функція $u(x,t)$ на відрізку $[0, L]$ буде розв'язком крайової задачі (1), (2), (4). Отже розв'язок задачі можна отримати, якщо функції $\varphi(x), \psi(x), f(x,t)$ продовжити непарно і періодично на всю вісь. І в системі COMSOL Multiphysics для цього існують потрібні інструменти.

Зауваження 1. Парне продовження функції $f(x)$ з додатної півосі $x > 0$ на всю вісь $-\infty < x < \infty$ виконується за формулою $F(x) = f(|x|)$, а непарне – за формулою $F(x) = \operatorname{sign}(x) \cdot f(|x|)$, де $\operatorname{sign}(x)$ функція знака.

Приклад 2. Застосування функцій користувача `Triangle`, `Rectangle`, `Ramp`, `Waveform` при побудові розв'язків одновимірного хвильового рівняння.

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `L=1, a=1 i T=2*L/a` (`T` період коливань).

Створіть в моделі одновимірний компонент.

Untitled.mph : RMB > Add Component - 1D

Його геометрія буде складатися з відрізка [0,*L*].

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1:

`Interval` – **Left endpoint** = 0

– **Right endpoint** = `L`

LMB > Build All Objects

На створеному відрізку генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Додайте в модель вузол динамічного дослідження.

`Home` -> Study - Add Study = `Add Study` ->

Preset Studies - Time Dependent -> **`x`**

Задайте діапазон і інтервал зміни часу `t`.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times** = `range(0,0.1,T-0.1)`

1. Скінченна струна довжини L закріплена на кінцях. Початковий профіль струни $u(x,0)=\varphi(x)=2\sqrt{x}(L-x)$. Початкова швидкість 0 ($u_t'(x,0)=\psi(x)=0$), зовнішня сила відсутня $f(x,t)=0$.

При обраних даних формула (3) набуває спрощеного вигляду

$$u(x,t)=\frac{\tilde{\varphi}(x+at)+\tilde{\varphi}(x-at)}{2}, \qquad (3_1)$$

де $\tilde{\varphi}(x)$ є непарним періодичним продовженням функції $\varphi(x)$ з відрізка [0, *L*] на всю дійсну вісь. Відповідно до зауваження 1, потрібно створити аналітичну функцію $2\,\mathrm{sign}(x)\cdot\sqrt{|x|}\,(L-|x|)$ і додати їй опцію періодичного продовження з відрізка [-*L*, *L*].

Global Definitions : RMB > Functions – Analytic

Function name = `F1`

`Definition` – **Expression** = `2* sign(x)*sqrt(abs(x))*(L-abs(x))`

–**Arguments** = `x`

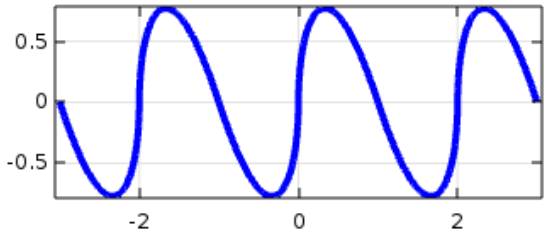
`Periodic Extension` – **Make periodic** = `True`

– **Lower limit, Upper limit** = `-L, L`

Для перевірки виконайте побудову графіка функції.

Plot Parameters – **Argument, Low limit, Upper limit**=`x,-3*L,3*L`

LMB> Plot



За формулою ($З_1$) побудуйте функцію розв'язку.

Global Definitions : RMB > Functions – Analytic

Function name = `u1`

Definition – **Expression,Arguments** = `(F1(x+a*t)+F1(x-a*t))/2,x,t`

Виконайте компіляцію моделі.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**. Його «робочий простір» містить інформацію про довжину відрізка, діапазон змінної `t`, та функцію `u1`.

Тепер можна будувати графік розв'язку.

`Results` –> Plot Group - 1D Plot Group

1D Plot Group 1:

Data – **Data set** = `Study 1/Solution 1`

– **Time selection** = `range(1,1,11)` (*перші* 11 *моментів часу*)

`1D Plot Group 1`–> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

Selection – **Selection, <>** = `Manual,1` (*відрізок*)

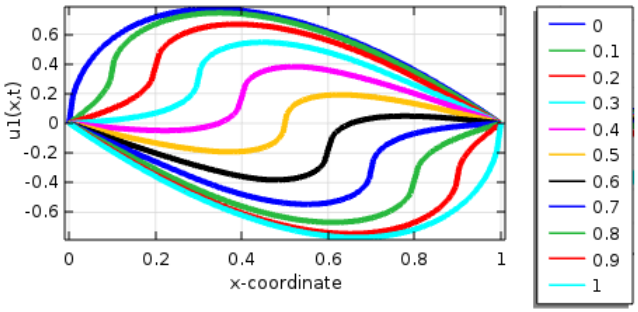
y-Axis Data – **Expression** = `u1(x,t)`

x-Axis Data – **Parameter, Expression** = `Expression,x`

Coloring and Style – **Width** = `3`

Legends – **Show legends** = `True`

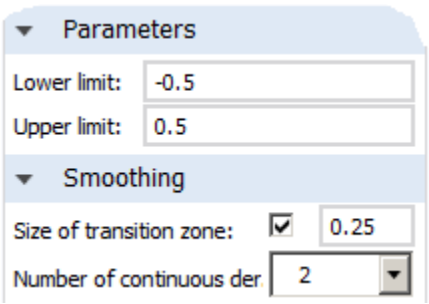
LMB > Plot

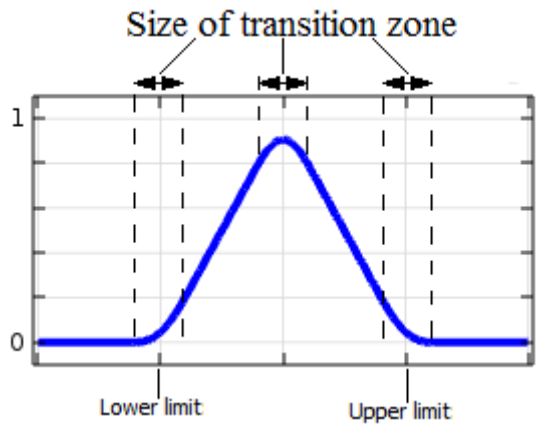


Анімацію коливань побудуйте самостійно.

2. Скінченна струна довжини L закріплена на кінцях. Початкове зміщення 0 $(u(x,0)=\varphi(x)=0)$. Початкова швидкість $u'_x(x,0)=\psi(x)$ в формі трикутного піку, зовнішня сила відсутня $f(x,t)=0$.

Для задання початкової швидкості використаємо `Triangle`-функцію. Сенс параметрів і її графік наведено на наступному рисунку.





Створіть незгладжену `Triangle`-функцію.

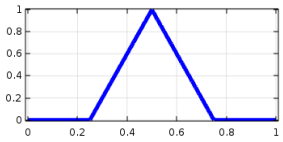
Global Definitions: RMB > Functions – Triangle

Function name = `tri1`

`Parameters`– Lower limit, Upper limit = `0.25,0.75`

Зніміть прапорець, який вмикає згладжування.

`Smoothing` – **Size of transition zone** = `False`



Для перевірки побудуйте графік функції.

LMB> Plot

Тепер створіть функцію, яка є непарним продовженням функції $\psi(x)$. Це можна виконати за формулою `tri1(x)-tri1(-x)`. Потім додайте опції періодичного продовження.

Global Definitions : RMB > Functions – Analytic

Function name = `F2`

`Definition` – **Expression** = `tri1(x)-tri1(-x)`

– **Arguments** = `x`

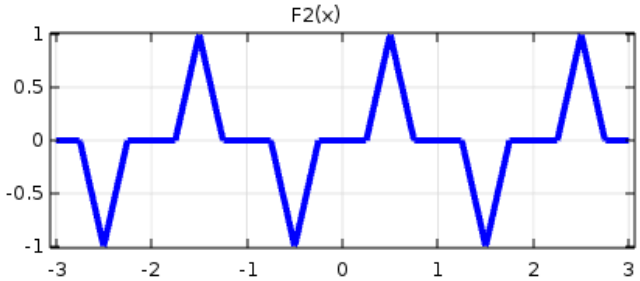
`Periodic Extension` – **Make periodic** = `True`

– **Lower limit, Upper limit** = `-L, L`

Для перевірки побудуйте графік функції.

`Plot Parameters` – **Argument, Low limit, Upper limit**=`x,-3*L,3*L`

LMB> Plot



Оскільки $\varphi(x)=0$ і $f(x,t)=0$, то формула (3) набуває спрощеного вигляду

$$u(x,t)=\frac{1}{2a}\int_{x-at}^{x+at}\tilde{\psi}(\alpha)d\alpha, \qquad (3_2)$$

де в якості $\tilde{\psi}(x)$ потрібно використати функцію `F2(x)`. Але в COMSOL Multiphysics оператори (зокрема оператор інтегрування) не можна застосовувати в тілах функцій. Тому замість функції $u(x,t)$ ми створимо змінну моделі, яка обчислюватиметься за останньою формулою.

Definitions -> **Variables** – Local Variables

Definitions - Variables 1:

Variables – **Name** = u2

–**Expression** = integrate(F2(ksi),ksi,x-a*t,x+a*t)/(2*a)

Поновіть обчислення і побудуйте графік змінної u2.

Study 1: RMB > Update Solution

Спершу побудуйте графіки струни в початкові моменти часу.

Results -> Plot Group - 1D Plot Group

1D Plot Group 2:

Data – **Data set** = Study 1/Solution 1

– **Time selection** = Interpolated

Опція Interpolated дозволяє обирати для графіків проміжні моменти часу, які не зазначені в діапазоні range(0,0.1,T-0.1) вузла **Study 1 - Step 1: Time Dependent**.

– **Times** = range(0,0.05,0.5)

1D Plot Group 1-> Add Plot – Line Graph

1D Plot Group 2 - Line Graph 1:

Selection – **Selection,** <> = Manual,1 (*відрізок*)

y-Axis Data – **Expression** = u2

x-Axis Data – **Parameter, Expression** = Expression,x

Coloring and Style – **Width** = 3

Legends – **Show legends** = True

LMB > Plot (*наступний рисунок ліворуч*)

Тепер побудуйте графік в подальші моменти часу.

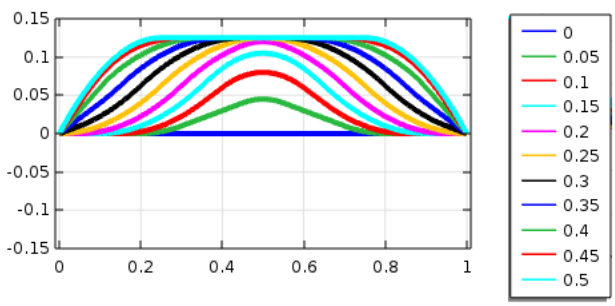
1D Plot Group 2: RMB > Duplicate

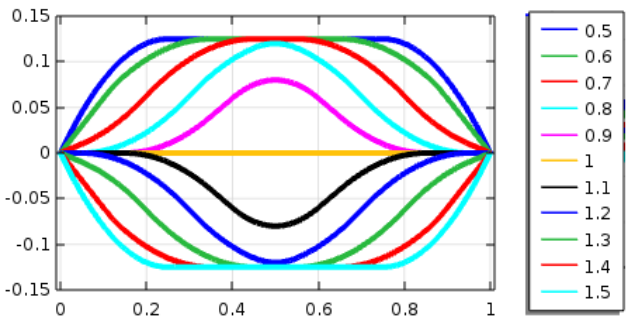
1D Plot Group 3:

Data – **Time selection** = Manual

– **Times** = range(6,1,16)

LMB > Plot (*наступний рисунок праворуч*)





Як бачите, відхилення струни в момент часу 0.5 максимальне, а потім струна починає повертатися до точки рівноваги і далі в зворотну сторону.

Анімацію коливань побудуйте самостійно.

3. Скінченна струна довжини L закріплена на кінцях. Початкове зміщення 0 $\left(u(x,0)=\varphi(x)=0\right)$, початкова швидкість 0 $\left(u'_x(x,0)=\psi(x)=0\right)$. Зовнішня сила

$f(x,t)=f(x)$ не залежить від часу і є `Rectangle`-функцією, зсунутою вбік відносно середини струни.

Розв'язок (3) в цьому випадку набуває вигляду

$$u(x,t)=\frac{1}{2a}\int_0^t \int_{x-a(t-\tau)}^{x+a(t-\tau)} \tilde{f}(\xi)d\xi \; d\tau, \qquad (3_3)$$

де в якості $\tilde{f}(x)$ потрібно використати непарне періодичне продовження функції $f(x)$.

Створіть незгладжену `Rectangle`-функції.

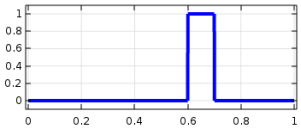
Global Definitions: RMB > Functions – Rectangle

Function name = `rect1`

`Parameters`– Lower limit, Upper limit = `0.6,7`

Зніміть прапорець, який вмикає згладжування.

`Smoothing` – Size of transition zone = `False`



Для перевірки побудуйте графік функції.

LMB> Plot

Тепер створіть функцію, яка є непарним продовженням попередньої. Це можна зробити за формулою `rect1(x)- rect1(-x)`. Ми ще використаємо числовий множник для збільшення амплітуди. Потім додайте опції періодичного продовження.

Global Definitions : RMB > Functions – Analytic

Function name = `F3`

`Definition` – Expression = `3*(rect1(x)- rect1(-x))`

– Arguments = `x`

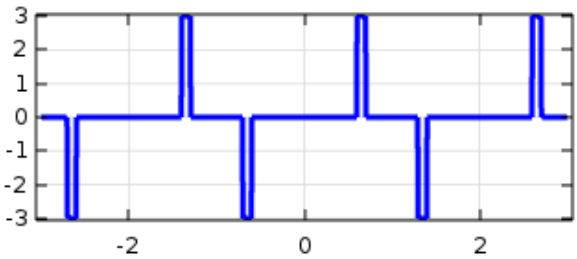
`Periodic Extension` – Make periodic = `True`

– Lower limit, Upper limit = `-L, L`

Для перевірки побудуйте графік функції.

`Plot Parameters` – Argument, Low limit, Upper limit=`x,-3*L,3*L`

LMB> Plot



Для побудови розв'язку (3_3) потрібно застосувати інтегрування (подвійне), яке можна використовувати лише при обчисленні змінних. Отже створіть відповідну змінну.

Definitions - Variables 1:

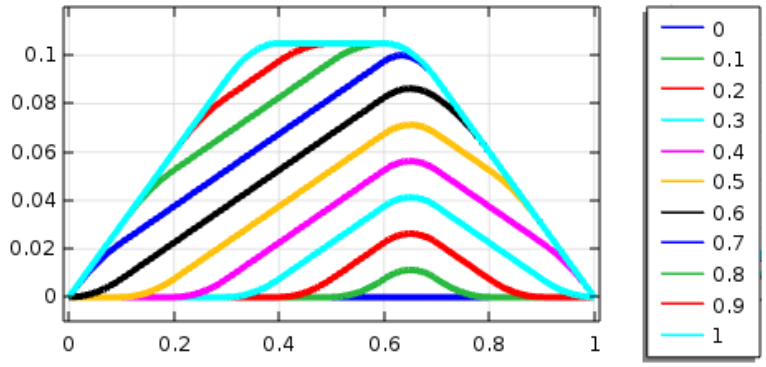
`Variables` – Name = `u3`

–Expression = `integrate(integrate(F3(ksi),ksi,`
`x-a*(t-tau), x+a*(t-tau)),tau,0,t)/(2*a)`

Поновіть обчислення і побудуйте графік змінної `u3`.

Study 1: RMB > Update Solution

1D Plot Group 3: RMB > Duplicate
1D Plot Group 4:
Data – **Time selection** = Manual
– **Times** = range(1,1,11) (*перші* 11 *моментів часу*)
1D Plot Group 4 - Line Graph 1:
y-Axis Data – **Expression** = u3
LMB > Plot



Майте на увазі, що подвійне інтегрування триває довше одинарного, тому побудова графіка забере деякий час.

Зауваження. Викладений тут підхід легко переноситься на скінченну струну з двома вільними кінцями, або одним закріпленим, а іншим вільним (без зовнішнього зусилля). В першому випадку в формулі Д'Аламбера потрібно виконати парне періодичне продовження початкових функцій $\varphi(x), \psi(x)$ з відрізка $[0, L]$ на дійсну вісь з періодом $2L$. В другому – потрібно початкові функції з відрізка $[0, L]$ продовжити парно на відрізок $[L,2L]$ (за умови, що точка $x=L$ струни вільна), а потім отриману функцію з відрізка $[0,2L]$ продовжити непарно на відрізок $[-2L,0]$ з подальшим періодичним продовження з періодом $4L$.

4. Лівий край напівнескінченної струни $(x \geq 0)$ в нульовий момент часу починає рухатися по закону $u(0,t) = \mu(t)$ $(\mu(0) = 0)$. Початкові умови дорівнюють нулю $u(x,0) = 0$, $u'_t(x,0) = 0$, зовнішня сила відсутня $(f(x,t) = 0)$. Дослідити коливання струни.

Граничний режим викликає хвилю, що поширюється вправо уздовж струни зі швидкістю a. Це означає, що розв'язок може бути представлений у вигляді $u(x,t) = f(x - at)$, де вид функції f визначиться з граничної умови при $x = 0$. Оскільки $u(0,t) = f(-at) = \mu(t)$, то $f(z) = \mu(-z/a)$. Тоді матимемо $u(x,t) = \mu(-\frac{x-at}{a}) = \mu\left(t - \frac{x}{a}\right)$. Але функція $\mu(z)$ визначена лише для значень $z \geq 0$. Оскільки розв'язок має задовольняти початковій умові, то $0 = u(x,0) = \mu(-x/a)$. Це означає, що на від'ємні значення аргументу z функцію $\mu(z)$ потрібно продовжити нулем, поклавши $\mu(z) = 0$ $(z < 0)$.

Функція, яка дорівнює нулю при $z < 0$ і дорівнює $\mu(z)$ при $z > 0$ може бути представлена у вигляді $\tilde{\mu}(z) = \mu(\mathrm{ramp}(z))$ (це можливо, оскільки вважаємо

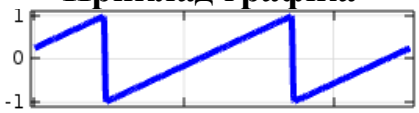
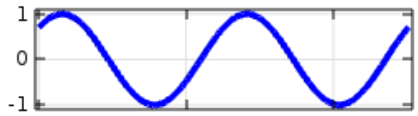
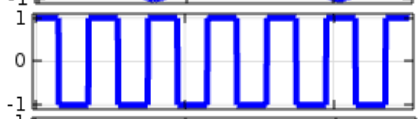
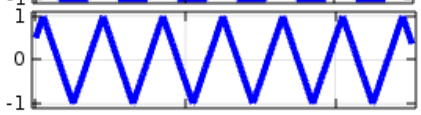
$\mu(0)=0$). Таким чином, маючи функцію граничного зміщення $\mu(t)$, задану для $t \geq 0$, можна побудувати розв'язок задачі коливань напівнескінченної струни (викликане граничним режимом) у вигляді

$$u(x,t)=\mu\left(\operatorname{ramp}\left(t-\frac{x}{a}\right)\right). \tag{5}$$

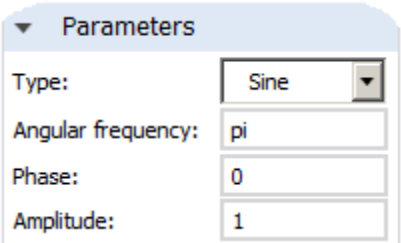
Нагадаємо, що через $\operatorname{ramp}(x-x_0)$ ми позначили незгладжену «одноповерхову» функцію-нахил (Ramp-функцію), де x_0 є значенням її опції **Location**.

Побудуйте графік останнього розв'язку, коли функція $\mu(x)$ є непарною періодичною трикутноподібною функцією. Для цього використайте waveform-функцію.

В COMSOL Multiphysics Waveform-функція (функція форми хвилі) – це періодична функція одного аргументу (наприклад, часу `t`) і однією з наступних форм.

Форма	**Приклад графіка**
Пилкоподібна (sawtooth)	
Синусоподібна (sine)	
Прямокутноподібна (square)	
Трикутноподібна (triangle)	

Всі ці функції мають вигляд $A\cdot f(\omega t+\varphi)$, де f визначається вибором однієї з чотирьох зазначених форм, а «кутова частота» ω (angular frequency), фаза φ (phase, вимірюється в радіанах) і амплітуда A задаються в налаштуваннях функції.



Частота коливань цих функцій (в герцах) обчислюється за формулою $\nu=\omega/2\pi$, а період $1/\nu=2\pi/\omega$.

Отже побудуйте спочатку функцію $\mu(x)$.

Global Definitions: RMB > Functions – Waveform

Function name = `mu`

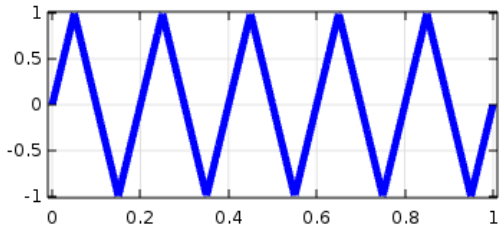
`Parameters`– Type = `Triangle`

`Parameters`– Angular frequency = `10*pi` (*частота* $\nu=\omega/2\pi=5$)

Перемикач згладжування не встановлюйте.

`Parameters`– Smoothing = `False`

LMB > Plot



Тепер побудуйте «одноповерхову» Ramp-функцію з одиничним нахилом.

Global Definitions: RMB > Functions – Ramp

Function name = `rm1`

Parameters– **Location, Slope, Cutoff** = `0,1,False`

Перемикач згладжування не встановлюйте.

Створіть функцію-суперпозицію двох попередніх.

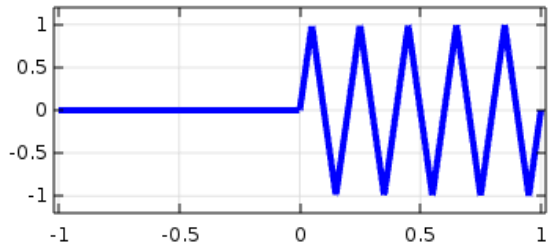
Global Definitions : RMB > Functions – Analytic

Function name = `Mu`

Definition – **Expression, Arguments** = `mu(rm1(x))`

Plot Parameters – **Argument, Low limit, Upper limit**=`x,-L, L`

LMB> Plot



Оновите обчислення і відповідно до (5), побудуйте графік функції $u(x,t)$, наприклад, в момент часу `t=0.7` (зображатимемо графік на одиничному відрізку, маючи на увазі, що він представляє лише ділянку напівнескінченної струни).

Study 1: RMB > Update Solution

1D Plot Group 3: RMB > Duplicate

1D Plot Group 5:

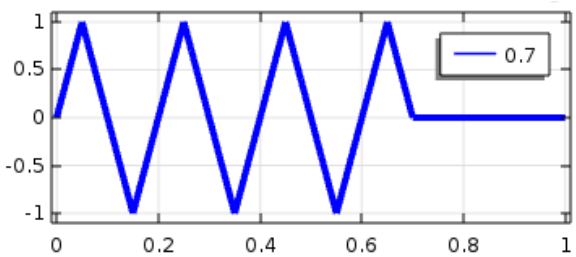
Data – **Time selection** = `From list`

– **Times** = `0.7`

1D Plot Group 5- Line Graph 1:

y-Axis Data – **Expression** = `Mu(t-x/a)`

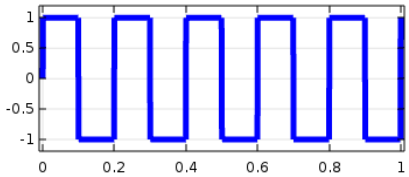
LMB > Plot



Зауваження. Розглянуті в цьому прикладі початкові функції не є гладкими. Тому отримані з них за формулою (3) розв'язки не будуть двічі неперервно диференційованими, тобто в окремих точках вони не задовольнятимуть

хвильовому рівнянню. Та додавання до початкових функцій $\varphi(x), \psi(x)$ в околі їх точок зламу вузьких зон згладжування майже не змінить розв'язки, але зробить їх достатньо гладкими, щоб задовольняти рівнянню (1). Побудовані вище негладкі в деяких точках розв'язки називають узагальненими.

Вправа. Побудуйте графік розв'язку задачі коливань напівнескінченної струни, якщо функція $\mu(x)$ є непарною періодичною прямокутноподібною функцією.



Вказівка. Якщо функцію $\mu(x)$ побудувати, використовуючи прямокутноподібну Waveform-функцію, то вона не буде дорівнювати нулю в нульовий момент часу $(\mu(0) \neq 0)$. Тоді функція $\tilde{\mu}(z)$ потребуватиме коригування, наприклад, наступного: $\tilde{\mu}(z) = \mu(\mathrm{ramp}(z)) - (1 - \mathrm{step}(x)) \cdot \mu(0)$. Тут застосовано від'ємник, який дорівнює $\mu(0)$ для від'ємних значень аргументу x і нулю – для додатних. Без нього не виконуватиметься умова $\tilde{\mu}(z) = 0$ при $z < 0$.

■

Схожі, але трохи складніші обчислення виконуються при розв'язанні одновимірного рівняння теплопровідності.

Задача про поширення тепла в тонкому однорідному необмеженому стрижні, бокова поверхня якого теплоізольована (тепло може поширюватися тільки уздовж осі *x*), формулюється так: знайти обмежену функцію $u(x,t)$ $(t > 0, -\infty < x < \infty)$, яка задовольняє рівнянню теплопровідності

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} \quad (t > 0, -\infty < x < \infty) \tag{6}$$

і початковій умові

$$u(x,0) = \varphi(x) \quad (-\infty < x < \infty). \tag{7}$$

де $u(x,t)$ температура перерізу *x* стрижня в момент часу *t* (стрижень тонкий, це означає, що температура точок поперечного перерізу однакова).

Загальний розв'язок цієї задачі представляється інтегралом Пуассона

$$u(x,t) = \frac{1}{2a\sqrt{\pi t}} \int_{-\infty}^{\infty} \varphi(\xi) e^{-\frac{(\xi - x)^2}{4a^2 t}} d\xi. \tag{8}$$

Якщо в інтегралі зробити заміну $\tau = \dfrac{\xi - x}{2a\sqrt{t}}$, то отримаємо

$$u(x,t) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \varphi\left(x + 2a\sqrt{t}\,\tau\right) e^{-\tau^2} d\tau. \tag{9}$$

В попередньому прикладі ми обчислювали інтеграли по скінченному проміжку. Тут границі необмежені і безпосереднє інтегрування за допомогою оператора `integrate(...)` не спрацьовує (можна обирати скінченні, але достатньо широкі межі інтегрування). Та якщо виконати заміну змінних, щоб інтеграл отримав скінченні границі, то ця перешкода зникає. Однією з найпростіших

може бути заміна $\tau = \dfrac{\xi}{1-|\xi|}$. Вважаючи, що $(|\xi|)'_{\xi} = \operatorname{sign}\xi$ і $\xi \cdot \operatorname{sign}\xi = |\xi|$, матимемо

$$d\tau = \frac{(1-|\xi|) - \xi\cdot(-\operatorname{sign}\xi)}{(1-|\xi|)^2} d\xi = \frac{1}{(1-|\xi|)^2} d\xi .$$

Тоді (9) набуває вигляду

$$u(x,t) = \frac{1}{\sqrt{\pi}} \int_{-1}^{1} \varphi\left(x + 2a\sqrt{t}\,\frac{\xi}{1-|\xi|}\right) e^{-\frac{\xi^2}{(1-|\xi|)^2}} \frac{1}{(1-|\xi|)^2} d\xi . \qquad (10)$$

Останній інтеграл вже можна обчислювати за допомогою оператора `integrate`.

Приклад 3. Застосування функцій користувача `Triangle`, `Rectangle`, `Waveform` при побудові розв'язків рівняння теплопровідності.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `L=3, a=1`.

Створіть в моделі одновимірний компонент.

Untitled.mph : RMB > Add Component - 1D

Його геометрія буде складатися з двох відрізків $[-L,0] \cup [0,L]$.

`Geometry` -> **`Primitives`**-> Interval

Geometry 1 - Interval 1:

`Interval` – **Number of intervals** = `Many`

– **Points** = `-L,0,L`

LMB > Build All Objects

Ці відрізки використовуватимуться для креслення над ними графіків, а побудова розв'язків буде виконуватися по формулі (10).

Генеруйте СкЕ сітку.

Mesh 1: `Mesh Settings` –**Element size** = `Finer`

LMB > Build All

Додайте в модель вузол динамічного дослідження.

`Home` -> Study - Add Study = `Add Study` ->

Preset Studies - Time Dependent -> **`x`**

Задайте діапазон і інтервал зміни часу `t`.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times** = `range(0,0.1,1)`

1. Дослідити задачу розповсюдження тепла в нескінченному стрижні. Початкова температура всюди нуль, крім відрізка $[-1,1]$, де вона дорівнює одиниці, тобто $u(x,0)=\varphi(x)=\begin{cases}1, -1\le x\le 1\\ 0, x<-1\vee x>1\end{cases}$.

Створіть $\varphi(x)$ у вигляді незгладженої `Rectangle`-функції.

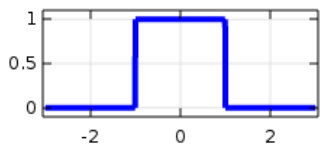
`Home` -> Definitions -> Functions -> Local –Rectangle

Function name = `rect1`

`Parameters`– **Lower limit, Upper limit** = `-1,1`

Зніміть прапорець, який вмикає згладжування.

`Smoothing` – **Size of transition zone** = `False`



Для перевірки побудуйте графік функції.

LMB> Plot

Оскільки в COMSOL Multiphysics в шаблонах функцій не можна використовувати оператор інтегрування, то для температури, яка обчислюється за формулою (10), ми створимо змінну моделі.

`Definitions` -> **`Variables`** – Local Variables

Definitions - Variables 1:

`Variables` – **Name** = `T1`

– **Expression** = `integrate(`
`rect1(x+2*a*sqrt(t)*ksi/(1-abs(ksi)))*`
`exp(-ksi^2/(1-abs(ksi))^2)/((1-abs(ksi))^2+eps),`
`ksi,-1,1)/sqrt(pi)`

Тут в останньому дільнику ми додали машинне епсілон. Воно не впливає на результат, але запобігає діленню на нуль.

Виконайте обчислення і побудуйте графік змінної `T1`.

Study 1: RMB > Compute

`Results` -> Plot Group - 1D Plot Group

1D Plot Group 1:

`Data` – **Data set** = `Study 1/Solution 1`

– **Time selection** = `All`

`1D Plot Group 1`-> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

`Selection` – **Selection, <>** = `Manual,1,2` (*відрізок* [-*L*, *L*])

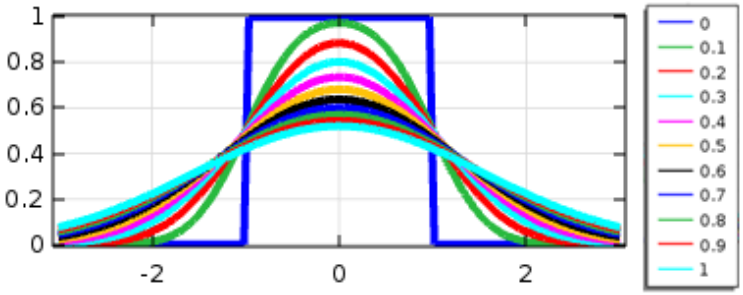
`y-Axis Data` – **Expression** = `T1`

`x-Axis Data` – **Parameter, Expression** = `Expression,x`

`Coloring and Style` –**Width** = `3`

`Legends` – **Show legends** = `True`

LMB > Plot



■

Розглянемо задачу поширення тепла в напівобмеженому стрижні, кінець $x = 0$ якого підтримується при нульовій температурі, тобто $u(0,t)=0 \ (t \geq 0)$. Розв'язання цієї задачі може бути отримано методом продовження, коли початкова температура $\varphi(x)$, яка задана на додатній півосі Ox, продовжується на від'ємну піввісь непарно. Можна показати [1], що формула Пуассона (8) з будь-якою непарною функцією $\varphi(\xi)$ представляє розв'язок задачі (6), (7), який обертається на нуль при $x = 0$. Він, тим самим, при додатних x дає розв'язок поставленої задачі для напівнескінченного стрижня.

2. Напівнескінченний стрижень. Початкова температура всюди 0, крім відрізка біля межі, де вона має трикутну форму. Температура лівого кінця дорівнює нулю, тобто $u(0,t)=0 \ (t \geq 0)$

В поточному файлі створіть незгладжену `Triangle`-функцію.

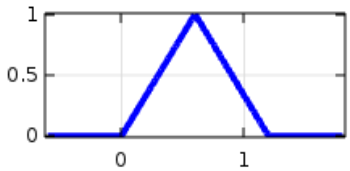
`Home` -> Definitions -> Functions -> Local – Triangle

Function name = `tri1`

`Parameters`– Lower limit, Upper limit = `0,1.2`

Зніміть прапорець згладжування.

`Smoothing` – **Size of transition zone** = `False`



Для перевірки побудуйте графік.

LMB> Plot

Створіть функцію, яка є непарним продовженням функції `tri1(x)`.

`Home` -> Definitions -> Functions -> Local – Analytic

Function name = `F2`

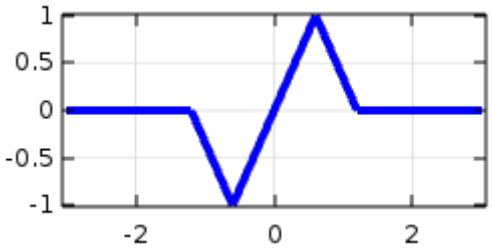
`Definition` – **Expression** = `tri1(x)-tri1(-x)`

– **Arguments** = `x`

Для перевірки побудуйте графік функції.

`Plot Parameters` – **Argument, Low limit, Upper limit**=`x,-L,L`

LMB> Plot



В таблицю змінних додайте нову змінну `T2`, яка обчислюватиметься за формулою (10), де в якості $\varphi(x)$ використовуватиметься функція `F2`.

Definitions - Variables 1:

Variables – **Name** = T2

– **Expression** = integrate(
F2(x+2*a*sqrt(t)*ksi/(1-abs(ksi)))*
exp(-ksi^2/(1-abs(ksi))^2)/((1-abs(ksi))^2+eps),
ksi,-1,1)/sqrt(pi)

Порівняно зі змінною T1 тут лише було замінено ім'я функції на F2.

Оновите обчислення і побудуйте графік змінної T2.

Study 1: RMB > Update Solution

1D Plot Group 1: RMB > Duplicate

1D Plot Group 2:

Data – **Time selection** = Interpolated

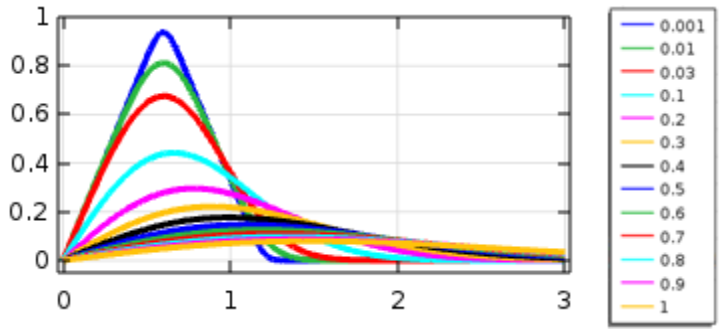
– **Times** = 0.001,0.01, 0.03 range(0.1,0.1,1)

1D Plot Group 2- Line Graph 1:

Selection – **Selection,** <> = Manual, 2 (*відрізок* [0, *L*])

y-Axis Data – **Expression** = T2

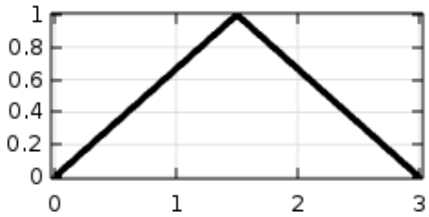
LMB > Plot



■

Розв'язок рівняння теплопровідності для стрижня скінченної довжини L, коли початкова температура задана функцією $\varphi(x)$, а кінці стержня підтримуються при нульовій температурі, можна знаходити по тій же формулі (10), якщо замість $\varphi(x)$ використати непарне періодичне продовження початкової температури з відрізка [0, L] на дійсну вісь (з відрізка [0, L] непарно на відрізок [-L,0], а потім періодично з періодом 2L).

3. Скінченний стрижень довжиною L. Температура кінців підтримується нульовою. Початкова температура $\varphi(x)$ має трикутну форму.



Створити непарну періодичну функцію, яка на відрізку [0, L] співпадає з $\varphi(x)$, можна за допомогою waveform-функції.

Home -> Definitions -> Functions -> Local – Waveform

Function name = F3

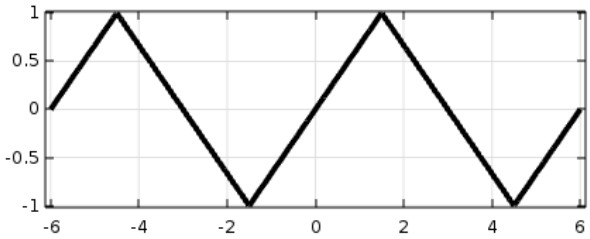
Parameters– **Type** = Triangle

Parameters– **Angular frequency** = pi/L (*період* $2\pi/\omega = 2L$)

Перемикач згладжування не встановлюйте.

Parameters– **Smoothing** = `False`

LMB > Plot



В таблицю змінних додайте нову змінну `T3`, яка обчислюватиметься за формулою (10), де в якості $\varphi(x)$ використовуватиметься функція `F3`.

Definitions - Variables 1:

Variables – **Name** = `T3`

–**Expression** = `integrate(`

```
F3(x+2*a*sqrt(t)*ksi/(1-abs(ksi)+eps))*
exp(-ksi^2/(1-abs(ksi))^2)/((1-abs(ksi))^2+eps),
                                ksi,-1,1)/sqrt(pi)
```

Порівнюючи з виразом для змінної `T2`, бачимо, що тут замінено ім'я функції на `F3` і в її аргументі, щоб уникнути ділення на нуль, використано машинне епсілон.

Оновите обчислення і побудуйте графік змінної `T3`.

Study 1: RMB > Update Solution

1D Plot Group 2: RMB > Duplicate

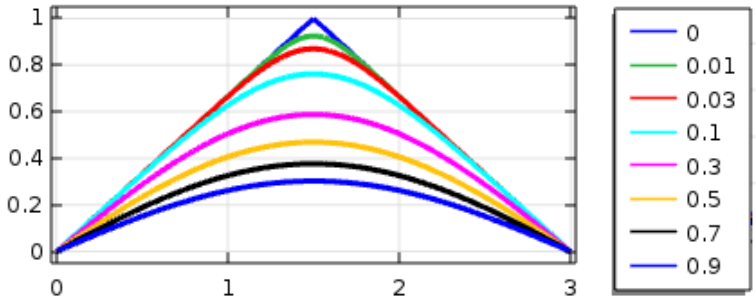
1D Plot Group 3:

– **Times** = `0, 0.01, 0.03 range(0.1,0.2,1)`

1D Plot Group 3 - Line Graph 1:

y-Axis Data – **Expression** = `T3`

LMB > Plot



Нагадаємо, що в системі існує можливість при наявності в моделі будь-якого непросторового параметра, наприклад часу, розглядати його (параметр) як просторову координату. Для цього застосовуються `Parametric Extrusion` джерела даних. Для різноманітності побудуємо графік змінної `T3` над прямокутником $[0,L]\times[0,1]$ площини $X\times t$ (хрестик $\times$ позначає декартовий добуток). Для цього спочатку створимо звужене на відрізок $[0,L]$ ДД **Study 1/Solution 1**.

Results: Data Sets - Study 1/Solution 1 : RMB > Duplicate

Data Sets - Study 1/Solution 1 (2) : RMB > Selection

Study 1/Solution 1 (2) - Selection:

`Geometric Entity Selection` – **Geometric entity level** = `Domain`

`Geometric Entity Selection` – **Selection** = `2` (*відрізок* [0,*L*])

Після створення копії джерела даних ми додали підвузол **Selection**, який дозволяє змінити рівень геометрії. В нашому випадку ми обрали відрізок [0,*L*].

Тепер створіть ДД `Parametric Extrusion 1D`.

Results - Data Sets : RMB > More Data Sets -> Parametric Extrusion 1D

Data Sets - Parametric Extrusion 1D 1 :

`Data` – **Data set** = `Study 1/Solution 1(2)`

`Data` –**Time selection** = `All`

`Settings` – **Level scale factor** = `True,1`

Остання опція означає, що в прямокутнику $[0,L]\times[0,1]$ площини $X\times t$ одинична «часова» довжина відповідатиме одиничній просторовій довжині (вздовж осі Y).

Використайте створене джерело даних для побудови профілю температури.

`Results` -> Plot Group - 2D Plot Group

2D Plot Group 4 :

`Data` – **Data set** = `Parametric Extrusion 1D 1`

`2D Plot Group 4` -> Add Plot - Line

2D Plot Group 4 - Line 1 :

`Expression` – **Expression** = `T3`

`Coloring and style` – **Line type** = `Tube`

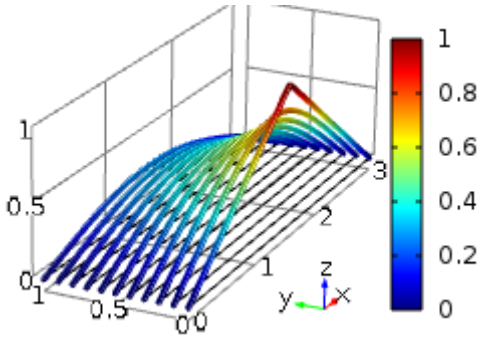
– **Tube radius expression** = `0.02`

– **Radius scale factor** = `True, 1`

RMB > Height Expression

Line 1 – Height Expression 1 : `Axis` – **Scale factor** = `True, 1`

LMB > Plot

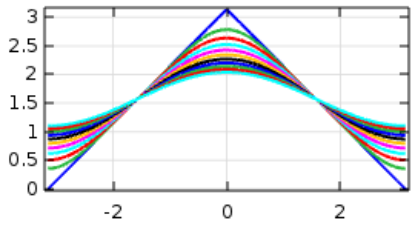


Тут вздовж осі y відкладено час t. Профіль температури зображено над відрізками `[0,L]` по часових шарах, які відповідають моментам часу `t` зі списку `range(0,0.1,1)`.

Вправа. Дослідити розповсюдження тепла в стрижні, який зігнутий в кільце одиничного радіуса. Початкова температура $\varphi(x)$ має трикутну форму.

Вказівка. Задача зводиться до розв'язання рівняння теплопровідності $u'_t = a^2 u''_{xx}$ $(t>0, -\pi<x<\pi)$ з початковою умовою $u(x,0)=\varphi(x)$ і додатковими умовами неперервності температури і теплового потоку $u(-\pi,t)=u(\pi,t)$,

$u'_x(-\pi,t)=u'_x(\pi,t)$, де $x=-\pi$ і $x=\pi$ відповідають спільному перерізу, який прийнятий як за початок так і за кінець зігнутого стрижня. Всі додаткові умови виконуватимуться, якщо застосувати періодичне продовження початкової функції $\varphi(x)$ на дійсну вісь і результівну функцію підставити в (10).



■

Розв'язок задачі розповсюдження тепла в необмеженому у всі боки двовимірному тілі з початковою температурою $u(x,y,0)=\varphi(x,y)$ дається формулою

$$u(x,y,t)=\frac{1}{4\pi a^2 t}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\varphi(\xi,\eta)\,e^{-\frac{(x-\xi)^2+(y-\eta)^2}{4a^2 t}}\,d\xi\,d\eta\,.$$

Якщо в інтегралі зробити заміну $\tau=\dfrac{\xi-x}{2a\sqrt{t}}$, $\nu=\dfrac{\eta-y}{2a\sqrt{t}}$, то ми отримаємо

$$u(x,y,t)=\frac{1}{\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\varphi\left(x+2a\sqrt{t}\,\tau,\,y+2a\sqrt{t}\,\nu\right)e^{-\tau^2-\nu^2}\,d\tau\,d\nu\,.$$

Заміна $\tau=\dfrac{\zeta}{1-|\zeta|}$, $\nu=\dfrac{\sigma}{1-|\sigma|}$ дає

$$u(x,y,t)=\frac{1}{\pi}\int_{-1}^{1}\int_{-1}^{1}\varphi\left(x+\frac{2a\sqrt{t}\,\zeta}{1-|\zeta|},\,y+\frac{2a\sqrt{t}\,\sigma}{1-|\sigma|}\right)\frac{e^{-\frac{\zeta^2}{(1-|\zeta|)^2}}}{(1-|\zeta|)^2}\,\frac{e^{-\frac{\sigma^2}{(1-|\sigma|)^2}}}{(1-|\sigma|)^2}\,d\zeta\,d\sigma\,. \qquad (11)$$

Якщо $\varphi(x,y)=\varphi_1(x)\cdot\varphi_2(y)$, то останній інтеграл розпадається в добуток двох інтегралів, які ми обчислювали вище. Але остання формула спрацьовує і тоді, коли цього немає.

Розглянемо задачу розповсюдження тепла в необмеженій площині, коли $\varphi(x,y)=1/(1+x^2+y^2)$.

Продовжимо роботу з поточним файлом. Створіть функцію $\varphi(x,y)$.

Home -> Definitions -> Functions -> Local – Analytic

Function name = F0

Definition – **Expression** = 1/(1+x^2+y^2)

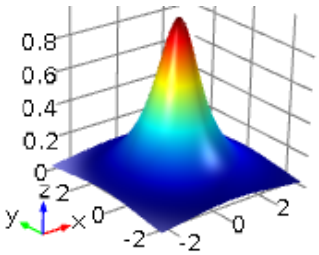
– **Arguments** = x,y

Для перевірки побудуйте графік функції.

Plot Parameters – **Argument, Low limit, Upper limit**=x,-L,L

Plot Parameters – **Argument, Low limit, Upper limit**=y,-L,L

LMB> Plot



В таблиці змінних створіть нову змінну `T0`, яка обчислюватиметься за формулою (11), де в якості $\varphi(x, y)$ використовуватиметься функція `F0`.

Definitions - Variables 1:

Variables – **Name** = `T0`

– **Expression** = `integrate(integrate(`
```
F0(x+2*a*sqrt(t)*ksi/(1-abs(ksi)),
                y+2*a*sqrt(t)*psi/(1-abs(psi)))*
exp(-ksi^2/(1-abs(ksi))^2)/((1-abs(ksi))^2+eps)*
exp(-psi^2/(1-abs(psi))^2)/((1-abs(psi))^2+eps),
                ksi,-1,1), psi,-1,1)/pi
```

Виконайте обчислення.

Study 1: RMB > Compute

Для побудови двовимірного графіка нам знадобиться `Grid 2D` джерело даних.

`Results` -> Data Set - More Data Sets -> Grid 2D

Results - Data Sets - Grid 2D 1 :

Data – **Source** = `Data set`

– **Data set** = `Study 1/Solution 1(1)`

Paramemter Bounds –

– **First parameter** – **Minimum, Maximum** = `-L,L`

– **Second parameter** – **Minimum, Maximum**= `-L,L`

Тепер побудуйте графік змінної `T0` «над» цим ДД в який-небудь момент часу.

`Results` -> **Plot Group** -> 2D Plot Group

2D Plot Group 5:

Data – **Data set** = `Grid 2D 1`

Data – **Time** = `0.2`

`2D Plot Group 5`-> **Add Plot** -> Surface

2D Plot Group 5 – Surface 1:

Expression – **Expression** = `T0`

Range – **Manual color range** = `True`

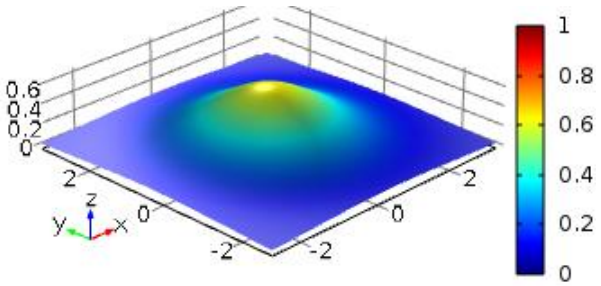
– **Minimum, Maximum** = `0,1`

RMB> Height Expression

2D Plot Group 5 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = `True,2`

LMB> Plot



Для наочності побудуйте графік перерізів поверхні `T0` вздовж напрямку $y = x$. Для цього знадобиться змінна, яка співпадатиме зі значеннями `T0` вздовж цього напрямку. Створіть іі в таблиці змінних.

Definitions - Variables 1:

`Variables` – **Name** = `T0xy`

– **Expression** = `subst(T0,y,x)`

Тут використано оператор підстановки: у виразі `T0` змінну `y` замінено на змінну `x`.

Оновите обчислення.

Study 1: RMB > Update Solution

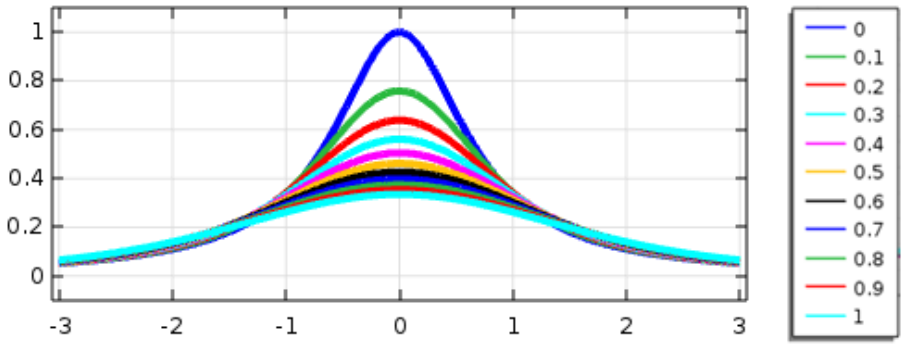
Тепер можна будувати графіки змінної `T0xy`.

1D Plot Group 1: RMB > Duplicate

1D Plot Group 6 - Line Graph 1:

`y-Axis Data` – **Expression** = `T0xy`

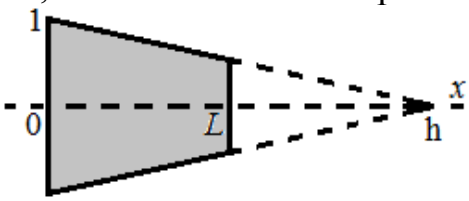
LMB > Plot



Вправа 1. Рівняння подовжніх коливань конічного стрижня має вигляд

$$a^2 \frac{\partial}{\partial x}\left[(h-x)^2 \frac{\partial u}{\partial x}\right] = (h-x)^2 \frac{\partial^2 u}{\partial t^2},\ a^2 = \frac{E}{\rho}. \qquad (12)$$

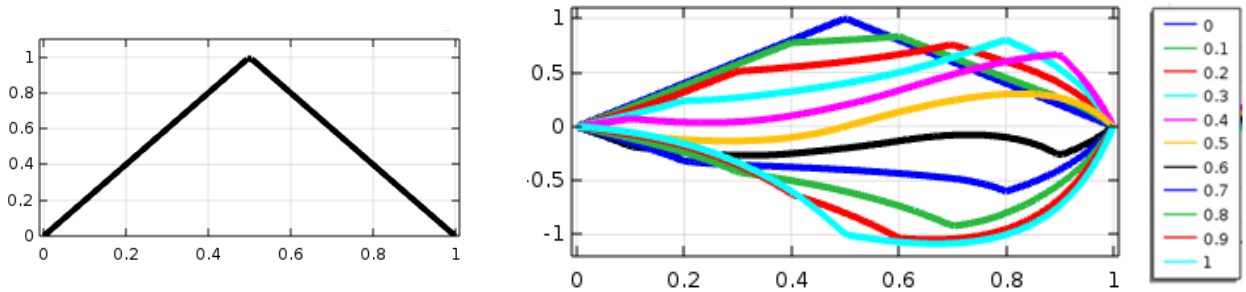
де h - висота повного конуса, частиною якого є стрижень ([2], с. 86, задача 3).



Зробивши заміну $v(x,t) = (h-x)\cdot u(x,t)$ ([2], с. 29, задача 2), отримуємо класичне хвильове рівняння $a^2 v''_{xx} = v''_{tt}(x,t)$ відносно функції $v(x,t)$. Початкова умова набуває вигляду $v(x,0) = (h-x)\varphi(x)$, де $\varphi(x) = u(x,0)$.

Якщо кінці конічного стрижня нерухомі, тобто $u(0,t)=0$, $u(L,t)=0$, то для функції $v(x,t)$ виконуватимуться такі ж граничні умови $v(0,t)=0$, $v(L,t)=0$. Таким чином, розв'язавши для функції $v(x,t)$ крайову задачу, наприклад, методом продовження, розв'язок рівняння (12) можна отримати за формулою $u(x,t)=v(x,t)/(h-x)$.

Розв'язати задачу подовжніх коливань конічного стрижня при $a=1$, $L=1$, $h=1.2$, кінці якого затиснені, початкова швидкість коливань нульова, початкове зміщення $\varphi(x)$ в формі трикутника. На наступному рисунку ліворуч зображено функцію $\varphi(x)$, а праворуч – графіки розв'язку $u(x,t)$.



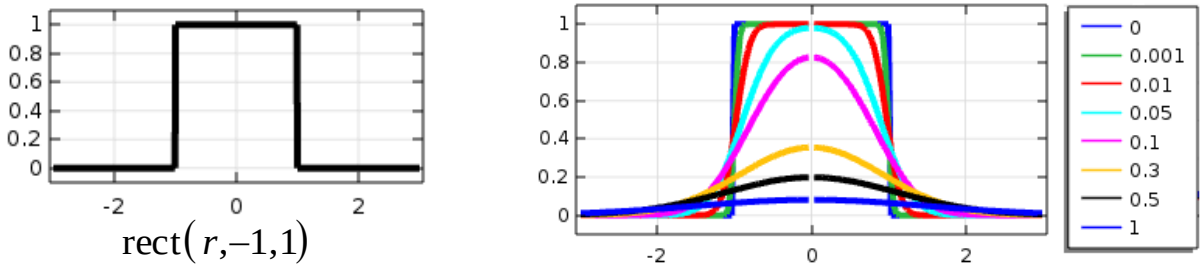
Вправа 2. Сферично-симетрична задача теплопровідності описується рівнянням (порівняйте з п. 3.28)

$$\frac{\partial u}{\partial t}=a^2\frac{1}{r}\frac{\partial^2}{\partial r^2}(ru).$$

Зробивши заміну $v(r,t)=r\cdot u(r,t)$, матимемо $v'_t=a^2 v''_{rr}$. Початкова умова $u(r,0)=\varphi(r)$ перетворюється на $v(r,0)=r\cdot\varphi(r)$.

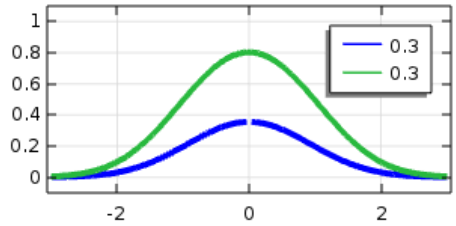
Дослідити задачу остигання нескінченного простору з сферично-симетричною початковою температурою $\varphi(r)=\mathrm{rect}(r,-1,1)$, джерела тепла відсутні, $a=1$.

На наступному рисунку ліворуч зображено функцію $\varphi(r)$, а праворуч – графіки розв'язку $u(r,t)$ в послідовні моменти часу.



$\mathrm{rect}(r,-1,1)$

Криві праворуч схожі на графіки, отримані нами при дослідженні розповсюдження тепла в нескінченному стрижні з такою ж початковою температурою. Але при порівнянні розв'язків в один і той же момент часу можна побачити, що остигання стрижня йде повільніше. На наступному графіку показані розв'язки поточної вправи (синя крива) і змінної `T1` (зелена крива) з прикладу 3 в момент часу `t=0.3`.



Зауваження. Всі розглянуті в цьому параграфі приклади можна розв'язати, використовуючи відповідні математичні інтерфейси. Спробуйте це зробити і порівняйте отримані розв'язки. Наведені приклади лише мали метою демонстрування можливостей інструментарію роботи з функціями.

6.3. Інтерполяційні функції.

Коли функція задана дискретним набором точок і потрібно визначити її значення в проміжних точках, то вдаються до інтерполяції або апроксимації. При цьому невідома справжня функція замінюється деякою іншою, часто кусковою, яка має досить прості вирази на ділянках. Якщо шукана функція набуває задані значення у вузлах, то кажуть про інтерполяцію, якщо ні, то про апроксимацію. Тут ми розглянемо особливості застосування інтерполяційних функцій, приклади апроксимації було розглянуто в п. 4.1.4.

В COMSOL Multiphysics вузли типу `Interpolation` створюють кусково-поліноміальну інтерполяційну функцію, яка «проходить» крізь задані точки $\{x_i, y_i\}_{i=0}^{n}$, і на кожній ділянці $(x_{i-1}, x_i]_{i=1}^{n}$ зображується окремим поліномом. Кусково-стала та ламана є поодинокими випадками кусково-поліноміальної інтерполяції, коли на кожній ділянці функція зображується прямою лінією. Але частіше використовують кубічні многочлени для представлення сегментів кускового полінома.

В COMSOL Multiphysics 5.1 у вузлах типу `Interpolation` можна використати 4 типи інтерполяції:

- Інтерполяція найближчого сусіда (nearest neighbor interpolation) будує кусково сталу функцію зі значенням y_i найближчої точки (x_i, y_i);
- Лінійна інтерполяція (linear interpolation) полягає в тому, що задані точки з'єднуються прямолінійними відрізками;
- Кусково-кубічна інтерполяція Ерміта (piecewise-cubic Hermite polynomial) на кожній ділянці $(x_{i-1}, x_i]_{i=1}^{n}$ використовує кубічні поліноми, які поєднуються в неперервну криву з неперервними першими похідними і збереженою монотонністю між вузлами;
- Кубічна сплайн-інтерполяція (cubic spline interpolation) створює неперервний кусково-кубічний поліном з неперервними першими і другими похідними.

Зазвичай будуються функції одного аргументу, але є можливість створювати інтерполяційні функції двох і трьох змінних (при завантаженні даних з файлу).

Розглянемо пояснювальні приклади.

Приклад 1. Порівняння різних типів інтерполяції.

Дано 6 точок з координатами `{{0,1},{1,2},{3,5},{4,4},{5,0},{6,3}}`. На них побудувати 4 можливі інтерполяційні функції і накреслити їх графіки.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = `None`

Побудуйте інтерполяційну функцію найближчого сусіда (nearest neighbor interpolation).

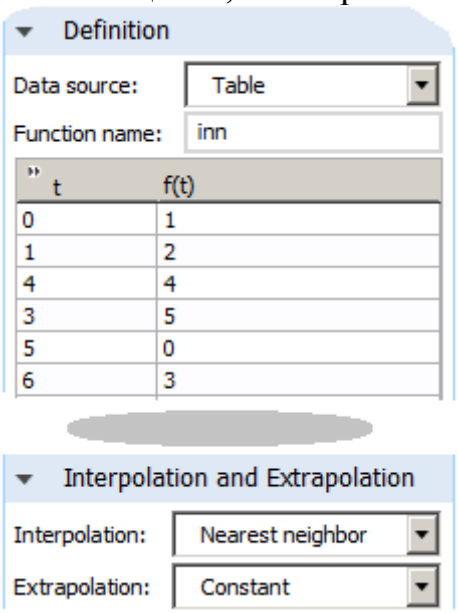
Global Definitions: RMB > Functions – Interpolation

Function name = `inn`

При побудові інтерполяційної функції координати вузлів можна обирати з текстового файлу або задавати безпосередньо в таблиці вікна властивостей.

`Definition` – **Data source** = `Table` (*координати вузлів в таблиці*)

Введіть координати вузлів в таблиці так, як зображено на наступному рисунку.



Зверніть увагу, що абсциси вузлів не зобов'язані прямувати в порядку зростання.

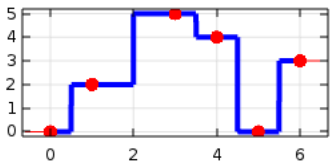
Оберіть тип інтерполяції.

`Interpolation and Extrapolation` – **Interpolation** = `Nearest neighbor`

Поведінка функції поза інтервалом [0,6] нас зараз не цікавить, тому опцію **Extrapolation** залишаємо незмінною (`Constant`).

Виконайте контрольну побудову графіка функції.

LMB> Plot



На тій же множині точок створіть кусково-лінійну інтерполяційну функцію.

Global Definitions - Interpolation 1: RMB > Duplicate

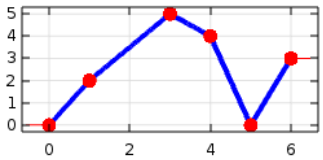
Global Definitions - Interpolation 2:

Function name = `ili`

Interpolation and Extrapolation – **Interpolation** = Linear

Виконайте контрольну побудову графіка функції.

LMB> Plot



Створіть кусково-кубічну інтерполяційну функцію Ерміта.

Global Definitions - Interpolation 1: RMB > Duplicate

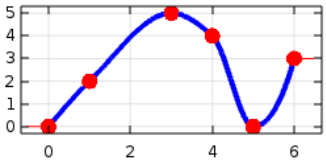
Global Definitions - Interpolation 3:

Function name = ipc

Interpolation and Extrapolation – **Interpolation** = Piecewise cubic

Виконайте контрольну побудову графіка функції.

LMB> Plot



Ще раз дублюйте вузол **Interpolation 1** і виберіть Cubic spline тип інтерполяції.

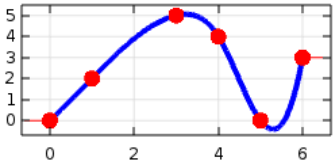
Global Definitions - Interpolation 1: RMB > Duplicate

Global Definitions - Interpolation 4:

Function name = ics

Interpolation and Extrapolation – **Interpolation** = Piecewise cubic

LMB> Plot



Побудуйте графіки інтерполяційних функцій на спільних осях. Щоб це зробити знадобиться джерело даних Grid 1D.

Results -> Data Set - More Data Sets -> Grid 1D

Data Sets - Grid 1D 1 :

Data – **Source, Function** = Function, All

Parameter Bounds – **Name** = t

Parameter Bounds – **Minimum, Maximum** = 0,6

Для зображення вузлів потрібно ще точкове ДД типу Cut Point 1D.

Results -> Data Set - More Data Sets -> Cut Point 1D

Data Sets - Cut Point 1D 1 :

Data – **Data set** = Grid 1D 1

Point Data – **Entry method, x** = Coordanates,range(0,1,6)

Тепер можна будувати графіки.

Results -> Plot Group - 1D Plot Group

1D Plot Group 1:

Data Set – **Data** = Grid 1D 1

Будуємо зображення вузлів інтерполяції.

1D Plot Group 1 -> Add Plot - Point Graph

1D Plot Group 1 - Point Graph 1:

Data Set – Data = Cut Point 1D 1
y-Axis Data – Expression = ics(t)
x-Axis Data – Parameter, Expression = Expression,t
Задайте колір і розмір точок.
Coloring and Style – Line style = Solid
– Color = Magenta
– Width = 8

Тепер побудуйте графіки інтерполяційних функцій.

1D Plot Group 1 -> Add Plot - Line Graph

1D Plot Group 1 - Line Graph 1:
y-Axis Data – Expression = inn(t)
x-Axis Data – Parameter, Expression = Expression,t
Coloring and Style –Width = 3
Legends – Show legends, Legends = True,Manual
– Legends = nearest

1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 2:
y-Axis Data – Expression = ili(t)
Legends – Legends = linear

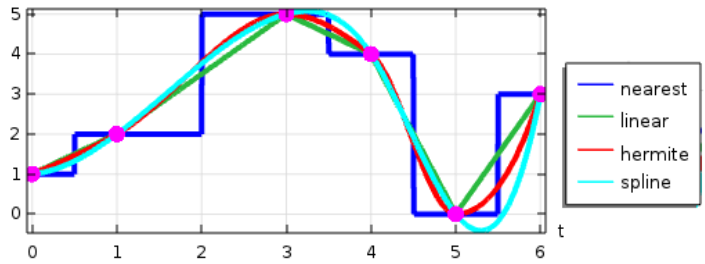
1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 3:
y-Axis Data – Expression = ipc(t)
Legends – Legends = hermite

1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 4:
y-Axis Data – Expression = ics(t)
Legends – Legends = spline

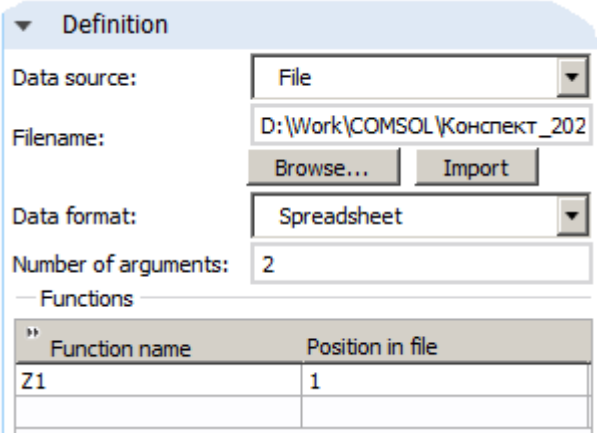
LMB > Plot



Побудуємо інтерполяційну функцію двох змінних. Це можна зробити, лише коли дані завантажуються з текстового файлу. Попередні дані теж можна було б завантажити з файлу. При чому однотипні дані (наприклад, абсциси) треба розміщати в один стовпець, а значення функції – в інший.

```
0.0   1.0
1.0   2.0
3.0   5.0
. . . .
```

Дані в рядку можна розділяти проміжком, комою або табуляцією. Фактично такий файл повторює спосіб введення даних в панелі властивостей `Interpolation` вузлів. При завантаженні даних з файлу стовпців може бути більше і в ПВ функції з'являється поле **Number of arguments,** в якому вказується кількість аргументів функції (від 1 до 3). Файли з таким способом інтерпретування даних (однотипні дані по стовпцях) в COMSOL називаються `spreadsheet` файлами, і цей тип обирається в полі **Data format**.



Всі стовпці з номерами більшими за кількість аргументів інтерпретуються, як значення функцій (по одному стовпцю для кожної функції), і імена цих функцій вказуються в таблиці **`Functions`** ПВ вузла.

Створіть текстовий файл з наступним вмістом і збережіть його з ім'ям, наприклад, `XYZ1.txt`.

```
0   0   0
0   1   1
0   2   2
1   0   1
1   1   1
1   2   1.5
2   0   1
2   1   2
2   2   1
```

Тут перші два стовпця представлятимуть аргументи (x, y) функції, а третій – її значення у відповідних точках.

Тепер побудуйте інтерполяційну функції двох змінних, використовуючи дані з цього файлу. Продовжить роботу з поточним файлом.

Global Definitions: RMB > Functions – Interpolation

`Definition` – **Data source** = `File` (*координати вузлів в файлі*)

Зверніть увагу на те, що поля з іменем функції вже немає , а для назв функцій з'являється окрема таблиця.

Натисніть кнопку Browse, знайдіть і виберіть файл `XYZ1.txt`.

`Definition` – **Filename** = `D:\Work\COMSOL\XYZ1.txt`

Залиште встановлений за замовчуванням `spreadsheet` формат файлу і задайте кількість аргументів функції.

`Definition` – **Number of arguments** = `2`

В таблиці **Functions** задайте ім'я функції.

Definition – **Functions – Function name** = Z1

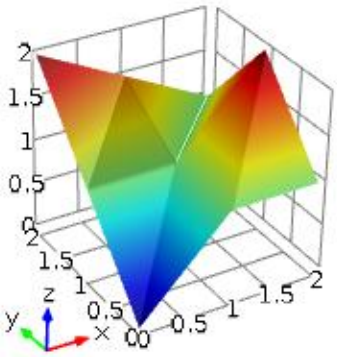
– **Functions – Position in file** = 1

В полі **Position in file** вказується номер стовпця (1 – перший після стовпців аргументів) з даними для зазначеної функції (отже 2 аргументи плюс 1, тобто третій стовпець файлу).

Для функції двох аргументів можливі лише інтерполяція найближчого сусіда і лінійна інтерполяція. Оберіть друге.

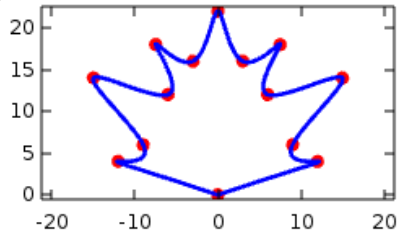
Interpolation and Extrapolation – **Interpolation** = Linear

LMB > Plot



Тепер функція Z1(x,y) створена. Зазвичай інтерполяція експериментальних даних потрібна для того, щоб аналітично описати їх залежності одне від одного. Потім побудовані функції використовуються у виразах матеріальних коефіцієнтів, початкових або граничних умовах. Але можна запропонувати і інші застосування, наприклад, геометричні.

Приклад 2. Побудувати фігуру (наступний рисунок ліворуч), боковий контур якої є параметричним кубічним сплайном (наступний рисунок праворуч).



Фігуру, зображену ліворуч, можна створити, застосовуючи різні графічні інструменти. Тут ми побудуємо її по параметричному рівнянню поверхні, в якому для тренування читача головні використовувані функції будуть інтерполяційними.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Untitled.mph (root): Unit System – <> = None

Почніть з контура фігури. Система не генерує параметричні сплайни. Тому потрібно створити два різні інтерполяційні сплайни такі що, використовуючи їх в якості координатних функцій $x_c(u), y_c(u)$, можна було б побудувати бажану криву. Нам знадобляться координати вузлів інтерполяції (x_i, y_i), а ще цим вузлам потрібно призначити монотонно зростаючі значення параметра u_i,

наприклад, `0,1,2,...`. Таким чином, по координатам вузлів (u_i, x_i) і (u_i, y_i) потрібно побудувати сплайни $x_c(u), y_c(u)$.

В попередньому прикладі при побудові інтерполяційних функцій ми створювали одну функцію в одному вузлі. Але при введенні даних з текстового файлу в одному вузлі можна створювати декілька функцій. Для `spreadsheet` файлів діють наступні домовленості. В першому стовпці завжди задаються значення першого аргументу інтерполяційної функції. В залежності від значення поля **Number of arguments** другий стовпець може бути значенням другого аргументу інтерполяційної функції або значенням самої функції. Теж стосується наступних стовпців – вони можуть бути значеннями подальших аргументів або значеннями іншої функції. Отже при завантаженні даних з файлу можна будувати інтерполяційні функції однієї, двох і трьох змінних, і одночасно вводити дані кількох інтерполяційних функцій. Тут ми продемонструємо другу можливість – одночасно завантажимо дані для двох інтерполяційних сплайнів $x_c(u), y_c(u)$.

Створіть текстовий файл з наступним вмістом і збережіть його, наприклад, з ім'ям `XYZ2.txt`.

```
0     0      0
1     -12    4
2     -9     6
3     -15    14
4     -6     12
5     -7.5   18
6     -3     16
7     0      22
8     3      16
9     7.5    18
10    6      12
11    15     14
12    9      6
13    12     4
14    0      0
```

Тут перший стовпець являє значення параметра *u* кривої у вузлах. Другий і третій – є абсцисами і ординатами точок інтерполяції.

Global Definitions: RMB > Functions – Interpolation

Definition – **Data source** = `File` (*координати з файлу*)
– **Filename** = `D:\Work\COMSOL\XYZ2.txt`

Залиште встановлений за замовчуванням `spreadsheet` формат файлу.

Definition – **Number of arguments** = `1` (*кількість аргументів функцій*)

В таблиці **Functions** задайте імена функцій.

Definition – **Functions** – **Function name** = `X`
– **Position in file** = `1`

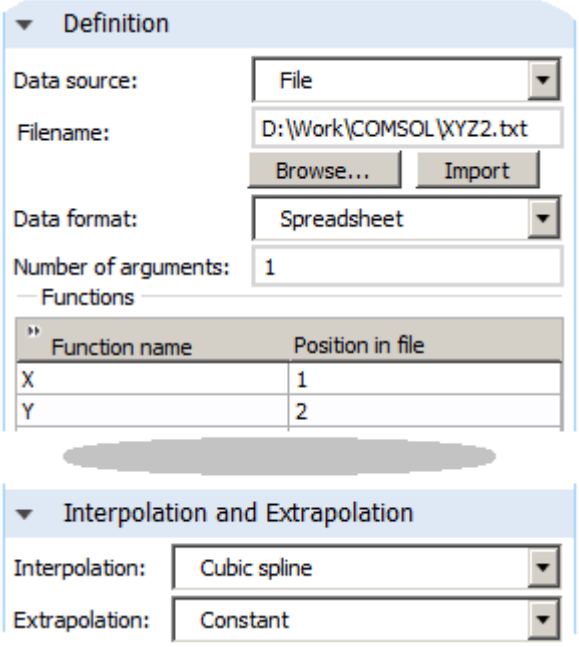
Definition – **Functions** – **Function name** = `Y`
– **Position in file** = `2`

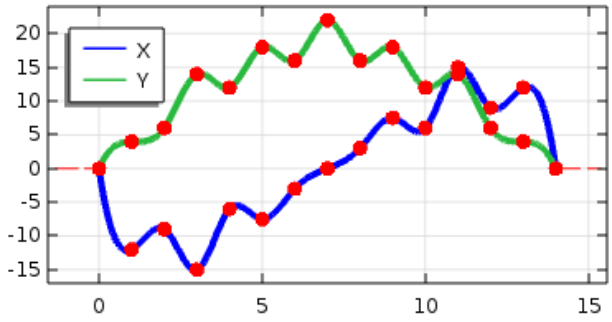
В полі **Position in file** вказується номер стовпця даних зазначеної функції з початком відліку після стовпця (стовпців) аргументу.

Оберіть тип інтерполяції. Нехай це буде сплайн-інтерполяція.

Interpolation and Extrapolation – **Interpolation** = Cubic spline

LMB > Plot (наступний рисунок праворуч)





Перед тим, як будувати поверхню, перевіримо, що за допомогою функцій X і Y ми в змозі побудувати потрібну нам параметричну криву.

Для параметра кривої створіть одновимірне джерело даних.

Results -> Data Set - More Data Sets -> Grid 1D

Data Sets - Grid 1D 1 :

Data – **Source, Function** = Function, All

Paramemter Bounds – **Name** = u

– **Minimum, Maximum** = 0,14

Для зображення вузлів потрібно точкове ДД типу Cut Point 1D.

Results -> Data Set - More Data Sets -> Cut Point 1D

Data Sets - Cut Point 1D 1 :

Data – **Data set** = Grid 1D 1

Point Data – **Entry method, x** = Coordanates, range(0,14)

Тепер можна будувати параметричний сплайн.

Results -> Plot Group - 1D Plot Group

1D Plot Group 1:

Data Set – **Data** = Grid 1D 1

1D Plot Group 1 -> Add Plot - Line Graph

1D Plot Group 1 - Line Graph 1:

y-Axis Data – **Expression** = Y(u)

x-Axis Data – **Parameter, Expression** = Expression,X(u)

Coloring and Style –**Width, Color** = 3, Blue

Для наочності зобразіть вузли інтерполяції.

1D Plot Group 1 -> Add Plot - Point Graph

1D Plot Group 1 - Point Graph 1:

Data Set – Data = Cut Point 1D 1

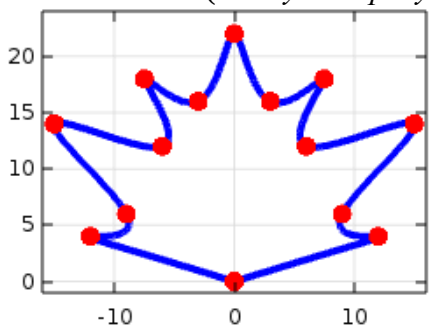
y-Axis Data – Expression = Y(u)

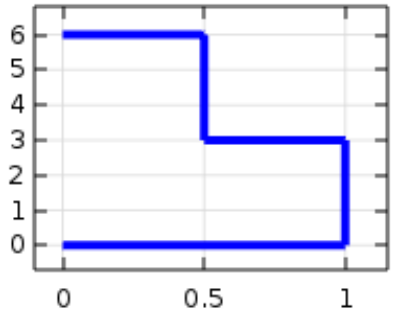
x-Axis Data – Parameter, Expression = Expression,X(u)

Задайте колір і розмір точок.

Coloring and Style – Color, Width = Red,7

LMB > Plot (*наступний рисунок ліворуч*)





Для побудови поверхні по параметричному рівнянню, нам знадобляться координатні функції східчастої ламаної, яка зображена на попередньому рисунку праворуч.

Для керування висотою фігури створіть параметр моделі H.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметр H=6.

Призначте вузлам ламаної наступні значення параметра v=0,1,1+H/2, 1.5+H/2,1.5+H,2+H. Це значення відстані вздовж ламаної від поточного вузла до початкового (0,0). Але це не обов'язково, потрібно лише щоб ці значення були монотонно зростаючими. Координати вузлів (x_i, y_i) візьміть з рисунка: (0,0),(1,0),(1,H/2),(0.5,H/2),(0.5,H),(0,H). По цим даним потрібно побудувати дві функції (ламані) $x_s(v), y_s(v)$, які при значеннях аргументу v_i набувають значення x_i і y_i відповідно. Оскільки параметри не можна використовувати в файлах даних, то для двох інтерполяційних ламаних створіть два вузли з задаванням даних в таблицях (тоді параметри моделі можна використовувати).

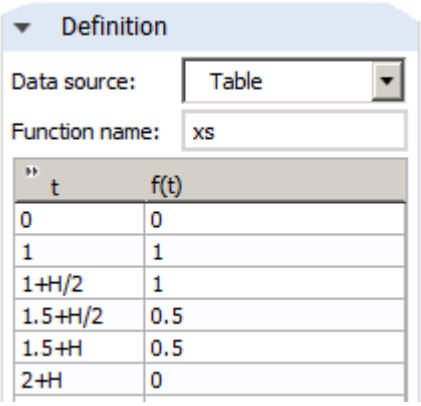
Побудуйте кусково-лінійну функцію $x_s(v)$.

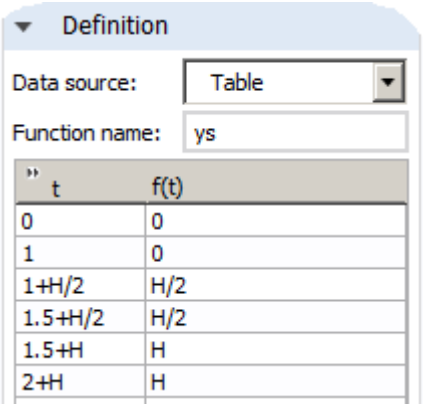
Global Definitions: RMB > Functions – Interpolation

Function name = xs

Definition – Data source = Table (*координати вузлів в таблиці*)

Введіть координати вузлів в таблиці так, як зображено на наступному рисунку ліворуч.





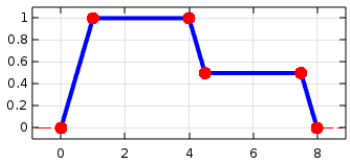
Оберіть тип інтерполяції.

Interpolation and Extrapolation – **Interpolation** = Linear

Опцію **Extrapolation** залиште незмінною (Constant).

Виконайте контрольну побудову графіка функції.

LMB> Plot



Тепер побудуйте функцію $y_s(v)$.

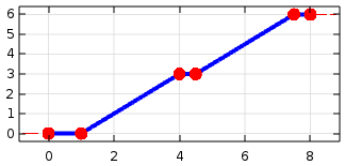
Global Definitions – Interpolation 2(xs): RMB > Duplicate

Function name = ys

Введіть координати вузлів функції такі, як зображено на попередньому рисунку праворуч.

Виконайте контрольну побудову графіка функції.

LMB> Plot



Перевірте, що $x_s(v), y_s(v)$ є координатними функціями в параметричному рівнянні східчастої кривої. Для цього створіть потрібне ДД і побудуйте криву.

Data Sets – Grid 1D 1 : RMB > Duplicate

Data Sets – Grid 1D 2 :

Paramemter Bounds – **Name** = v

Paramemter Bounds – **Minimum, Maximum** = 0, 2+H

Results -> Plot Group – 1D Plot Group

1D Plot Group 2:

Data Set – **Data** = Grid 1D 2

1D Plot Group 1 -> Add Plot – Line Graph

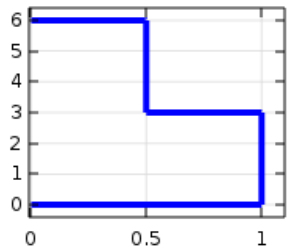
1D Plot Group 2 – Line Graph 1:

y-Axis Data – **Expression** = ys(v)

x-Axis Data – **Parameter, Expression** = Expression,xs(v)

Coloring and Style –**Width, Color** = 3, Blue

LMB> Plot



Параметричне рівняння потрібної поверхні матиме вигляд

$$\begin{aligned} Sx(u,v) &= xs(v)\cdot X(u) \\ Sy(u,v) &= xs(v)\cdot Y(u) \\ Sz(u,v) &= ys(v) \end{aligned} \qquad (1)$$

Воно схоже на рівняння поверхні обертання східчастої ламаної навкруги осі Z, де замість множників $\cos u$, $\sin u$ стоять координатні функції $X(u), Y(u)$ параметричного сплайна, побудованого вище.

Для побудови зображення поверхні по параметричним рівнянням потрібно мати `Parameterized Surface` джерело даних. Оскільки точки поверхні будуть розташовані в тривимірному просторі, то додатково знадобиться тривимірне ДД.

`Results` -> Data Set – More Data Sets -> Grid 3D

Results – Data Sets - Grid 3D 3 :

Data – **Source, Function** = `Function, All`

Paramemter Bounds –

– **First parameter** – **Minimum, Maximum** = `-16,16`

– **Second parameter** – **Minimum, Maximum** = `0,22`

– **Third parameter** – **Minimum, Maximum** = `-1,1+H`

Тепер побудуйте `Parameterized Surface` джерело даних.

`Results` -> Data Set - More Data Sets -> Parameterized Surface

Data Sets - Parameterized Surface 1 :

Data – **Data set** = `Grid 3D 3`

Parameter – **First parameter** – **Name** = `u`

– **First parameter** – **Minimum, Maximum** = `0,14`

– **Second parameter** – **Name** = `v`

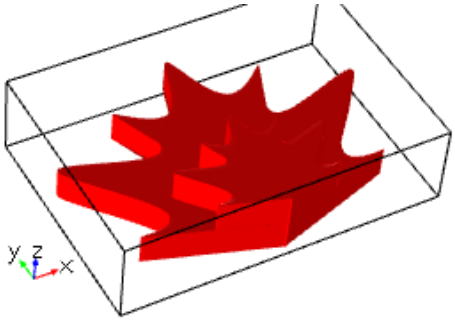
– **Second parameter** – **Minimum, Maximum** = `0,2+H`

Expression – **x** = `xs(v)*X(u)`

– **y** = `xs(v)*Y(u)`

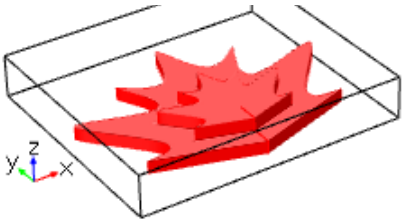
– **z** = `ys(v)`

LMB > Plot



Розфарбовувати поверхню цього ДД в тривимірній графічній групі ми не станемо.

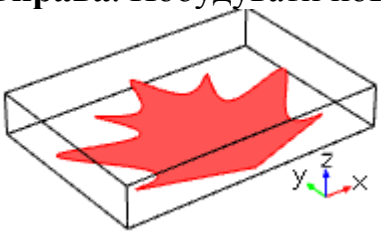
Зауваження 1. Якщо ви побажаєте зсунути нижню частину фігури відносно верхньої вниз на ΔY одиниць, то в рівняннях (1) від функції $Y(u)$ слід відняти сталу ΔY, записавши друге рівняння у вигляді $Sy(u,v) = xs(v)\cdot(Y(u) - \Delta Y)$. Звичайно знадобиться змінити діапазони батьківського джерела даних (в цьому прикладі **Grid 3D 3**).

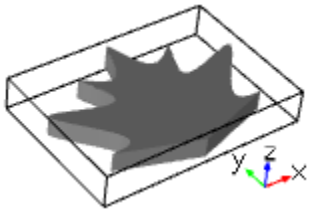


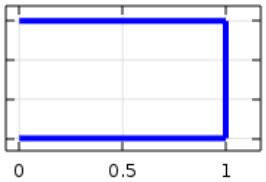
Зауваження 2. Розрахункову геометрію у формі бажаної фігури по параметричним рівнянням (1) вам навряд чи вдасться побудувати. Це пов'язано з тим, що при побудові геометрії використовується метод граничного моделювання (**b**oundary **rep**resentation або B-rep метод). B-rep модель твердого тіла полягає в тому, що поверхні представляються за допомогою неоднорідних раціональних B-сплайнів (**N**on-**U**niform **R**ational **B-S**plines, NURBS). Як налідок, в гілці Geometry координатні функції параметричної поверхні повинні бути підходящим чином переобчислені, і COMSOL вірогідно, в нашому випадку з цим не впорається. Але джерело даних по параметричним рівнянням поверхні створити можна, що ми і зробили.

Зауваження 3. В цьому параграфі ми ще не торкнулися кількох моментів. В текстових файлах, призначених для інтерполяційних функцій, можна використовувати коментарі, рядки яких починаються з символу `'%'` (процент). Для таких рядків діють певні домовленості і за їх допомогою можна керувати даними файлу. Ми не обговорювали також сенс полів **Define primitive function** і **Define inverse function**. Крім того, користувач може відрегулювати параметри екстраполяції (поведінку інтерполяційної функції поза зоною її визначення) в розділі **`Interpolation and Extrapolation`**. Ще, окрім `spreadsheet` способу розміщення даних в файлі, можна використовувати `sectionwise` і `grid` способи. Вони дещо складніші за `spreadsheet`, і читач може розібратися з цим самостійно. Не обговорили ми також сенс прапорця **Define random function**. З ним ми познайомимо читача в цій главі трохи пізніше.

Вправа. Побудувати поверхню у вигляді плоского листа та товстого листа.







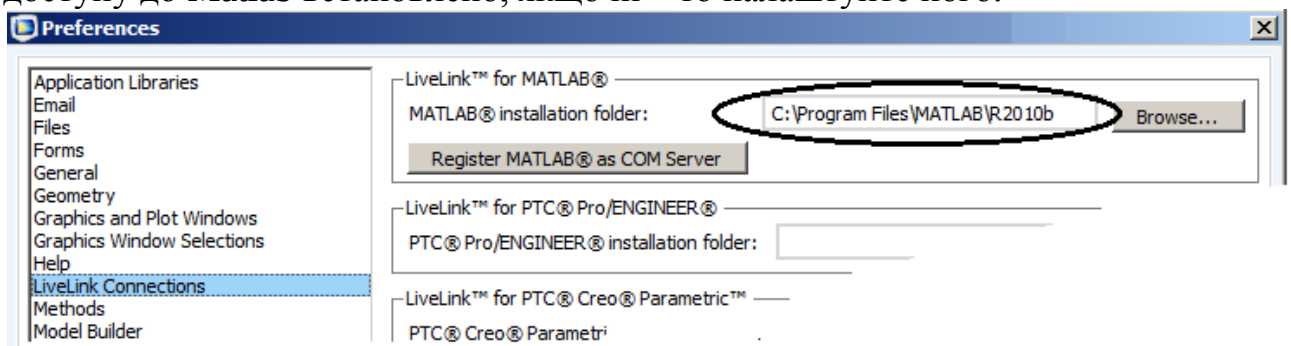
Вказівка. Параметричним рівнянням плоского листа буде `x=v*X(u)`, `y=v*Y(u)`, `z=0`, де $0 \le v \le 1$, $0 \le u \le 14$, і `X(u)`,`Y(u)` – функції, побудовані в

розглянутому прикладі. Параметричне рівнянні товстого листа матиме вид (1), де в якості функцій $xs(v), ys(v)$ слід використати координатні функції ламаної, зображеної на попередньому рисунку праворуч.

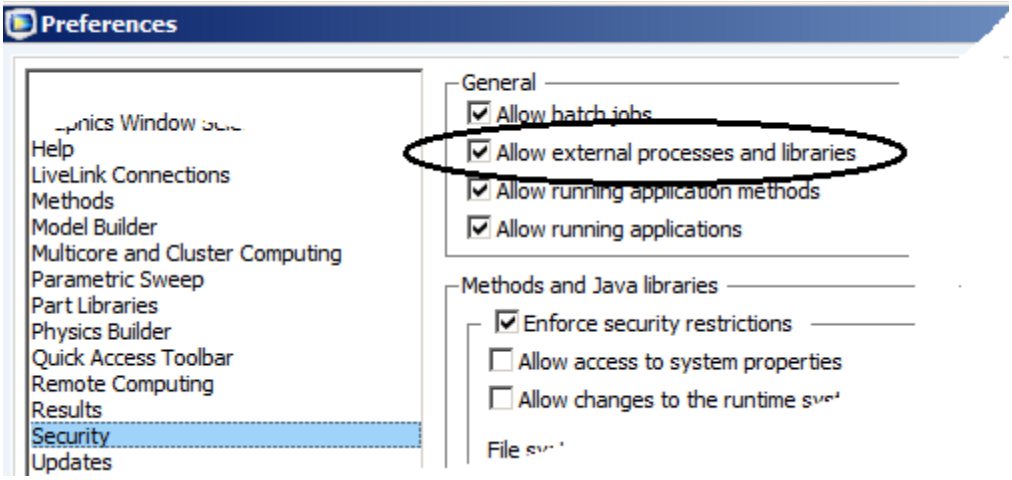
6.4. Функції Matlab.

Інколи вбудованих шаблонів функцій недостатньо. Оскільки COMSOL Multiphysics не має повноцінної мови програмування, то на допомогу може прийти Matlab, з яким у COMSOL Multiphysics добре налаштована взаємодія. Зокрема існує можливість виконувати в COMSOL функції Matlab. В попередніх главах ми вже наводили приклади їх застосування (див. п. 5.4.2). Тут ми лише нагадаємо, як це робиться. Якщо ви не знайомі з Matlab, то пропустіть цей параграф.

Ми припускаємо, що Matlab працював на вашому комп'ютері до того, як була встановлена система COMSOL (при установці COMSOL Multiphysics розпізнає версію Matlab і запам'ятовує шляхи на його файли). Якщо Matlab встановлювався пізніше, то в COMSOL знадобиться додаткове налаштування. Для цього виконайте команду меню File - Preferences. Відкриється однойменне вікно. Розкрийте сторінку **`LiveLink Connections`** і переконайтеся, що шлях доступу до Matlab встановлено, якщо ні – то налаштуйте його.



Далі в налаштуваннях COMSOL Multiphysics потрібно увімкнути підтримку зовнішніх функцій. У вікні Preferences розкрийте сторінку **`Security`** і встановіть прапорець `Allow external processes and libraries`.



Щоб налаштування вступили в дію, натисніть кнопку OK, закрийте і знову відкрийте COMSOL Multiphysics. Тепер, коли буде виконуватися модель з

Matlab функціями, COMSOL автоматично завантажить Matlab процес, який виконає функції і поверне значення в модель.

Зауваження. В різних версіях COMSOL Multiphysics 5.X назви сторінок, полів і прапорців вікна Preferences можуть трохи відрізнятися.

Ще одна загальна проблема виникає у Matlab-а. Файли функцій, які ви будете використовувати в COMSOL, Matlab повинен знаходити за будь яких обставин. В COMSOL Multiphysics 5.1 найпростішим, здається, скопіювати файли Matlab-функції в той же каталог, в якому ви зберегли модель (тобто `*.mph` файл). Можна також запам'ятати в шляхах доступу Matlab ім'я каталогу з потрібними функціями (в Matlab команда меню Set Path). В ОС Windows, щоб система COMSOL Multiphysics виявляла Matlab функції, їх можна зберігати в спеціальній папці, і створити змінну середовища COMSOL_MATLAB_PATH, яка буде вказувати шлях на цей каталог.

Приклад. Застосування Matlab функцій.

Розв'язання. Створіть в текстовому редакторі Matlab наступну функцію.

```
function out = testfunc(x)
    out=(x-3).^2/3;  % використання крапки обов'язково
end
```

Збережіть функцію в файлі з ім'ям `testfunc.m`. На інструментальній стрічці Matlab (закладка `HOME`) натисніть командну кнопку Set Path, і у однойменному вікні додайте шлях на папку, в яку було збережено цей файл.

Для перевірки працездатності функції в командному вікні Matlab виконайте наступний код

```
x=0:0.1:6; y=testfunc(x); plot(x,y,'LineWidth',3);
```

В результаті повинен бути побудований графік параболи.

В COMSOL створіть новий файл.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = None

Збережіть поки що порожню модель в файл.

Перед застосуванням Matlab-функції ви повинні повідомити систему, де знаходиться її файл. Найпростіше скопіювати його в той же каталог, в якому ви зберегли модель поточного прикладу (тобто `*.mph` файл), і в Matlab командою Set Path встановити шлях доступу до каталогу з цією функцією.

Тепер в COMSOL для побудови шаблону функції виконайте інструкцію

Global Definitions: RMB > Functions – MATLAB

В дереві моделі з'явиться вузол **MATLAB 1**. В його ПВ в розділі **`Functions`** в таблиці функцій в стовпці **Function name** введіть ім'я `testfunc`, а в стовпці **Arguments** введіть `x`.

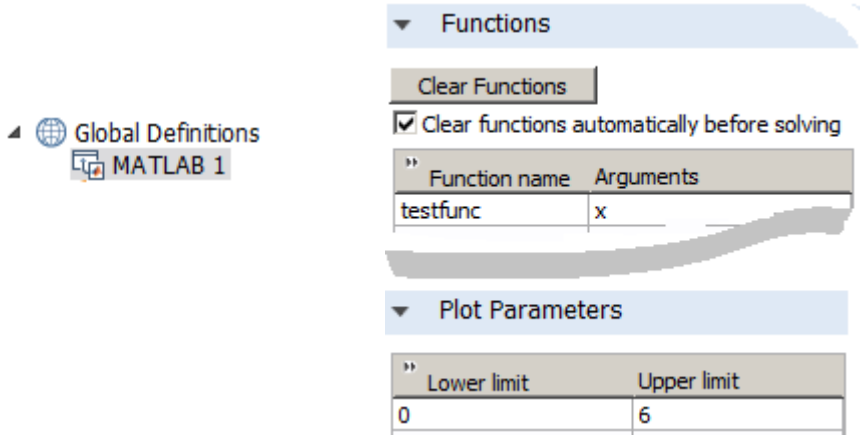
Global Definitions - MATLAB 1:

`Functions` – **Function name** = `testfunc`

– **Arguments** = `x`

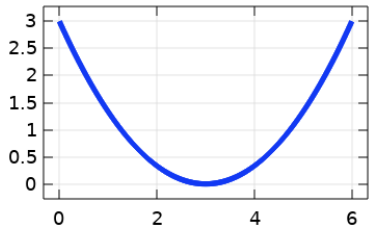
Для перевірки працездатності функції в розділі **`Plot Parameters`** введіть діапазон зміни аргументу і натисніть кнопку Plot.

`Plot Parameters` – Low limit, Upper limit = `0,6`



LMB> Plot

В пам'ять комп'ютера завантажиться командне вікно Matlab і у графічному вікні COMSOL буде побудована парабола.



Якщо буде отримано повідомлення про помилку, то скоріш за все в Matlab некоректно налаштовано шлях доступу до файлу `testfunc.m`. Перевірте його.

Для тренування створіть в Matlab функцію двох змінних і протестуйте її працездатність в COMSOL.

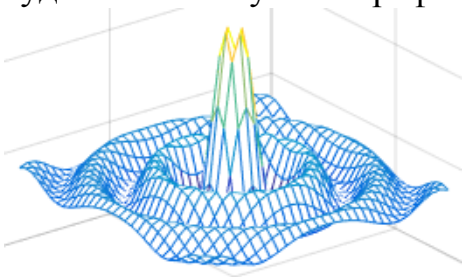
Створіть в текстовому редакторі Matlab наступну функцію.

```
function z = sindivr(x,y)
    rad=sqrt(x.^2+y.^2);
    % операція '.^' (крапка капелюшок) - це поелементне
    % піднесеня до стпеня
    z=sin(rad)./rad;
end
```

Збережіть функцію в файл з ім’ям `sindivr.m`. в той же каталог, в якому було створено функцію `testfunc.m`. (на цей каталог ви в Matlab вже встановили шлях доступу). Для перевірки працездатності функції в командному вікні Matlab виконайте наступний код

```
x=-15:1:15; y=-15:1:15; [X,Y]=meshgrid(x,y);
Z=sindivr(X,Y); mesh(X,Y,Z);
```

В Matlab повинен бути побудований наступний графік.



Тепер додайте шаблон функції `sindivr` у вузол **MATLAB 1**.

Global Definitions - MATLAB 1:

В новому рядку таблиці функцій введіть

Functions – **Function name** = `sindivr`

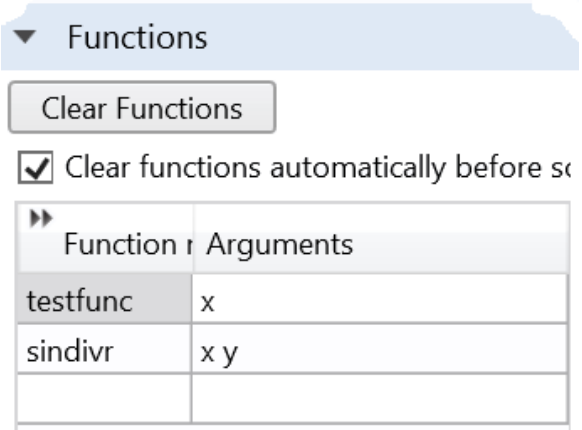
– **Arguments** = `x,y`

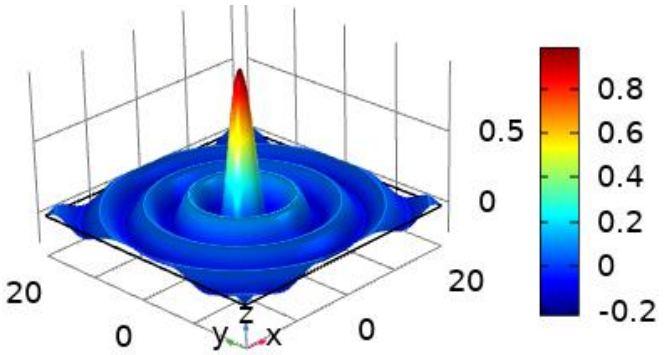
Для перевірки працездатності функції в розділі **Plot Parameters** введіть діапазони зміни аргументу і натисніть кнопку Plot.

Plot Parameters – **Low limit, Upper limit** = `-15,15`

– **Low limit, Upper limit** = `-15,15`

LMB> Plot



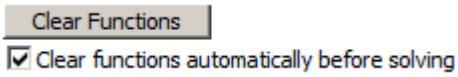


Якщо в розділі **Functions** в таблиці вказано кілька функцій, то будується графік тільки тієї, яка має однакову кількість аргументів (вони перелічені в стовпці **Arguments** через кому або проміжок) з кількістю діапазонів, заданих в розділі **Plot Parameters**. Якщо багато функцій мають однакову кількість аргументів, то будується графік першої з них. Використовуйте кнопки Move Up і Move Down для зміни порядку функцій в таблиці.

Якщо в ПВ вузла **MATLAB 1** натиснути кнопку Create Plot, то в гілці **Results** буде створена графічна група і в ній вузол з графіком функції. Він буде спиратися на автоматично створене в гілці **Data Sets** джерело даних типу `Grid`.

При створенні функцій в Matlab, призначених для використання в COMSOL, потрібно додержуватись деяких домовленостей. Функції можуть мати будь-яку кількість аргументів (зазвичай векторних), а повертати повинні один (вектор). Входові вектори можуть мати довільний, але однаковий розмір; вектор, що повертається, повинен мати такий же розмір. Наприклад, для функції двох змінних $f(x,y)$ в Matlab в якості x і y будуть передаватися вектори $\{x_i\}_{i=1}^n$ і $\{y_i\}_{i=1}^n$ (однакової довжини n), а елементи вектора результату $\{z_i\}_{i=1}^n$ будуть обчислюватися за формулою $z_i = f(x_i, y_i)$, і вектор z теж матиме довжину n.

Створену Matlab-функцію можна використовувати в будь-якому місці COMSOL моделі. Якщо з якоїсь причини ви внесли зміни в код Matlab функції, то в ПВ вузла **MATLAB 1** слід використати кнопку Clear Functions і потім виконати обчислення моделі повторно. Можна також встановити перемикач **Clear functions automatically before solving**. Тоді перед кожним обчисленням функція буде автоматично заново компілюватися.



Використання Matlab функції аналогічно використанню інших функцій.

6.5. Генерування випадкових даних.

Інколи в моделях потрібно працювати з випадковими даними. В COMSOL можливість їх створеання надають шаблон функції користувача `Random`, вбудовані функції `random` і `randomnormal`, а також інтерполяційні функції. Перед описом різних способів вибірки випадкових чисел на їх основі нагадаємо головні визначення.

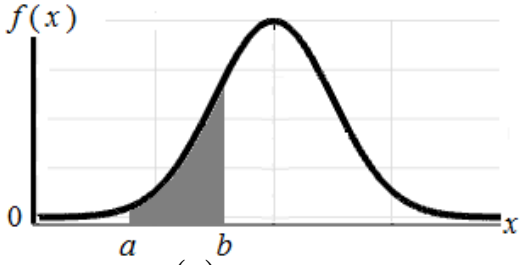
Якщо випадковій події поставлено у відповідність деяке число, то кажуть, що задана випадкова величина. Коли значення, які може набувати ця величина, є дискретним (скінченним або нескінченним) набором чисел $x_1, x_2, \ldots, x_n, \ldots$, то і сама випадкова величина називається дискретною. Якщо ж значення, які вона може приймати, заповнюють скінченний або нескінченний проміжок (a, b) числової осі, то така випадкова величина називається неперервною.

Кожному значенню x_n випадкової величини дискретного типу відповідає певна ймовірність p_n; кожному проміжку (a, b) з області значень неперервної випадкової величини відповідає певна ймовірність $P(a < X < b)$ того, що значення X потрапляють в цей проміжок.

Співвідношення, що встановлює тим чи іншим чином зв'язок між можливими значеннями випадкової величини і їх ймовірностями, називається законом розподілу випадкової величини. Ймовірність $P(a < X < b)$ того, що значення X неперервної випадкової величини потрапляє в проміжок (a, b) обчислюється з використанням функції густини (щільності) ймовірності $f(x)$ (probability distribution function — PDF) за формулою

$$P(a < X < b) = \int_a^b f(x)\, d\, x.$$

Графік функції $f(x)$ називається кривою розподілу. Геометрично ймовірність потрапляння випадкової величини в проміжок (a, b) дорівнює площі криволінійної трапеції, обмеженої зверху кривою розподілу, знизу - віссю Оx, зліва і справа – прямими $x = a, x = b$.



Функція щільності ймовірності $f(x)$ має наступні властивості:

1^0. $f(x) \geq 0$;

2^0. $\int\limits_{-\infty}^{\infty} f(x)dx = 1$.

Функція $F(x) = P(X < x)$ (ймовірність того, що значення X випадкової змінної не перевищують число x) називається функцією розподілу ймовірності випадкової величини (cumulative distribution function – CDF). Вона пов'язана з функцією щільності ймовірності $f(x)$ співвідношенням $F(x) = \int\limits_{-\infty}^{x} f(\xi)d\xi$.

Зазначимо основні властивості функції розподілу ймовірності:

1^0. $F(x)$ – неспадна функція.

2^0. $F(-\infty) = 0$,

3^0. $F(+\infty) = 1$.

Математичне сподівання (математичне очікування) неперервної випадкової величини визначається за формулою

$$M(X) = \int\limits_{-\infty}^{+\infty} x\, f(x)d\,x$$

де $f(x)$ - щільність ймовірності випадкової величини.

Дисперсією називають математичне сподівання квадрата відхилення випадкової величини від її математичного очікування

$$D(X) = M\left([X - M(X)]^2\right).$$

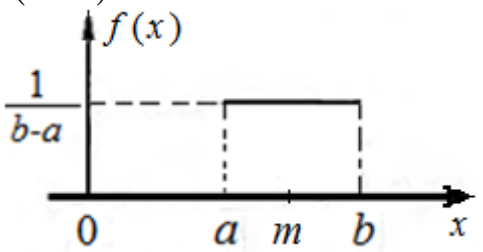
Якщо ввести позначення $M(X) = m$, то формула для дисперсії неперервної випадкової величини X матиме вигляд.

$$D(X) = \int\limits_{-\infty}^{+\infty} (x - m)^2\, f(x)d\,x$$

Середнім квадратичним відхиленням випадкової величини X називається величина $\sigma_x = \sqrt{D(X)}$. Середнє квадратичне відхилення є мірою розсіювання значень випадкової величини навколо її математичного очікування.

Головною відміною випадкових змінних одна від одної є їх функції густини розподілу ймовірності.

Рівномірним називається розподіл такої випадкової величини, всі значення якої лежать на відрізку $[a,b]$ і мають сталу щільність ймовірності $1/(b-a)$ на цьому відрізку. Тоді середнє значення дорівнює $m = (a+b)/2$, а дисперсія $D = \sigma^2 = (b-a)^2/12$.

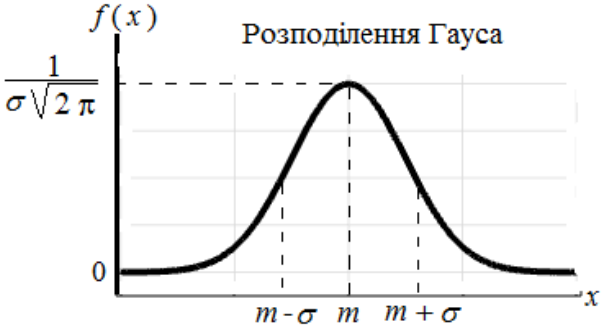


$$f(x) = \begin{cases} 0 & , x < a \\ \dfrac{1}{b-a} & , a \leq x \leq b \\ 0 & , x > b. \end{cases}$$

Нормальним називається розподіл випадкової величини, який характеризується щільністю

$$f(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{-(x-m)^2/2\sigma^2}. \qquad (1)$$

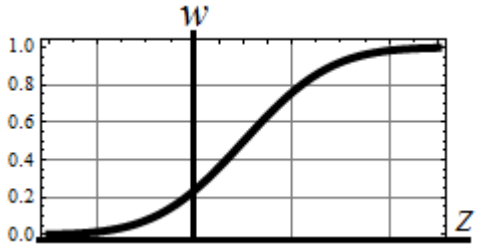
Графік цієї функції показаний на наступному рисунку. Він симетричний відносно прямої $x=m$, максимум досягається при $x = m$ і дорівнює $1/\left(\sigma\sqrt{2\pi}\right)$.



Оскільки $\int_{-\infty}^{\infty} x\, f(x)\, d\,x = m$, то параметр m є математичним очікуванням випадкової величини. Також $\int_{-\infty}^{\infty}(x-m)^2\, f(x)\, d\,x=\sigma^2$. Тому $D(x)=\sigma^2$, тобто σ є середнім квадратичним відхиленням величини X. Коли щільність розподілу випадкової величини описується цією функцією, то приблизно 70% її значень потрапляють в діапазон $[m-\sigma,\, m+\sigma]$, і приблизно 95% - в діапазон $[m-2\sigma,\, m+2\sigma]$.

COMSOL Multiphysics «вміє» генерувати випадкові величини лише з рівномірною і нормальною густиною розподілу ймовірності. Але іноді може знадобитися генерування випадкової величини з іншими законами розподілу. Тоді можна скористатися наступними міркуваннями.

Нехай є випадкова величина w з рівномірним розподілом на відрізку [0, 1]. Для побудови випадкової величини z з заданою функцією розподілу $F(z)$ треба розв'язати рівняння $F(z)=w$ відносно z. Це пояснює наступний рисунок



Тобто випадкову величину z треба побудувати за формулою

$$z=F^{-1}(w), \qquad (2)$$

де w випадкова величина з рівномірним розподіленням на відрізку [0, 1].

Перед застосуванням випадкових чисел потрібно розуміти алгоритми їх генерування. Випадкова величина - це величина, яка набуває в результаті експерименту одне з багатьох значень, причому поява того чи іншого значення до його вимірювання не можна передбачити. Комп'ютерні алгоритми не вміють генерувати справжні випадкові числа. Дійсно, точно повторивши алгоритм генератора випадкових чисел, можна отримати таке ж точно число, що і в

попередньому експерименті. Отже комп'ютерні алгоритми генерують лише псевдовипадкові числа. Псевдовипадкове число - це випадкове число, однозначно генероване з початкового значення (seed). Результат генерування буде відрізнятися при різних початкових значеннях, хоча одне і те ж початкове значення дасть при повторенні однаковий результат. Наприклад, якщо обчислити вбудовану функцію `random` з якимось початковим значенням, скажімо `random(1)`, то результат не покаже зв'язку з числом 1 (в зв'язку з цим число буде виглядати випадковим), однак значення залишиться без змін після кількох повторних обчислень (звідки випливає його псевдовипадковість). Отже псевдовипадкові числа відтворювані, оскільки можна отримати однаковий результат, виконавши моделювання кілька разів з одним і тим же початковим значенням.

Інколи для великої кількості об'єктів (наприклад частинок в пучку) потрібно генерувати для кожного з них кілька випадкових чисел (координат, швидкостей, зовнішніх сил, тощо). Тоді в функціях генераторах випадкових чисел корисними будуть додаткові аргументи. Наприклад, першим буде індекс частинки, а інші задаватимуться фіксовано. Виходячи з наведених міркувань, у вбудованих в COMSOL рандомних функцій можна застосовувати не один, а кілька аргументів, які використовуються в якості початкових значень для алгоритмів генерування псевдовипадкових чисел.

В COMSOL Multiphysics є кілька способів задати псевдовипадкові числа. Одним з них є `Random` функція користувача, яку можна створювати у вузлах **Global Definitions** або **Definitions** компонента. Тоді в ПВ створюваного вузла **Random 1** в списку **Distribution** потрібно вказати тип функції густини розподілу ймовірності (`Uniform` або `Normal`). Для обох випадкових функцій треба вказати середнє значення (Mean) і кількість аргументів (від 1 до 3).

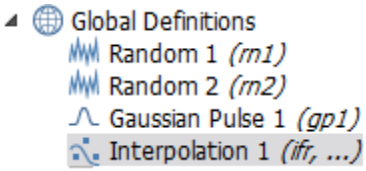
Якщо використовується рівномірний розподіл (Uniform), то потрібно додатково задати параметр Range (діапазон). Тоді для середнього значення $m=(a+b)/2$ і діапазону $\sigma=b-a$ функція густини розподілу ймовірностей виглядатиме наступним чином:

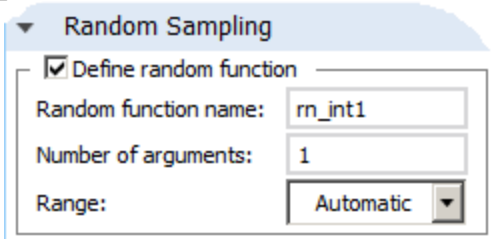
$$f(x)=\begin{cases} 0 & , x\le m-\sigma/2 \\ 1/\sigma, & m-\sigma/2<x<m+\sigma/2 \\ 0 & , x\ge m+\sigma/2 \end{cases}.$$

Якщо використовується нормальний розподіл (Normal, розподіл Гауса), вкажіть значення параметрів **Mean** (середнє) і **Standard Deviation** (середнє квадратичне відхилення). Для середнього значення m і середньоквадратичного відхилення σ функція густини розподілу ймовірностей визначатиметься за формулою (1).

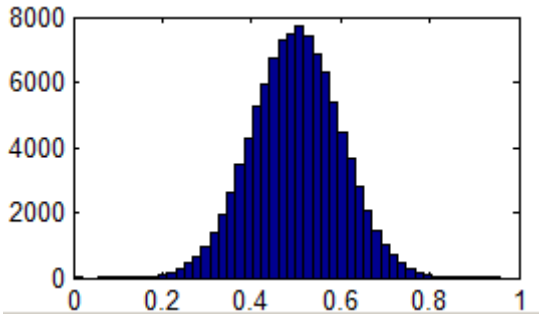
Як альтернативу функції користувача Random можна застосовувати вбудовані функції `random(x,y)` і `randomnormal(x,y)` з одним або двома аргументами. В `random` реалізовано рівномірний розподіл із середнім значенням 0 і діапазоном 1; в `randomnormal` - нормальний розподіл із середнім значенням 0 і середньоквадратичним відхиленням 1.

Ще один спосіб генерування псевдовипадкових величин полягає в використанні інтерполяційних функцій з увімкненою опцією **Define random function** (визначити випадкову функцію). Це автоматично створює випадкову функцію на її основі з вибіркою, яка має цей розподіл.





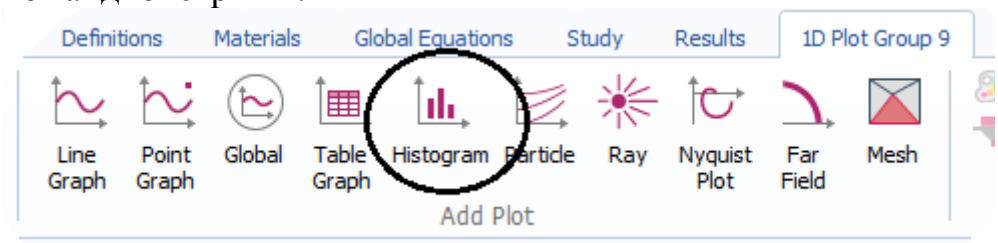
При зображенні випадкових величин часто застосовуються гістограми. Так називають спосіб зображення табличних даних у вигляді стовпчастої діаграми, яка будується наступним чином. Спочатку множину значень розбивають на кілька інтервалів (кошиків, bins), які відкладають на горизонтальній осі. Потім над кожним з них креслиться прямокутник висотою (або площею), що дорівнює кількості табличних значень, які потрапляють в цей інтервал. Замість кількості може відкладатися частота появи значень у відповідному кошику (кількість значень в кошику поділена на загальну кількість елементів). Якщо інтервали мають різну ширину, то висоту прямокутників обирають так, щоб площі стовпчиків були пропорційні частоті потрапляння значень в кошик. Іноді гістограму називають частотним розподіленням. На наступному рисунку наведена гістограма 100000 випадкових значень, розподілених за нормальним законом з середнім значенням m=0.5 і середньоквадратичним відхиленням $\sigma = 0.1$, розподілених по 50 кошикам однакової ширини.



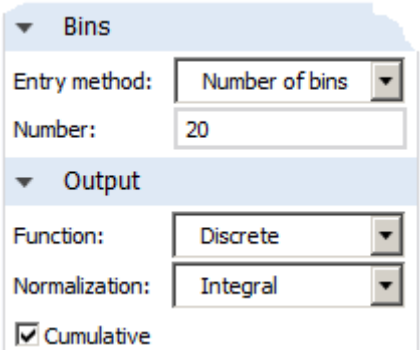
Можна будувати гістограми (стовпчасті діаграми) двовимірних даних.

В COMSOL ще використовуються накопичувальні (cumulative) гістограми. Вони будуються так, що висота кожного прямокутника над інтервалом (кошиком) пропорційна кількості (або частоті) всіх значень, які потрапляють в цей і всі кошики з меншими значеннями.

Для побудови одновимірної гістограми в COMSOL призначений графічний вузол `Histogram`, який додається в одновимірну графічну групу відповідною кнопкою командної стрічки.



В ПВ створюваного вузла в полі **Expression** вводиться вираз, наприклад, `rn1(x)`, де `rn1` – ім'я функції, а `x` – координати вздовж одновимірного ДД. Функція обчислюватиметься в кожному вузлі ДД і повертатиме таку ж кількість значень, утворюючи набір даних для гістограми. В інших полях ПВ цього вузла задається спосіб побудови кошиків (їх кількість або границі), та налаштовується тип гістограми (стовпчикова чи неперервна ламана).



Конкретніше налаштування `Histogram` вузлів ми обговоримо в прикладах.

Приклад. Створення і використання випадкових величин в COMSOL Multiphysics.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = `None`

Рівномірне розподілення. Створіть `Random` функцію користувача з рівномірним розподіленням.

Global Definitions : RMB > Functions – Random

Function name = `rn1`

`Parameters` – **Number of arguments** =`1`

– **Distribution** = `Uniform`

– **Mean, Range** = `0.5,1`

Для побудови графіка цієї функції нам знадобиться одновимірне джерело даних `Grid 1D`.

`Results` -> Data Set - More Data Sets -> Grid 1D

Results - Data Sets - Grid 1D 1 :

`Data` – **Source, Function** = `Function, All`

`Paramemter Bounds` – **Minimum, Maximum** = `0,1`

Тут діапазон [0,1] обирається з міркувань, що він співпадатиме з діапазоном значень функції `rn1(x)` ($0.5 \pm 0.5 = [0,1]$).

В розділі **Resolution** вказуються кількості точок дискретизації. Зазвичай графік будується в цих точках. Для наочності візьміть невелику кількість точок, наприклад, 11.

`Resolution` – **Resolution** = `11`

Створіть похідне джерело даних `Cut Point 1D` з точками, розташованими у вузлах **Grid 1D 1**.

`Results` -> Data Set - More Data Sets -> Cut Point 1D

Data Sets - Cut Point 1D 1 :

Data – **Data set** = Grid 1D 1

Point Data – **Entry method, x** = Coordanates, range(0,0.1,1)

Побудуйте графіки (лінійний і точковий) функції rn1(x) «над» цими джерелами даних. Для лінійного графіка значення функції будуть обчислюватися лише у вузлах дискретизації. Оскільки точок мало, то ми спостерігатимемо ламану.

Results **-> Plot Group ->** 1D Plot Group

1D Plot Group 1:

Data – **Data set** = Grid 1D 1

1D Plot Group 1-> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

y-Axis Data – **Expression** = rn1(x)

x-Axis Data – **Parameter, Expression** = Expression,x

Coloring and Style –**Width** = 3

В тих же вузлах побудуйте точки.

1D Plot Group 1 -> Add Plot - Point Graph

1D Plot Group 1 - Point Graph 1:

Data Set – **Data** = Cut Point 1D 1

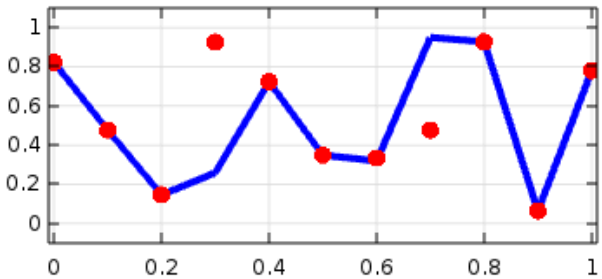
y-Axis Data – **Expression** = rn1(x)

x-Axis Data – **Parameter, Expression** = Expression,x

Задайте колір і розмір точок.

Coloring and Style – **Color, Width** = Red,4

LMB > Plot



Хоча обчислення випадкової функції rn1(x) виконувалося на одній і тій же множині точок (на **Grid 1D 1** – 11 рівномірно розташованих точок на відрізку [0,1] включно з кінцями, і в точках з координатами range(0,0.1,1), утворюючих ДД **Cut Point 1D 1**) похибка в обчислені абсцис 4-ї і 8-ї точок призвели в них до суттєвої різниці результату. Координати інших вузлів співпали точно (в пам'яті комп'ютера всі двійкові розряди в представлені чисел співпали), що призвело до генерування однакових псевдовипадкових чисел .

Створіть одновимірне ДД з густішим розташуванням точок дискретизації, наприклад, 100 точок на відрізку [0,1].

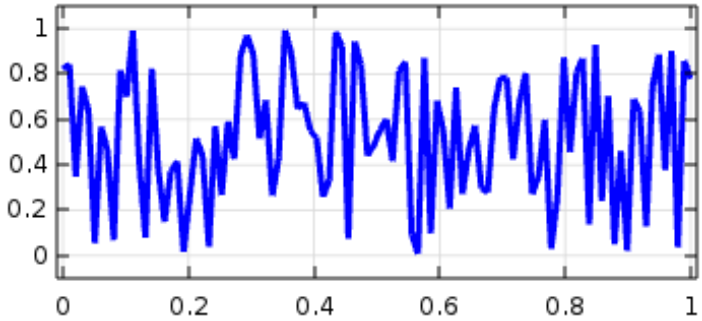
Data Sets - Grid 1D 1 : RMB > Duplicate

Data Sets - Grid 1D 2 :

Resolution – **Resolution** = 100

Побудуйте на ньому графік функції rn1(x).

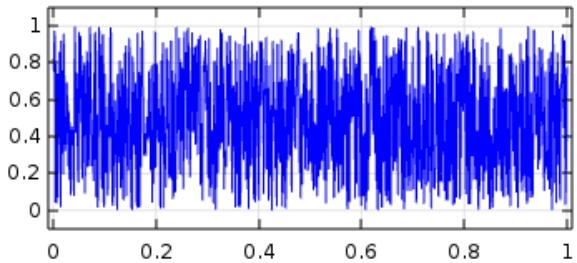
1D Plot Group 1: RMB > Duplicate
1D Plot Group 2:
Data – **Data set** = Grid 1D 2
1D Plot Group 1 - Point Graph 1: RMB > Delete
LMB > Plot

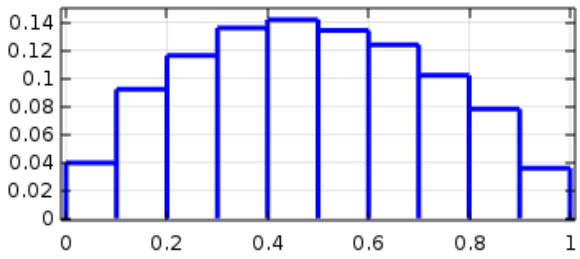


Функція `rn1(x)` використовує аргумент *x* – координати вузлів (100 рівномірно розташованих значень на відрізку [0,1] включно з кінцями) в якості початкових значень і повертає псевдовипадкові значення, які відхиляються від середнього `m=0.5` не більше ніж на ±0.5. На рисунку з 11 точками це було не так помітно внаслідок невеликої кількості точок.

Побудуйте гістограму множини значень функції `rn1(x)`. Гістограма буде точніше відповідати поведінці випадкової величини, якщо точок буде більше. Створіть ще один `Grid 1D` вузол з 1000 точок дискретизації.

Data Sets - Grid 1D 1 : RMB > Duplicate
Data Sets - Grid 1D 3 :
Resolution – **Resolution** = 1000
1D Plot Group 2: RMB > Duplicate
1D Plot Group 3:
Data – **Data set** = Grid 1D 3
1D Plot Group 1 - Line Graph 1:
Coloring and Style –**Width** = 1
LMB > Plot (*наступний рисунок ліворуч*)

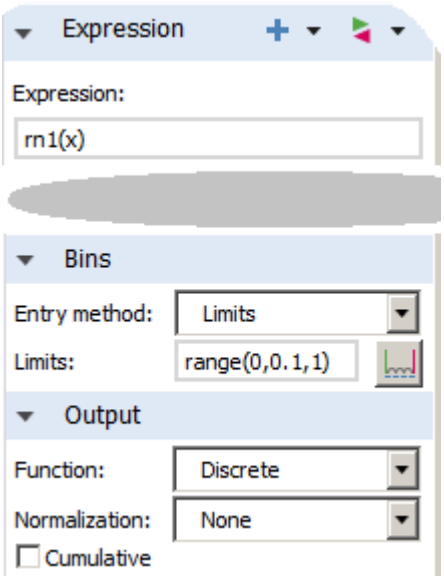




`Results` **-> Plot Group ->** 1D Plot Group
`1D Plot Group 4`-> Add Plot – Histogram

1D Plot Group 4 – Histogram 1:

`Data` – **Data set** = `Grid 1D 3`
`Expression` – **Expression** = `rn1(x)`
`Bins` – **Entry method** = `Limits`
– **Limits** = `range(0,0.1,1)`
`Output` – **Function** = `Discrete`
LMB > Plot (*попередній рисунок праворуч*)

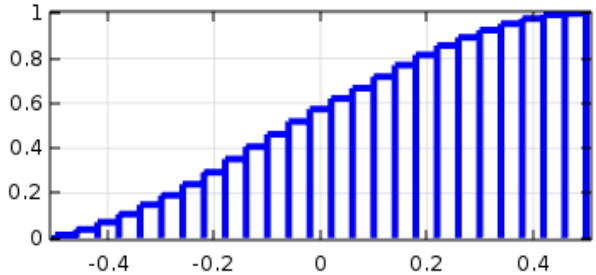


Тут для аргументу `x` з ДД **Grid 1D 3** значення функції `rn1(x)` розкидається по відрізку [0,1], який відкладається на горизонтальній осі гістограми. Тому межі кошиків `range(0,0.1,1)` розташовані на цьому відрізку [0,1].

При рівномірному розподілі гістограмні стовпчики повинні були б бути приблизно однакової висоти. Як бачимо, рівномірності немає. Але якщо встановити прапорець **Cumulative**=`True` (тобто будувати накопичувальну гістограму), то відмінність від лінійного розподілу не так помітна.

1D Plot Group 3 – Histogram 1:

`Bins` – **Entry method** = `Number of bins`
– **Number** = `25`
`Output` – **Cumulative** = `True`



Зауваження. В COMSOL 5 всіх версій (від 5.1 до 5.5) не дуже коректно кресляться гістограми, а функції генерування псевдовипадкових чисел працюють як треба. Це можна перевірити, створивши в COMSOL MATLAB - функцію-оболонку функції `rand` Matlab, до якої питань немає. Використавши її для генерування псевдовипадкових чисел, і побудувавши гістограму в COMSOL, отримаємо результат несхожий з гістограмою, побудованою в Matlab. Отже проблема полягає в обчисленні гістограм в COMSOL, а не в генеруванні випадкових чисел.

Нормальне розподілення і функція Гауса (Gaussian Pulse). Для випадкової функції користувача з нормальним розподіленням ймовірності налаштування вузла властивостей виконується так само як і для рівномірного розподілення.

Для тренування побудуємо на площині в околі нуля множину випадкових точок, координати яких розподілені за нормальним законом.

Global Definitions : RMB > Functions – Random

Function name = rn2

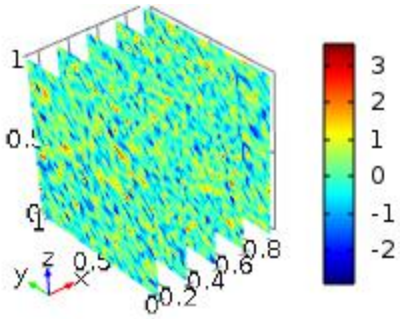
Тут ми створимо випадкову функцію трьох змінних з нормальним законом розподілення. Перші два аргументи будуть використані в якості початкових значень, куди підставлятимуться координати прямокутної сітки точок, а третій - для розрізнення псевдовипадкових даних.

Parameters – **Number of arguments** =3

– **Distribution** = Normal

– **Mean, Standard deviation** = 0,1

LMB > Plot



Для побудови множини точок на площині використаємо графічний вузол Scatter Surface. Його графік будується наступним чином. В ПВ вузла в розділі **Evaluation Points** задаються координати точок, утворюючих сітку. В розділі **Expression** в полях **x component** і **y component** задаються вирази, які обчислюються по координатам точок. Значення цих виразів утворюють нову сітку точок, яка і зображується.

Але спочатку нам знадобиться двовимірне джерело даних.

Results -> Data Set - More Data Sets -> Grid 2D

Results - Data Sets - Grid 2D 4 :

Data – **Source, Function** = Function, All

Paramemter Bounds – **First parameter** –

– **Name, Minimum, Maximum** = x,-3,3

Paramemter Bounds – **Second parameter** –

– **Name, Minimum, Maximum** = y,-3,3

Тепер можна будувати Scatter Surface графік.

Results -> Plot Group - 2D Plot Group

2D Plot Group 5 :

Data – **Data set** = Grid 2D 4

Plot Settings – **Plot data set edges** = False

Plot Group 5 **- Add Plot- >** More Plots - Scatter Surface

2D Plot Group 5 - Scatter Surface 1 :

Expression – **x component** = rn2(x,y,1)

– **y component** = rn2(x,y,2)

Evaluation Points – **x grid points – Entry method** =Coordinates

– **x grid points - Coordinates** = range(-3,0.25,3)

Evaluation Points – **y grid points – Entry method** =Coordinates

– **y grid points - Coordinates** = `range(-3,0.25,3)`

В розділі **Radius** в полі **Expression** можна ввести функцію, яка буде визначати радіуси/розміри точок.

Radius – **Expression** =`0.05`

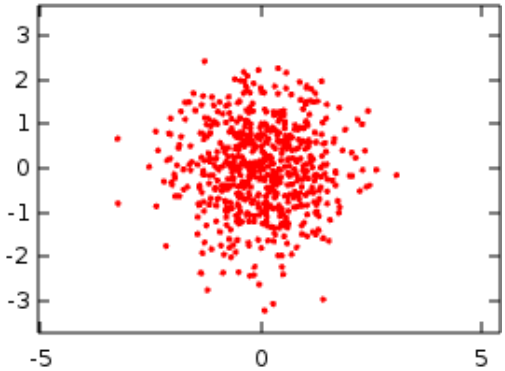
– **Radius scale factor** = `True, 1`

Поле **Color** – **Expression** використовується при фарбуванні точок.

Color – **Expression** = `0`

Coloring and Style – **Coloring, Color** = `Uniform, Red`

LMB > Plot



Ще раз зауважимо, що в полях **x component** і **y component** застосована функція `rn2(x,y,N)`, де в якості перших двох початкових значень використовуються координати точок сітки (узяті з розділу **Evaluation Points**), а третє початкове значення призначене для того, щоб для абсцис і ординат генерувалися різні послідовності псевдовипадкових величин.

Тепер побудуйте двовимірну гістограму.

`Results` -> Plot Group - 2D Plot Group

2D Plot Group 6:

Plot Settings – **Plot data set edges** = `False`

`Plot Group 6` **– Add Plot ->** More Plots - Histogram

2D Plot Group 6 – Histogram 1:

Data – **Data set** = `Grid 2D 4`

x-Expression – **Expression** = `rn2(x,y,1)`

y-Expression – **Expression** = `rn2(x,y,2)`

Bins – **x bins** – **Entry method, Number** = `Number of bins,20`

– **y bins** – **Entry method, Number** = `Number of bins,20`

Output – **Function** = `Discrete`

Output – **Normalization** = `Integral`

RMB > Height Expression

Histogram 1 – Height Expression 1 : Axis – **Scale factor** = `True, 100`

LMB > Plot (*наступний рисунок ліворуч*)

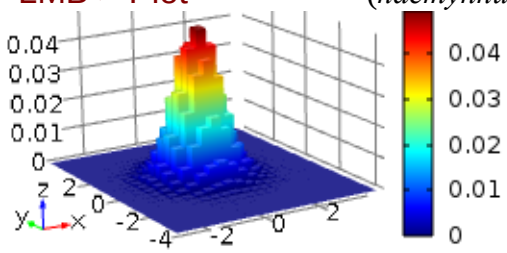
Якщо змінити «нормування» гістограми в цьому вузлі, то отримаємо графік, зображений праворуч.

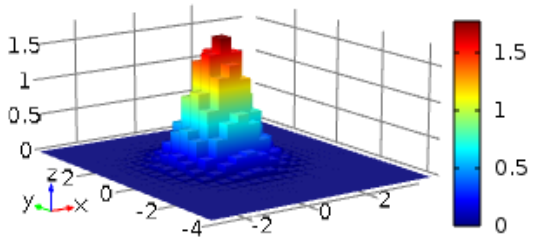
2D Plot Group 6 – Histogram 1:

Output – **Normalization** = `None`

Histogram 1 – Height Expression 1 : Axis – **Scale factor** = `True, 2`

LMB > Plot (*наступний рисунок праворуч*)





При побудові 2D гістограми відрізки координатних осей X і Y розбиваються на інтервали. Це розбиття утворює прямокутні кошики, а зображувана поверхня показує відносну кількість елементів/значень, які потрапили в такі кошики. Поверхня зображується у вигляді стовпчиків, якщо у вузлі гістограми значення опції **`Output`** – **Function** обрано `Discrete`, і як неперервна поверхня - якщо `Continuous`. Коли використана опція **Normalization** =`Integral`, то кожний стовпчик показує його відносний розмір/об'єм.

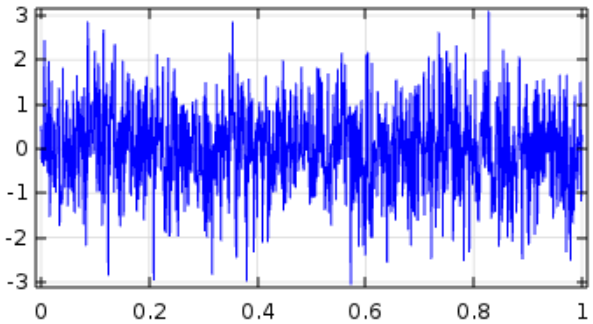
Порівняємо одновимірну гістограму випадкової величини, яка має нормальне розподілення, з графіком функції густини нормального розподілення. Для різноманітності використаємо вбудовану функцію `randomnormal(x)`. Вона генерує нормально розподілену випадкову величину з нульовим середнім і одиничним середньоквадратичним відхиленням. Спочатку побудуйте графік випадкової величини.

1D Plot Group 3: RMB > Duplicate

2D Plot Group 7 – Line Graph 1:

`Expression` – **Expression** = `randomnormal(x)`

LMB > Plot



В новій гілці створіть її гістограму.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 8` **– Add Plot ->** Histogram

1D Plot Group 8 – Histogram 1:

`Data` – **Data set** = `Grid 1D 3`

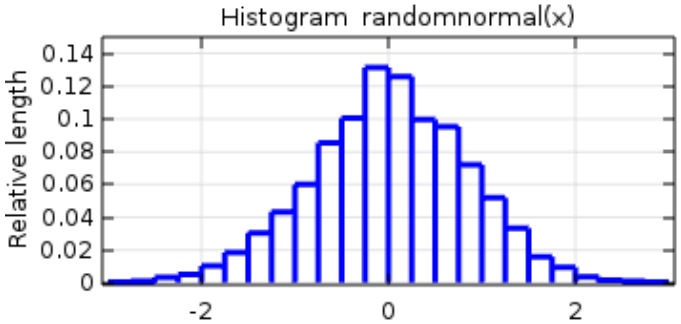
`Expression` – **Expression** = `randomnormal(x)`

`Bins` – **Entry method, Limits** = `Limits,range(-3,0.25,3)`

`Output` – **Function** = `Discrete`

– **Normalization** = `Integral`

LMB > Plot



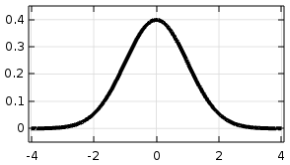
Тепер побудуйте графік функції Гауса. Її можна було б створити як аналітичну, обчислювану за формулою (1). Але в COMSOL для цієї функції є шаблон `Gaussian Pulse`.

Global Definitions: RMB > Functions – Gaussian Pulse

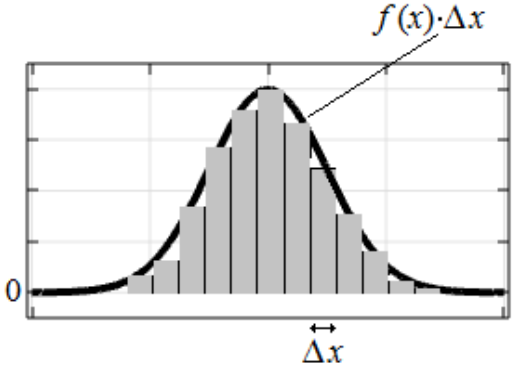
Function name = `gp1`

`Parameters` – **Location, Standard deviation** = `0,1`

LMB > Plot



Щоб порівняти графік функції Гауса і гістограму, потрібно їх привести до спільного масштабу. На стовпчастій гістограмі при встановленій опції **Normalization**=`Integral` площа прямокутників пропорційна відносній кількості значень випадкової величини, які потрапили в відповідний інтервал. Для графіка функції $f(x)\cdot\Delta x$, де $\Delta x = x_i - x_{i-1}$ - ширина кошиків гістограми, висота стовпчиків під кривою над інтервалом $[x_{i-1}, x_i]$ буде приблизно дорівнювати $f((x_{i-1}+x_i)/2)\cdot\Delta x$ (якщо всі стовпчики однакової ширини), тобто висоті стовпчиків гістограми. Тому графік функції $f(x)\cdot\Delta x$ (а не графік $f(x)$) буде приблизно відтворювати форму гістограми.



Для побудови графіка функції Гауса нам знадобиться одновимірне джерело даних `Grid 1D` з діапазоном `[-3,3]`, що приблизно співпадає з множиною значень функції `randomnormal(x)`, бо на гістограмі вздовж горизонтальної осі відкладається саме вона.

`Results` -> Data Set - More Data Sets -> Grid 1D

Results - Data Sets - Grid 1D 5 :

`Data` – **Source, Function** = `Function, All`

`Paramemter Bounds` – **Minimum, Maximum** = `-3,3`

В гілку, де побудована гістограма, додайте графік створеної функції Гауса з множником 0.25 – шириною стовпчиків гістограми.

`1D Plot Group 8`-> Add Plot – Line Graph

1D Plot Group 8 - Line Graph 1:

Data – **Data set** = `Grid 1D 5`

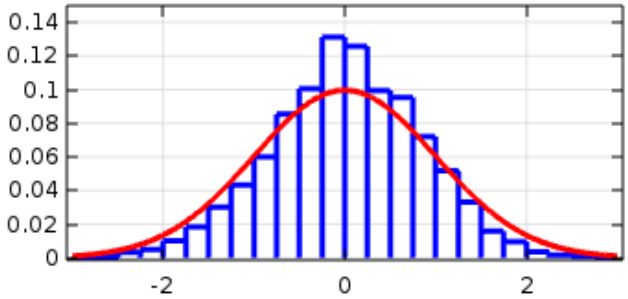
y-Axis Data – **Expression** = `gp1(x)*0.25`

Множник `0.25` – це ширина кошиків на гістограмі.

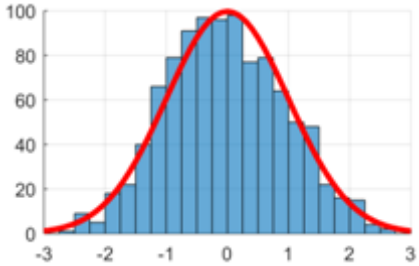
x-Axis Data – **Parameter, Expression** = `Expression,x`

Coloring and Style – **Width, Color** = `3,Red`

LMB > Plot



Розходження гістограми і кривої, вірогідно, пов'язані з проблемами побудови гістограм в COMSOL. На наступному рисунку зображені ті ж самі графіки, але створені в Matlab.



Звичайно, що масив випадкових чисел, згенерований в Matlab, відрізняється від масиву COMSOL. Але при багатократному застосуванні в Matlab наступного сценарію, розбіжностей, таких як в COMSOL, не спостерігається.

```
%Створити графчне вікно з білим фоном
figure('Color','w'); hold on; grid on;
% Гістограма нормально розподіленої випадкової величини
% 1000 значень
X=randn(1,1000);
edges=[-3:0.25:3];    % межі кошиків
histogram(X,edges);
% Графік функції Гауса
x=-3:0.1:3;
y=1000*0.25/sqrt(2*pi)*exp(-x.^2/2);
plot(x,y,'r','LineWidth',3);
```

Єдина різниця гістограм полягає в тому, що в Matlab стовпчики відображають кількість елементів в кошику (а не відносну кількість). У зв'язку з чим при побудові кривої ми додатково використали множник 1000 (загальну кількість згенерованих випадкових чисел).

Розподілення Релея. Кажуть, що випадкова величина розподілена за законом Релея, якщо її густина обчислюється за формулою

$$f(z)=\begin{cases} \dfrac{x}{\sigma^2}e^{-x^2/2\sigma^2}, & z\ge 0 \\ 0, & z<0 \end{cases}. \qquad (3)$$

Математичне очікування і дисперсія дорівнюють

$$\mu=\sigma\sqrt{\frac{\pi}{2}} \quad \text{і} \quad D(X)=\frac{4-\pi}{4}\sigma^2.$$

Функція розподілення має вид

$$F(x)=P(X\le x)=\int_{-\infty}^{x} f(x)dx=\begin{cases} 0, & x\le 0 \\ 1-e^{-x^2/2\sigma^2}, & x>0 \end{cases}.$$

Щоб отримати вираз випадкової величини z з релеєвською щільністю ймовірності через рівномірно розподілену випадкову величину w, треба розв'язати рівняння $F(z)=w$, тобто $1-e^{-x^2/2\sigma^2}=w$ (див. ф. (2)). Звідки $z=\sigma\sqrt{-2\ln(1-w)}$. Оскільки w рівномірно розподілена на відрізку [0,1], то $1-w$ теж матиме таке розподілення, і $1-w$ можна замінити на w. Таким чином, приходимо до формули

$$z=\sigma\sqrt{-2\ln w}. \qquad (4)$$

Тобто випадкова величина z, яка генеруватиметься за формулою (4), де w – рівномірно розподілена на відрізку [0,1], матиме розподілення Релея.

Побудуйте графік випадкової величини за формулою (4) при $\sigma=3$. Спочатку створіть відповідний параметр моделі.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `sgm, 3`

Тепер можна будувати графік.

`Results` **`->`** **`Plot Group`** **`->`** 1D Plot Group

1D Plot Group 9:

`Data` – **Data set** = `Grid 1D 3`

`1D Plot Group 9`-> Add Plot – Line Graph

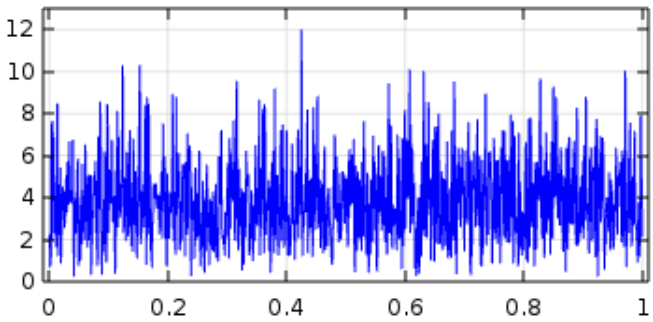
1D Plot Group 9 - Line Graph 1:

`y-Axis Data` – **Expression** = `sgm*sqrt(-2*log(0.5+random(x)))`

Тут функція `random(x)` генерує рівномірно розподілену випадкову величину з середнім 0 і розкидом 1, тобто зі значеннями в діапазоні `[-0.5,0.5]`. Вираз `0.5+random(x)` вже створює випадкову величину зі значеннями в діапазоні `[0,1]`.

`x-Axis Data` – **Parameter, Expression** = `Expression,x`

LMB > Plot



Побудуйте гістограму цього набору даних.

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 10` **– Add Plot ->** Histogram

1D Plot Group 10 – Histogram 1:

`Data` – **Data set** = `Grid 1D 3` (*цим задається діапазон зміни аргументу* x)

`Expression` – **Expression** = `sgm*sqrt(-2*log(0.5+random(x)))`

З графіка випадкової величини можна зрозуміти, що розкид значень приблизно розташований на відрізку [0,10]. Тому ми його оберемо для задання діапазону кошиків.

`Bins` – **Entry method, Limits** = `Limits, range(0,0.25,10)`

`Output` – **Function** = `Discrete`

– **Normalization** = `Integral`

LMB > Plot

В цій же групі побудуйте графік густини (3) розподілу «релеєвської» випадкової величини з урахуванням множника `0.25` – ширини кошиків гістограми. Для цього нам знадобиться `Grid 1D` джерело даних з діапазоном [0,10].

`Results` -> Data Set - More Data Sets **->** Grid 1D

Results - Data Sets - Grid 1D 6 :

`Data` – **Source, Function** = `Function, All`

`Paramemter Bounds` – **Minimum, Maximum** = `0,10`

`1D Plot Group 10`-> Add Plot – Line Graph

1D Plot Group 10 - Line Graph 1:

`Data` – **Data set** = `Grid 1D 6`

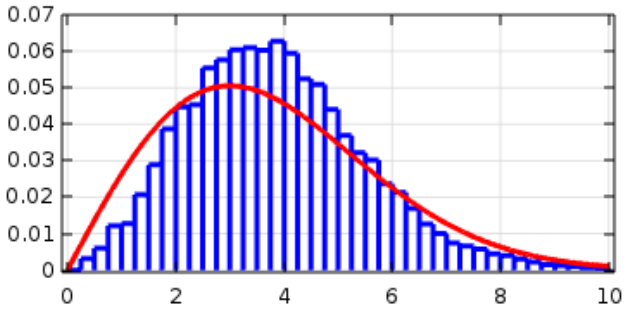
`y-Axis Data` – **Expression** = `x/sgm^2*exp(-x^2/(2*sgm^2))*0.25`

Множник `0.25` – це ширина кошиків на гістограмі.

`x-Axis Data` – **Parameter, Expression** = `Expression,x`

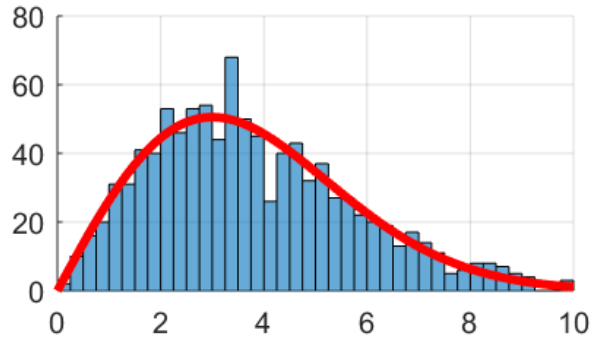
`Coloring and Style` –**Width, Color** = `3,Red`

LMB > Plot



Тут знову бачимо не зовсім коректне створення гістограм в COMSOL Multiphysics 5.1. Для порівняння наведемо сценарій Matlab, який виконує ті ж самі побудови.

```
% Вікно з білим фоном
figure('Color','w');hold on; grid on;
% Гістограма випадкової величини (1000 значень)
X=rand(1,1000);
Z=3*sqrt(-2*log(X));
edges=[0:0.25:10];        % межі кошиків
histogram(Z,edges);
% Графік кривої Релея
x=-0:0.1:10; sgm=3;
y=1000*0.25/sgm^2*x.*exp(-x.^2/(2*sgm^2));
plot(x,y,'r','LineWidth',3);
```



Збережіть COMSOL файл, але не закривайте його.

З випадковими числами тісно поєднані методи Монте-Карло – група методів, призначених для дослідження випадкових процесів. Не вдаючись в подробиці, розглянемо два простих приклади.

Обчислення числа π. Вписавши коло в квадрат, можна виразити відношення площі кола до площі квадрата наступним чином $\frac{S_{circle}}{S_{square}} = \frac{\pi r^2}{(2r)^2} = \frac{\pi}{4}$. Якщо заповнити квадрат точками з випадковими координатами, то відносна кількість точок, які потрапили в коло, буде пропорційна відношенню зазначених площ. Тоді $\pi = 4 \cdot S_{circle} / S_{square} \approx 4 \cdot N_{circle} / N_{square}$, де N_{circle} - кількість точок, що потрапили в коло, N_{square} - кількість точок в квадраті (або загальна кількість точок, якщо всі вони потрапляють в квадрат).

Реалізуємо описані обчислення в COMSOL, узявши коло $x^2 + y^2 \le 0.5^2$ і квадрат $-0.5 \le x \le 0.5$, $-0.5 \le y \le 0.5$. Відкрийте останній файл. Обчислення можна виконати в гілці **Derived Values** у вузлі типу `Point Evaluation`. Оскільки йому потрібно джерело даних, то створіть його з однієї точки.

`Results` -> Data Set - More Data Sets -> Cut Point 1D

Data Sets - Cut Point 1D 2 :

`Data` – **Data set** = `Grid 1D 1`

`Point Data` – **Entry method, x** = `Coordanates, 0`

Тепер створіть вузол `Point Evaluation`.

Derived Values: RMB > Point Evaluation

Point Evaluation 1:

Data – **Data set** = `Cut Point 1D 2`

Expression – **Expression** = `4*sum(if(random(i,5)^2+ random(i,6)^2<0.5^2,1,0),i,1,10000)/10000`

Функції `random(i,N)` генерують випадкові числа з діапазону `[-0.5,0.5]`. Обчислюється сума їх квадратів і, якщо вона менша за `0.5^2` (тобто згенерована пара чисел, що становить координати точки всередині кола радіуса `0.5`), то умовний оператор повертає 1, інакше – 0. Оператор `sum(...,i,1,10000)` обчислює кількість цих одиниць, тобто кількість точок N_{circle}, що потрапили в коло. Потім результат перетворюється відповідно до формули $4 \cdot N_{circle} / N_{square}$, яка приблизно відповідає числу π.

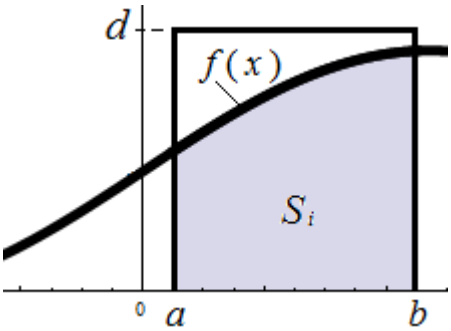
Expression – **Description**= `True, Pi`

LMB > Evaluate

Буде створено таблицю **Table 1** з однією коміркою і значенням `3.1380`. Обравши інші початкові значення в другому аргументі функцій `random(i,N)`, отримаємо інші результати, але достатньо близькі до числа π (при великій кількості випадкових точок).

Pi, Point: 0
3.1380

Обчислення визначеного інтеграла $S_i = \int_a^b f(x)dx$ $(f(x) \geq 0)$ який, як відомо, представляє площу криволінійної трапеції, обмеженою зверху кривою $y = f(x)$, знизу віссю X, зліва і справа прямими $x = a$, $x = b$.



Візьмемо відрізок інтегрування [a,b] і обмежимо його прямокутником $a \leq x \leq b$, $0 \leq y \leq d$, де $\max_{a \leq x \leq b} f(x) \leq d$ з площею $S_{rect} = (b-a) \times d$. Якщо заповнити прямокутник точками з випадковими координатами, то відносна кількість точок N_i / N_{rect}, які потраплять в криволінійну трапецію, буде пропорційна відношенню зазначених площ. Тобто $S_i / S_{rect} \approx N_i / N_{rect}$, або $S_i \approx S_{rect} \cdot N_i / N_{rect}$, де N_i - кількість точок, які потрапили в зону криволінійної трапеції, а N_{rect} - загальна кількість точок (якщо всі вони потрапляють в прямокутник).

Реалізуємо описані обчислення в COMSOL, узявши $f(x) = \sin \pi x$, і поклавши $a = 0, b = 1$. Звичайно відповідь відома $\int_0^1 \sin \pi x \, dx = 2/\pi \approx 0.63662$.

Обчислення виконаємо в поточному файлі у вузлі `Point Evaluation`.

Derived Values - Point Evaluation 1: RMB > Duplicate
Point Evaluation 2:

Expression – **Expression** = `sum(if(0.5+random(i,1)<sin(pi*(0.5+random(i,2))),1,0),i,1,10000)/10000`

Тут перевіряється нерівність $y < \sin \pi x$, де x, y пара випадкових чисел, рівномірно розподілених на відрізку [0,1]. Якщо вона виконується, то це означає, що випадкова точка з координатами (x, y) потрапила всередину криволінійної трапеції.

LMB > Evaluate –Table 1

В таблиці **Table 1** буде додано комірку зі значенням `0.63970`.

Pi, Point: 0	Pi, Point: 0
3.1380	0.63970

Зауваження. Обчислення визначеного інтеграла як площі під кривою краще виконувати стандартними методами чисельного інтегрування. Але коли крива задана неявно, або набором нерівностей, то ймовірнісний підхід може виявитися кращим.

Випадкові функції з інтерполяційною густиною розподілення. В COMSOL є спосіб створення випадкових функцій з густиною розподілення, заданою як інтерполяційна функція. Якщо вважати інтерполяційну функцію щільністю розподілення, то задати функцію генерування випадкової величини на її основі можна автоматично, встановивши прапорець **Define random function** (визначити випадкову функцію) в ПВ цієї інтерполяційної функції. Поруч можна задати ім'я такої функції (за замовчуванням `rn_int1`)

Продемонструємо зазначену можливість на прикладі наступних даних.

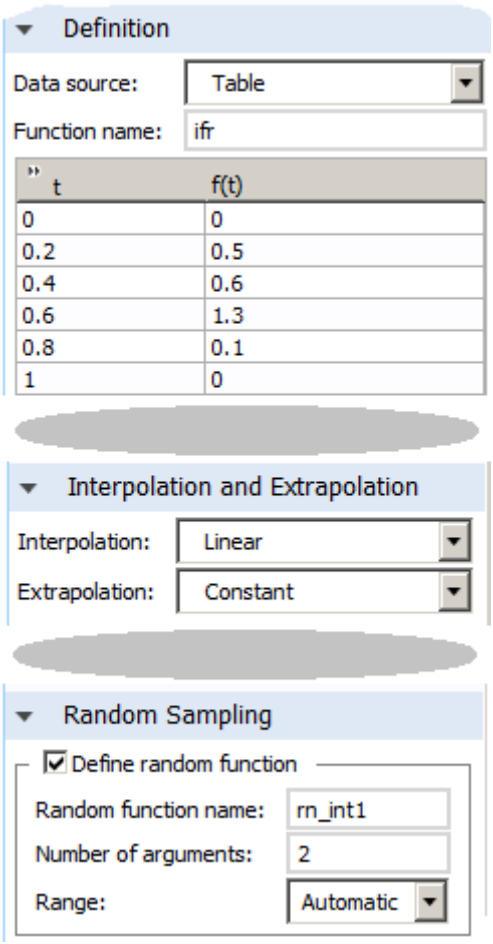
x	0	0.2	0.4	0.6	0.8	1
$f(x)$	0	0.5	0.6	1.3	0.1	0

Відкрийте останній файл і створіть в ньому інтерполяційну функцію.

Global Definitions: RMB > Functions – Interpolation

Definition – **Data source** = `Table` (*координати вузлів в таблиці*)
– **Function name** = `ifr`

Введіть координати вузлів в таблиці так, як зображено на наступному рисунку.



Оберіть тип інтерполяції.

`Interpolation and Extrapolation` – **Interpolation** = `Linear`

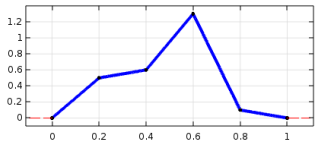
Встановіть прапорець створення випадкової функції і визначіть кількість її аргументів, наприклад, 1.

`Random Sampling` – **Define random function** = `True`

– **Number of arguments** = `1`

Ім'я функції генерування випадкової величини залиште незмінним - `rn_int1`.

Виконайте контрольну побудову графіка (будується крива `ifr(x)`, тобто густина ймовірності, а не випадкова функція `rn_int1(x)`).



LMB> Plot

Тепер побудуйте графік випадкової вибірки.

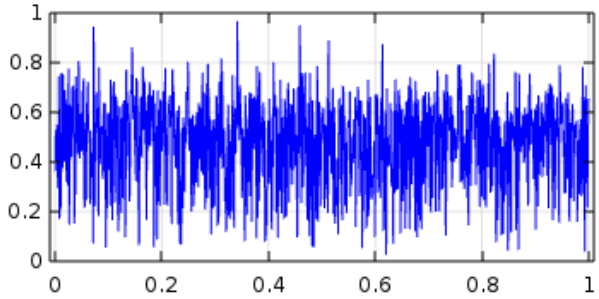
1D Plot Group 9: RMB > Duplicate

1D Plot Group 11- Line Graph 1:

`Data` – **Data set** = `Grid 1D 3`

`y-Axis Data` – **Expression** = `rn_int1(x)`

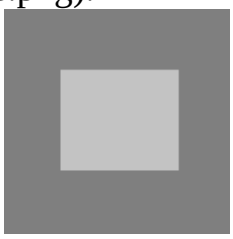
LMB> Plot



Зі створеною функцією `rn_int1(x)` можна працювати, як з іншими випадковими функціями.

Image-функції і генерування шуму. Випадкові величини можна розглядати як шум у сигналі або на зображенні. Розглянемо приклад накладання гаусового шуму на зображення, яке складається лише з двох складових яскравості (наприклад, з двох сірих кольорів). Зробимо це в поточному файлі.

Спочатку створіть в Paint (або в будь якому іншому графічному редакторі) зображення розміром 128×128 пікселів (розмір несуттєвий), і збережіть його в файл (наприклад TwoColorSquare.png).



В COMSOL є функція-оболонка `Image`, що призначена для роботи з зображеннями, як з функціями двох змінних. Вона є функцією користувача, що інтерпретує кольорове зображення як функцію двох змінних (за замовчуванням перетворює кольорові яскравості пікселів в значення `(R+G+B)/3`).

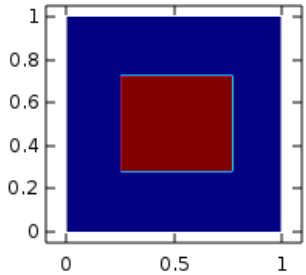
Відкрийте останній файл і створіть в ньому Image-функцію.

Global Definitions: RMB > Functions – Image

Function name = `im1`

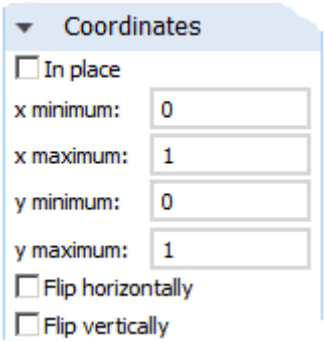
File – **Filename** = `D:\Work\COMSOL\TwoColorSquare.png`

LMB> Plot



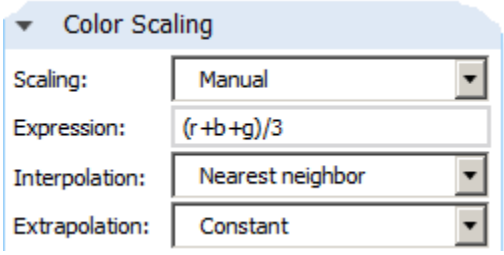
Колір зображення відрізняється від оригіналу тому, що при побудові рисунка система обрала «власну» палітру кольорів. В оригінальному рисунку кожний піксель кольоровий і представлений значеннями трьох яскравостей основних кольорів: Red, Green, Blue (R,G,B). Після завантаження рисунка в COMSOL ці три числа перетворилися на одно (середнє `(r+g+b)/3`), якому в різних палітрах відповідають різні кольори. В тій, яку обрала система по команді Plot, зображення виглядає так, як тут показано.

В ПВ створюваного вузла **Image 1** в розділі **`Coordinates`** визначаються діапазони зміни аргументів функції в межах зображення. Якщо в якості координат потрібно використовувати індекси пікселів (по горизонталі і вертикалі), то встановіть прапорець **In place**. За допомогою прапорців **Flip horizontally** і **Flip vertically** можна перевертати зображення по горизонталі і/або вертикалі.

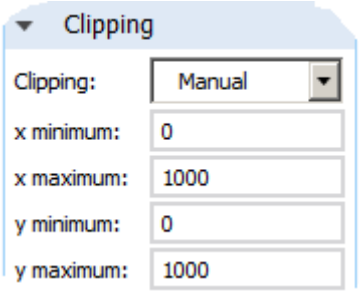


В розділі **`Color Scaling`** в списку **Scaling** можна обрати режим `Manual` ручного переведення векторів (`R,G,B`) кольорів пікселів до одного значення (за замовчуванням встановлена формула `(r+g+b)/3`). В результаті створюється матриця зображення. Вона використовується як дані інтерполяційної функції двох змінних. Тип інтерполяції визначається налаштуванням в полі **Interpolation** (`Nearest neighbor` – найближчий сусід, або `Linear` - лінійний).

Поведінка функції поза зоною інтерполяції визначається налаштуванням в полі **Extrapolation**. Це можуть бути значення `Constant`, `Linear`, `Nearest function` або `Specific value`.



В розділі **`Clipping`** визначається обрізання зображення. Тут вказуються мінімальний і максимальний індекси пікселів, визначаючи зону графічного файла, яка передаватиметься в Image-функцію. Якщо в списку **Scaling** вказано `None`, то відсікання немає і передається все зображення. При значенні `Manual` в полях, що розташовані нижче, вказуються номери пікселів (цілі числа), які визначають межі обрізання.



Для демонстрування можливостей додавання до Image-функції іншої функції двох змінних створимо дві аналітичні функції. Перша `bub1(x,y,r,x0,y0,c)`, як функція змінних `(x,y)`, є циліндричним стовпчиком радіуса `r` з центром в точці `(x0,y0)` і висотою `c`.

Global Definitions: RMB > Functions – Analytic

Function name = `bub1`

Definition – **Expression** = `if((x-x0)^2+(y-y0)^2<r^2,c,0)`

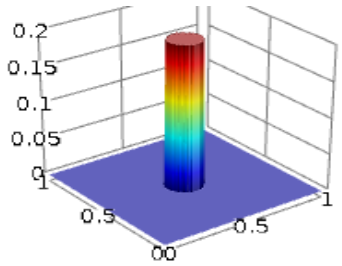
– **Arguments** = `x,y,r,x0,y0,c`

Виконайте побудову графіка функції, використавши фіксовані значення чотирьох останніх аргументів, задавши їх в розділі **Plot Parameters**.

LMB> Plot

Plot Parameters

Argument	Lower limit	Upper limit
x	0	1
y	0	1
r	0.1	0.1
x0	0.5	0.5
y0	0.5	0.5
c	0.2	0.2



Друга функція `bub3(x,y)` є сумою трьох функцій `bub1(x,y,...)`.

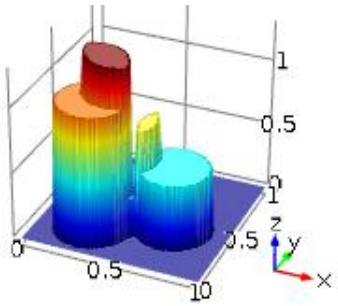
Global Definitions: RMB > Functions – Analytic

Function name = `bub3`

Definition – **Expression** = `bub1(x,y,0.2,0.3,0.25,1)+ bub1(x,y,0.2,0.7,0.4,0.5)+bub1(x,y,0.2,0.4,0.4,0.3)`

– **Arguments** = `x,y`

LMB> Plot



Для побудови двовимірного зображення нам знадобиться `Grid 2D` джерело даних розміром [0,1]×[0,1], який визначено діапазонами, заданими в розділі **Coordinates** ПВ Image-функції.

`Results` -> Data Set - More Data Sets -> Grid 2D

Results - Data Sets - Grid 2D 7 :

Data – **Source, Function** = `Function, All`

Побудуйте зображення суми Image-функції і `bub3(x,y)`.

`Results` -> **Plot Group** -> 2D Plot Group

2D Plot Group 12:

Data – **Data set** = `Grid 2D 7`

`2D Plot Group 12`-> **Add Plot** -> Surface

2D Plot Group 12 – Surface 1:

Expression – **Expression** = `im1(x,y)+bub3(x,y)`

Coloring and Style – **Coloring, Color** = `Color table,GrayScale`

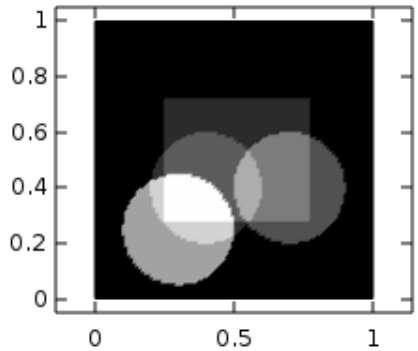
LMB> Plot (*наступний рисунок ліворуч*)

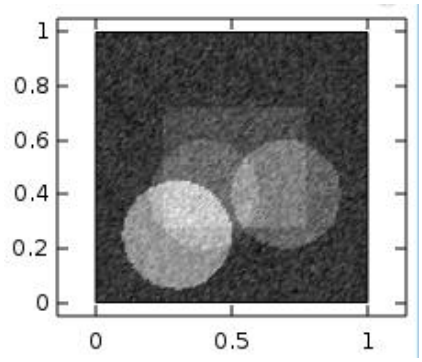
Тепер на зображення накладемо «шум», додавши до останнього виразу випадкову функцію двох змінних. Щоб «основу» зображення було видно, амплітуду шуму оберемо невеликою.

2D Plot Group 12 – Surface 1:

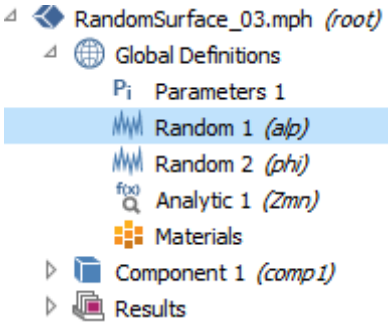
Expression – **Expression** = `im1(x,y)+bub3(x,y)+rn2(x,y,1)/6`

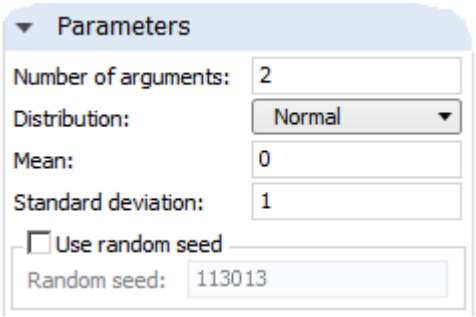
LMB> Plot (*наступний рисунок праворуч*)





Зауваження. В новітніх версіях COMSOL у рандомних функцій з'явилася опція **Use random seed**, яка дозволяє задавати додаткову ініціалізацію при генеруванні випадкових чисел. Ось як виглядає розділ **Parameters** ПВ вузла **Random (alp)** в COMSOL 5.5.





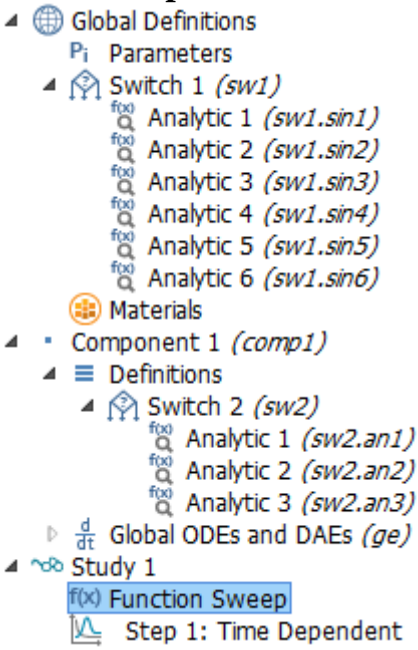
Можливості використання випадкових функцій в COMSOL великі. Але, як ви бачили, інколи їх недостатньо. Тоді на допомогу може прийти Matlab. Існують різні режими взаємодії цих систем. Найпростіший полягає в створенні в COMSOL функцій-оболонок до функцій Matlab. Це питання було розглянуто в попередньому параграфі.

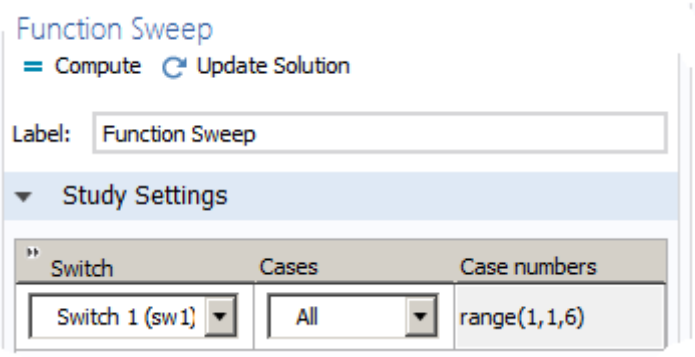
6.6. Прогонка по функціям (Function Sweep).

Інколи виникає потреба в аналізуванні моделі, використовуючи різні функції зовнішнього впливу, або різні функціональні залежності матеріальних коефіцієнтів, різні функції початкових умов, тощо. Це схоже на параметричний аналіз, коли модель обчислюється і аналізується при різних значеннях параметра. Аналізування моделі, коли в рівняннях або додаткових умовах змінюються функції-коефіцієнти, в COMSOL називається Function Sweep (прогонка по функціям). «Список» функцій, по яким виконується прогонка,

задається у спеціальному вузлі функцій Switch, створюваному в гілці Definitions (глобальній або в компоненті). Об'єкт (або функція), що являє цей вузол, постає в ролі «перемикача». Ви додаєте звичайні функції користувача як підвузли вузла Switch, і вони потім по черзі використовуються при Function Sweep обчисленні моделі. Перемикачу дається назва (за замовчуванням `sw1`) і воно вводиться, як ім'я функції в полі виразу, де ви хочете переключатися між функціями.

Прогонка по функціям вмикається при додаванні у гілку **Study** вузла **Function Sweep**.





В ПВ цього вузла в таблиці **`Study Settings`** в стовпці `Switch` обирається функція-перемикач. Для цього призначена кнопка + Add. В стовпці `Cases` вказується спосіб вибору дочірніх функцій (`All` або `User defined`). Щоб використати всі функціональні вузли під комутатором, виберіть `All`. Якщо обрано `User defined`, то в стовпці `Case numbers` слід вказати номери функцій в списку перемикача (з відліком зверху).

Якщо прогонка по функціям не застосовується (**Function Sweep** вузол у вузлі **Study** не використовується), то перемикач (за замовчуванням) повертає свою першу функцію. Після виконання прогонки по функціям у «постпроцесорних» вузлах ви можете обрати будь-які функції зі списку перемикача, наприклад, побудувати графіки отриманих розв'язків лише для деяких «дочірніх» функцій.

Приклад 1. Анімація графіка функції $\sin nx$ за допомогою Function Sweep.

Розв'язання. Створіть модель задачі з динамічним `Global ODEs and DAEs` інтерфейсом.

File -> New -> Model Wizard -> 0D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Задайте діапазон зміни параметра `t`.

Step 1: Time Dependent:

Study Settings – **Times** = `range(0,0.1,2*pi)`

Побудуйте функцію-перемикач зі списком дочірніх функцій $\sin x$, $\sin 2x$, ... , $\sin 6x$, тобто $\sin nx$ при $n = 1,2,3,4,5,6$.

Global Definitions: RMB > Functions – Switch

Function name = `sw1`

Global Definitions – Switch 1 : RMB > Analitic

Definition – **Expression** = `sin(x)`

Switch 1 : RMB > Analitic

Definition – **Expression** = `sin(2*x)`

Switch 1 : RMB > Analitic

Definition – **Expression** = `sin(3*x)`

Switch 1 : RMB > Analitic

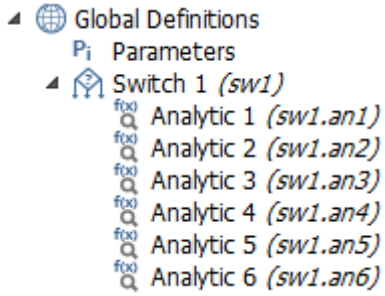
Definition – **Expression** = `sin(4*x)`

Switch 1 : RMB > Analitic

Definition – **Expression** = `sin(5*x)`

Switch 1 : RMB > Analitic

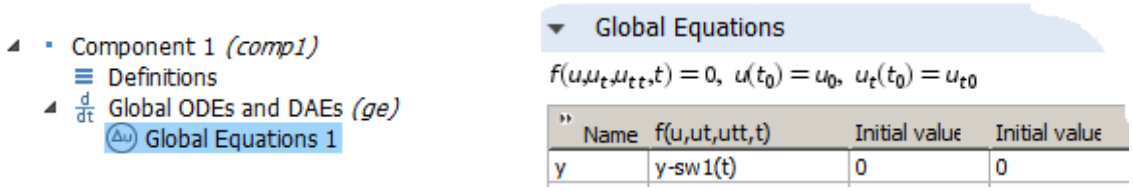
Definition – **Expression** = `sin(6*x)`



В ПВ вузла **Global Equations 1** заповніть таблицю рівнянь.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – Name, f(u, ut, utt, t) = `y, y-sw1(t)`

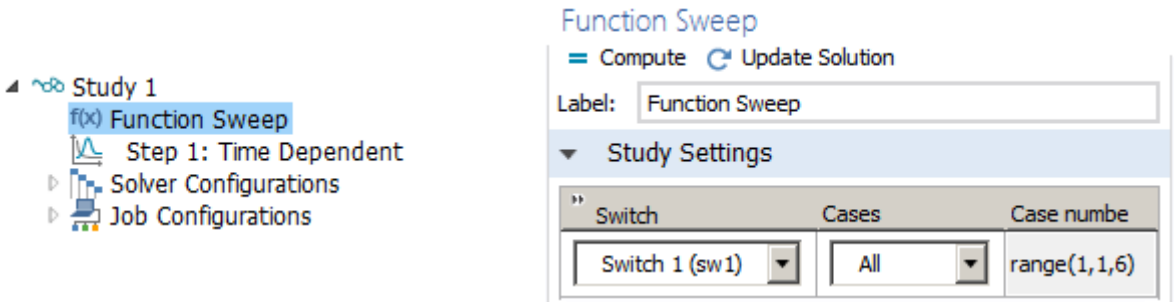


Зверніть увагу, як використано функцію-перемикач. В полі **f(u,ut,utt,t)** введено його ім'я `sw1(t)`. Система при Function Sweep аналізі замість нього буде по-черзі підставляти імена дочірніх функцій перемикача.

Додайте вузол прогонки по функціям.

Study 1: RMB > Function Sweep

В ПВ створеного вузла **Function Sweep** в розділі **Study Settings** є таблиця для вибору функцій-перемикачів. Клацніть в лівій колонці таблиці правою кнопкою миші і виконайте команду Add. Із списку виберіть, створений раніше перемикач `Switch 1(sw1)`. У другому стовпці `Cases` таблиці залиште значення `All`, тобто вкажіть прогонку по всім дочірнім функціям обраного перемикача.



Виконайте команду Compute і в графічній зоні вузла **Results - 1D Plot Group 1 - Global 1** отримайте графіки змінної y, які відповідають різним дочірнім функціям перемикача **Switch 1**. Налаштуйте вузол **Global 1**.

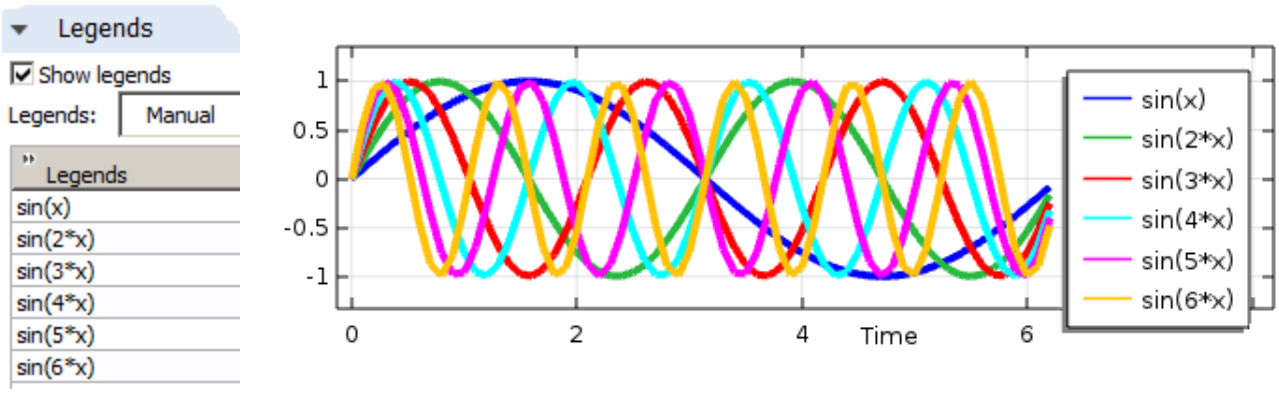
1D Plot Group 1 - Global 1:

Coloring and Style – **Line style** – **Width** = 2

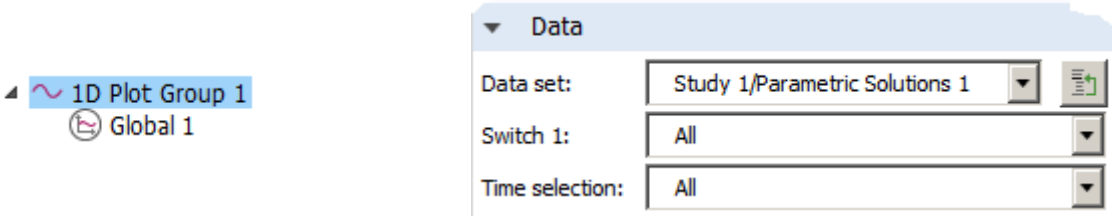
Legends – **Show legends, Legends** = True, Manual

– **Legends** = sin(x);sin(2*x);sin(3*x);sin(4*x);
sin(5*x);sin(6*x)

LMB> Plot



На цьому прикладі можна пояснити використання деяких опцій анімації. Спочатку погляньте на розділ **Data** графічної групи **1D Plot Group 1**. Він має наступний вигляд.



З нього видно, що в моделі є два параметри – часу і перемикача функцій **Switch 1**, і анімацію можна виконати по будь-якому з них.

Побудуйте вузол анімації на базі створеної графічної групи.

Results –> Animation –> Player

Animation 1:

Scene – **Subject** = 1D Plot Group 1

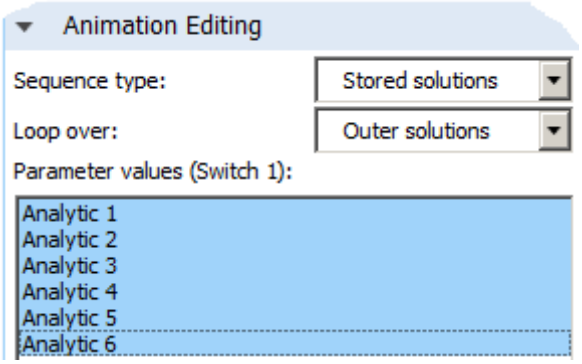
Якщо ви збираєтесь виконувати анімацію, змінюючи час, власне значення або параметр прогонки, то в полі **Sequance type** потрібно задавати значення Stored solutions. Воно обрано за замовчуванням і ми його залишимо.

В полі **Loop over** вказується параметр (або їх комбінація), який буде мінятися від кадру до кадру. В нашому прикладі це може бути час t, або

параметр прогонки. Щоб це був параметр (з Parametric, Auxiluary або Function Sweep), треба вибрати значення Outer solutions.

Animation Editing – Loop over = `Outer solutions`

Отже кадри анімації будуватимуться при зміні номера функції. В списку **`Parameter values (Switch 1)`**, розташованому нижче, можна обрати всі або деякі значення параметра (або комбінації параметрів). За замовчуванням обрані всі. Щоб обрати деякі, потрібно натиснуту на клавіатурі клавішу CTRL, а потім виділити мишею в списку потрібні значення. Щоб виділити діапазон значень, натисніть клавішу SHIFT, клацніть мишею по першому значенню діапазону, а потім по останньому.

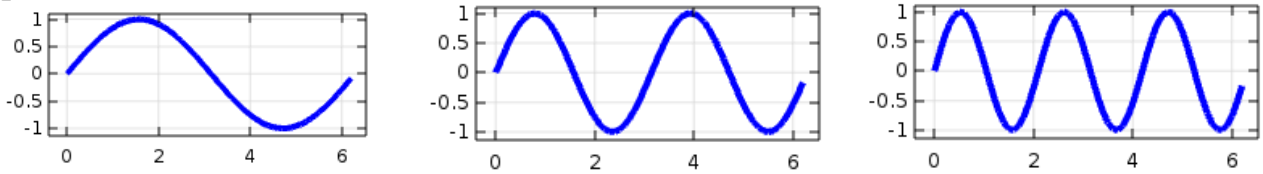


Frames – Frame selection = `All`

LMB > Generate Frame

Graphics (графічне вікно) : LMB > Play

На наступному рисунку наведені зображення змінної *y*, (тобто функції `sw1(t)`), отримані у вузлі анімації для перших трьох кроків функціональної прогонки.



$\sin x$ $\sin 2x$ $\sin 3x$

Зауваження. В розглянутій задачі простіше було б створити параметр моделі *n*, задати рівняння $y - \sin nx = 0$, і застосувати Parametric Sweep аналіз по скалярному параметру `n`. Але при використаному тут підході можна обирати будь-які функції, а не схожі синусоїди.

В поточному прикладі ми обрали найпростіший нульвимірний інтерфейс `Global ODEs and DAEs(ge)`. Але застосування прогонки по функціям при інших інтерфейсах нічим не відрізняється від показаного. Створюється перемикач функцій і його ім'я записується в полі, де потрібно змінювати функції. При `Function Sweep` аналізі це ім'я по-черзі замінюється іменами дочірніх функцій перемикача, а результати всіх обчислень (за замовчуванням) запам'ятовуються.

6.7. Демонстраційні приклади.

6.7.1. Генерування випадкової хвильоподібної поверхні.

Будь-яку хвильоподібну поверхню можна розглядати як складену з багатьох елементарних «хвиль» виду $a\cdot\cos\left(k_x x+k_y y+\varphi\right)$, де φ - фаза, $\boldsymbol{k}=\left(k_x,k_y\right)$ - хвильовий вектор, що показує «напрям хвилі» і її «довжину», a - амплітуда. При побудові прямокутної ділянки поверхні розміром `[0,1]x[0,1]` можна обмежитися значеннями $\boldsymbol{k}_{mn}=2\pi\left(m,n\right)$, де m,n – цілі з деякого скінченого інтервалу, наприклад, `[-M,M]`. Тоді

$$f\left(x,y\right)==\sum_{m,n}a_{mn}\cos\left(2\pi\left(m\,x+n\,y\right)+\varphi_{mn}\right).$$

Остання формула створює періодичну функцію з періодом, що визначається найповільнішими коливаннями, які відповідають числам m=1 або n=1. Щоб уникнути випадкової сталої, з останньої суми має сенс виключити доданок з одночасним m=0, n=0.

Для випадкової поверхні найбільш логічним значенням амплітуд були б випадкові числа з рівномірним або нормальним розподілом. Однак в природі повільні коливання зазвичай мають більшу амплітуду ніж швидкі. Це відповідає зменшенню амплітуд зі збільшенням індексів m і n, наприклад, наступним чином $a_{mn}\sim\dfrac{1}{m^2+n^2}$. Дотримуючись випадковості, коефіцієнт пропорційності $\alpha\left(m,n\right)$ оберемо як випадкову величину, розподілену за законом Гауса, і покладемо $a_{mn}=\dfrac{\alpha\left(m,n\right)}{m^2+n^2}$. Фазу $\varphi_{mn}=\varphi\left(m,n\right)$ кожного доданка обиратимемо як рівномірно розподілену величину в діапазоні $0\le\varphi\left(m,n\right)\le\pi$ (величина більша за π еквівалентна зміні знака амплітуди). Таким чином формула, яка використовуватиметься для створення випадкової поверхні, матиме вигляд

$$f\left(x,y\right)=\sum_{\substack{m,n=-M\\ m,n\neq 0}}^{M}\frac{\alpha\left(m,n\right)}{m^2+n^2}\cos\left(2\pi\left(m\,x+n\,y\right)+\varphi\left(m,n\right)\right). \qquad (1)$$

Приклад. Генерування випадкової поверхні хвильоподібного виду.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Значення M задайте як параметр моделі. Ще один параметр H застосуємо в якості масштабного множника до амплітуд «хвиль».

Global Definitions : RMB > Parameters

`Parameters` – Name, Expression = `M,10`

`Parameters` – Name, Expression = `H,0.025`

Для визначення величин $\alpha(m,n)$ і $\varphi(m,n)$ створіть дві випадкові функції двох змінних.

Global Definitions : RMB > Functions – Random

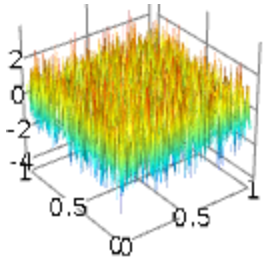
Function name = `alp`

Parameters – **Number of arguments** =2

– **Distribution** = `Normal`

– **Mean, Standard deviation** = `0,1`

LMB > Plot



Global Definitions : RMB > Functions – Random

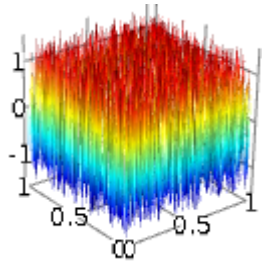
Function name = `phi`

Parameters – **Number of arguments** =2

– **Distribution** = `Uniform`

– **Mean, Standard deviation** = `0,pi`

LMB > Plot



Перед створенням змінної за формулою (1), створіть функцію, яка обчислює один доданок в цій формулі. Він є функцією 4-х змінних `x,y,m,n`.

Global Definitions: RMB > Functions – Analytic

Function name = `Zmn`

Definition – **Expression** = `alp(m,n)*cos(2*pi*(m*x+n*y)+ phi(m,n))/(m^2+n^2)`

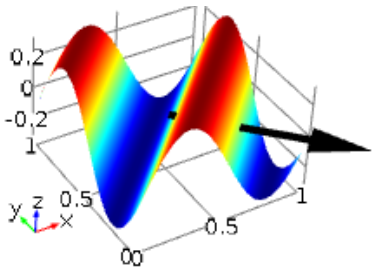
Definition – **Arguments** = `x,y,m,n`

Для перевірки виконайте побудову графіка цієї функції при фіксованих значеннях `m=1` і `n=-1`.

LMB> Plot

Plot Parameters

Argument	Lower limit	Upper limit
x	0	1
y	0	1
m	1	1
n	-1	-1



Для наочності ми ще додали зображення напрямку «хвилі» у вигляді вектора $\boldsymbol{k}=(m,n)$ (команда Plot його не малює).

Додайте в модель компонент і в його вузлі **Geometry** створіть параметричну поверхню.

Untitled.mph (root) : RMB > Add Component -> 3D

`Geometry` -> **`Primitives - More Primitives`** – Parametric Surface

Geometry 1 – Parametric Surface 1 (ps1):

Діапазони параметрів залиште незмінними. Тоді поверхня буде створюватися над квадратом `[0,1]x[0,1]`.

Expression – **x** = `s1`

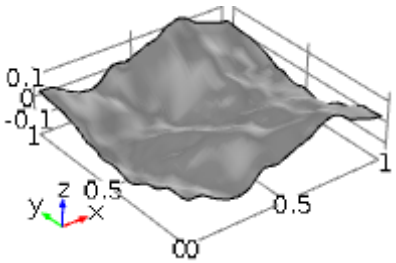
– **y** = `s2`

– **z** = `H*sum(sum(if((m!=0)||(n!=0), Zmn(s1,s2,m,n),0),m,-M,M),n,-M,M)`

Advanced Settings – **Relative tolerance** = 1E-3

Зверніть увагу на те, що ми зменшили точність «внутрішнього» зображення поверхні (опція **Relative tolerance**=1E-3). Вона використовується в рівняннях B-сплайнів, які застосовуються системою при її моделюванні. Зазвичай кількість вузлових точок в сплайнах підбирається так, щоб відносна похибка не перевищувала значення в полі **Relative tolerance**, або доки не досягнуто кількості вузлів, зазначеної в полі **Maximum number of knots**.

LMB > Build Selected



Як бачите, система побудувала поверхню, але попередила що бажаної похибки не досягнуто.

Побудуйте тіло, яке обмежене зверху створеною поверхнею. Для цього використайте «блок», який ця поверхня перерізає, а потім видаліть його верхню частину.

Geometry -> **Primitives** – Block

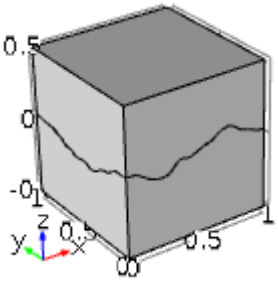
Geometry 1 – Block 1:

Size – **Width, Depth, Height** = 1,1,1 (*три поля*)

Position – **Base** = Corner

Position – **x, y, z** = 0,0,-0.5

LMB > Build Selected



Наступним кроком буде видалення верхньої частини. Для цього знадобиться інструмент «розбиття об’єктів», який створює Partition Objects вузол. Він реалізує розділення геометричних об'єктів на окремі частини. В цьому вузлі вказується поділюваний об’єкт (фігура у просторі) і геометричні об’єкти, які застосовуються в якості розрізу (поверхня). Результатом є частини первинного об’єкта, що в поєднанні являють початковий об’єкт.

Geometry -> **Booleans and Partitions** – Partition Objects

Partition Objects 1 (par1):

Partition Objects–**Objects to partition** = blk1 (*тіло, що роз’єднується*)

Partition Objects – **Tool objects** = ps1 (*розділяюча поверхня*)

LMB > Build Selected

Після застосування операції Partition Objects, верхівка блока буде відокремлена від низу. Цього зразу не видно, але на наступному кроці в операції видалення ви вже зможете її вибирати.

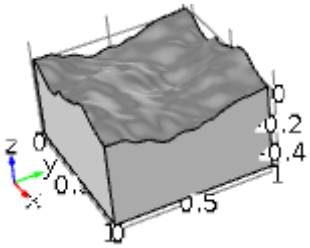
Geometry **-> Operations** – Delete

Delete Entities 1 (del1):

Entities or Objects to Delete – **Geometric entity level** = Domain

– **Selectin :** = par1-2

LMB > Build All Objects



Створене тіло можна використовувати для цілей побудови СкЕ сітки і розв'язання в ньому, як в розрахунковій геометрії, ДРЧП.

6.7.2. Сферично-симетричні коливання газу.

Розповсюдження звука в газі описується хвильовим рівнянням $u''_{tt} = a^2 \Delta u$, де $u(x, y, z, t)$ – потенціал швидкостей. Сам вектор швидкості дорівнює $\boldsymbol{v} = -\mathrm{grad}\, u$ ([2], с. 106). Конденсація газу $s = (\rho - \rho_0)/\rho_0$, тобто відносна зміна його густини ρ, пов'язана з потенціалом u співвідношенням $s = u'_t / a^2$. Таким чином, знання потенціалу швидкостей достатньо для визначення всього процесу коливань газу, оскільки $\rho = \rho_0 (1 + s)$.

Припустимо, що зона початкової конденсації газу є кулею одиничного радіуса, від якої у всі боки рівномірно розповсюджуватимуться коливання. Розмістимо початок координат в центр цієї кулі. В сферично-симетричному випадку рівняння, яке описує зміну потенціалу швидкостей u, матиме вигляд

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{1}{r} \frac{\partial^2}{\partial r^2} (r u).$$

Зробивши заміну $v(r,t) = r \cdot u(r,t)$, отримуємо $v''_{tt} = a^2 v''_{rr}$. Початкова умова $u(r,0) = \varphi(r)$ перетворюється на $v(r,0) = r \cdot \varphi(r)$, а умова $u'_t(r,0) = \psi(r)$ - на $v'_t(r,0) = r \cdot \psi(r)$.

Приклад. Дослідити сферично-симетричну задачу коливань газу, коли початкові швидкості всюди дорівнюють нулю, а початкова конденсація є сталою s_0 всередині кулі одиничного радіуса і дорівнює нулю поза нею. Тобто

$$u(r,t) = \varphi(r) = 0, \quad u'_t(r,0) = \psi(r) = \begin{cases} a^2 s_0, & r < 1 \\ 0, & r > 1 \end{cases}.$$

Обрати $a = 1$, $s_0 = 1$.

Оскільки розглядатиметься сферично-симетрична задача, то потрібно розв'язати рівняння $v''_{tt} = a^2 v''_{rr}$ на додатній півосі $0 \le r < \infty$ з початковими умовами $v(r,0) = 0$, $v'_t(r,0) = \tilde{\psi}(r) = r \cdot \psi(r)$. Вважаючи $u(0,t)$ обмеженим для всіх моментів часу, зі співвідношення $v = r \cdot u$ отримуємо $v(0,t) = 0$. Це буде граничною умовою в точці $r = 0$. З неї випливає можливість застосування метода продовження: початкові функції потрібно продовжити на всю дійсну вісь непарно і застосувати формулу (3) п. 6.2. В поточному прикладі ($\varphi(r) = 0$, $f \equiv 0$) вона набуває вигляду

$$v(r,t) = \frac{1}{2a} \int_{r-at}^{r+at} \Psi(\alpha) d\alpha ,$$

де $\Psi(r)$ - є непарним продовження функції $\tilde{\psi}(r) = r \cdot \psi(r)$, яке можна представити у вигляді $\Psi(r) = r \cdot \psi(|r|)$. Остаточно отримуємо

$$u(r,t) = \frac{1}{2ar} \int_{r-at}^{r+at} \alpha \, \psi(|\alpha|) d\alpha .$$

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `L=6, a=1 i T=L/a` (час спостереження).

Створіть в моделі одновимірний компонент.

Untitled.mph : RMB > Add Component - 1D

Його геометрія буде складатися з відрізка [0, *L*].

Geometry->**`Primitives`->** Interval

Geometry 1 - Interval 1:

`Interval` – Left endpoint, Right endpoint = `0,L`

LMB > Build All Objects

Генеруйте СкЕ сітку.

Mesh 1: `Mesh Settings` –Element size = `Extremely fine`

LMB > Build All

Додайте в модель вузол динамічного дослідження.

Home -> Study - Add Study = Add Study ->

Preset Studies - Time Dependent -> **x**

У досліджуваної задачі є кілька особливих часових інтервалів, в які бажано спостерігати за процесом коливань ретельніше. Для них частоту спостережуваних моментів збільшимо (частіше в околі моментів `t=0` і `t=1`).

Study 1 - Step 1: Time Dependent:

`Study Settings` –Times = `0,0.01,0.02,0.03,0.05,0.07,0.09,`
`range(0.1,0.1,0.9), 0.95,0.97,0.98,0.99,1,1.01,`

`1.02,1.03,1.05,range(1.1,0.1,T)`

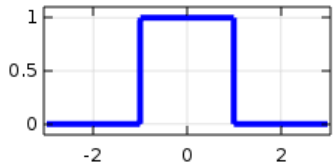
Створіть функцію $\psi(x)$ початкової конденсації у вигляді `Rect`-функції.

Global Definitions: RMB > Functions – Rectangle

Function name = `rect1`

Parameters– **Lower limit, Upper limit** = `-1, 1`

Smoothing – **Size of transition zone** = `False`



Для перевірки виконайте побудову графіка функції.

LMB> Plot

Оскільки шукана величина $v(r,t)$ обчислюється інтегруванням, то її треба задати як змінну моделі. Для конденсації $s = u_t' / a^2$ теж створимо змінну.

`Definitions` -> **Variables** – Local Variables

Definitions - Variables 1:

Variables – **Name** = `v`

– **Expression** = `integrate(ksi*rect1(abs(ksi)), ksi,x-a*t,x+a*t)/(2*a)`

Variables – **Name** = `s`

– **Expression** = `d(v,t)/(a^2*x)`

Виконайте обчислення і побудуйте графік змінної `u=v/x` (потенціалу швидкостей).

Study 1: RMB > Compute

`Results` –> Plot Group - 1D Plot Group

1D Plot Group 1:

Data – **Dataset** = `Study 1/Solution 1`

– **Time selection** = `Manual`

– **Times (1-74)** = `range(3,5,75)`

`1D Plot Group 1`–> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

Selection – **Selection, <>** = `Manual,1` (*відрізок* [*0, L*])

y-Axis Data – **Expression** = `v/x` (*потенціал швидкостей*)

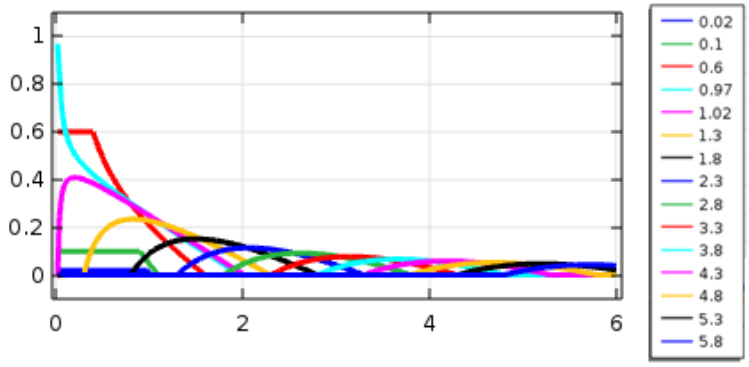
Тут в якості *r* – відстані до початку координат, виступає змінна `x`.

x-Axis Data – **Parameter, Expression** = `Expression,x`

Coloring and Style –**Width** = `3`

Legends – **Show legends** = `True`

LMB > Plot



Графік малоінформативний, тому краще спостерігати за анімацією зміни потенціалу *u* (анімацію побудуйте самостійно).

Побудуйте графіки конденсації в ті ж самі моменти часу.

1D Plot Group 1: RMB > Duplicate

1D Plot Group 2:

Axis – **Manual axis limits** = `True`

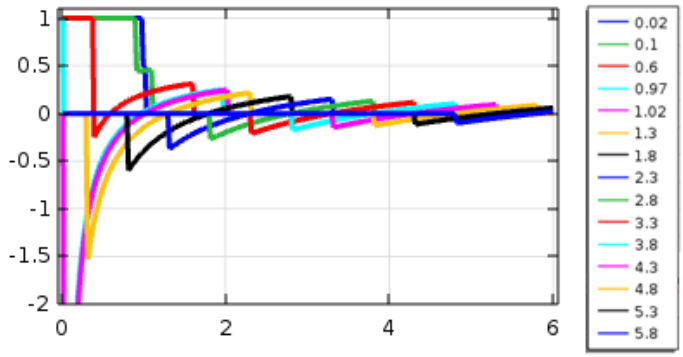
– **y minimum, y maximum** = `-2,1.1`

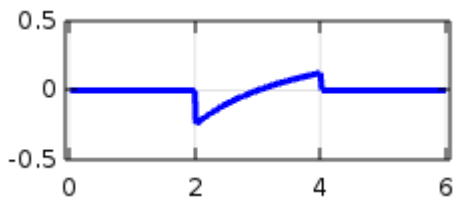
1D Plot Group 2 - Line Graph 1:

y-Axis Data – **Expression** = `s` (*конденсація*)

LMB > Plot (*наступний рисунок ліворуч*)

Якщо побудувати графік конденсації в фіксований момент часу, наприклад, при `t=3`, то отримаємо рисунок, зображений праворуч (зробіть це самостійно).





Побудуйте графік залежності конденсації від часу `t` в фіксованій точці середовища, наприклад, розташованій на відстані 3 одиниці від початку координат.

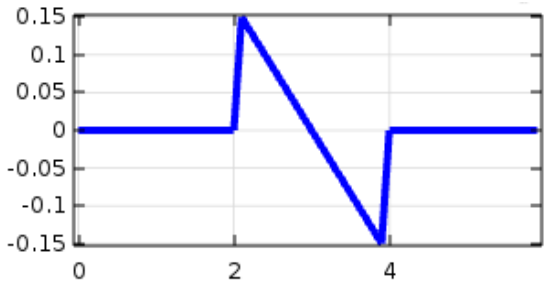
`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 3` -> Add Plot – Global

1D Plot Group 3 - Global 1:

y-Axis Data – **Expression** = `at1(3,s)` (*конденсація s в точці r=3*)

LMB > Plot



Графік сферичної хвилі конденсації `s` буде малоінформативним, тому побудуємо його як поверхню над двовимірним перерізом сферичного простору. Оскільки координата *x* постає в ролі відстані до початку координат, то її треба реалізувати як абсцису в ДД `Revolution 1D`. Створіть його.

`Results` -> Data Set - More Data Sets -> Revolution 1D

Results - Data Sets - Revolution 1D 1 :

Data – **Data set** = `Study 1/Solution 1`

Axis Data – **x** = `0`

Тепер над цим ДД побудуйте поверхню змінної `s`.

`Results` -> Plot Group - 2D Plot Group

2D Plot Group 4 :

`Data` – **Data set** = `Revolution 1D 1`

– **Time** = `4`

`2D Plot Group 4` -> Add Plot - Surface

2D Plot Group 4 - Surface 1 :

`Expression` – **Expression** =`s`

Щоб палітра кольорів і діапазон значень (по вертикалі) були однаковими для всіх моментів часу, бажано встановити їх мінімальні і максимальні значення в опціях поточного вузла.

`Range` – **Manual color range** = `True`

– **Minimum, Maximum** = `-1,1`

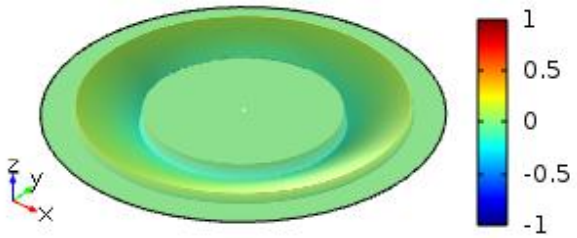
`Range` – **Manual data range** = `True`

– **Minimum, Maximum** = `-1,1`

RMB > Height Expression

Surface 1 – Height Expression 1 : `Axis` – **Scale factor** = `True, 3`

LMB > Plot



З останніх графіків зрозуміло, що сферичний шар, який представляє хвилю, розділено навпіл. В його зовнішній частині конденсація газу додатна, у внутрішній – від'ємна.

Зауваження. В [2] (с. 117) наводиться розв'язок розглянутої задачі в «кусковому» вигляді для $r > R$

$$s(r,t)=\begin{cases} 0 \quad , \quad 0 \le t \le (r-R)/a \\ s_0(r-at)/2r \, , (r-R)/a \le t \le (r+R)/a \\ 0 \quad , \quad t \ge (r+R)/a \end{cases},$$

де R – радіус кулі, в якій початкова конденсація ненульова. Перевірте, що він співпадає з розв'язком, отриманим нами, створивши, наприклад, функцію `S(r,t)=if(abs(t-r)<1,(r-t)/(2*r),0)` і порівнявши її графіки, з графіками, побудованими в прикладі.

6.7.3. Поперечні коливання нескінченного пружного стрижня.

Поперечні коливання пружного стрижня описуються диференціальним рівнянням в частинних похідних

$$u''_{tt} + a^2 u''''_{xxxx} = 0,$$

де $u(x,t)$ - поперечний зсув точок осі стрижня по відношенню до початкового прямолінійного стану.

Будемо досліджувати рівняння на нескінченній прямій $-\infty < x < \infty$. Для однозначного визначення розв'язку потрібно задати початкові умови

$$u(x,0) = f(x) \quad \text{і} \quad u_t'(x,0) = a\, g''(x),$$

де $g''(x)$ позначає другу похідну функції $g(x)$, і функції $f(x)$ і $g(x)$ такі, що всі використовувані далі інтеграли збігаються (на нескінченності достатньо швидко прямують до нуля). Розв'язок Бусинеска цієї задачі має вигляд [3 с. 530], [4, с. 138].

$$u(x,t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f\left(x - 2\tau\sqrt{at}\right) \cdot \left(\cos\tau^2 + \sin\tau^2\right) d\tau \; +$$

$$+ \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} g\left(x - 2\tau\sqrt{at}\right) \cdot \left(\cos\tau^2 - \sin\tau^2\right) d\tau \; . \qquad (1)$$

Розглянемо задачу, коли початкова швидкість коливань дорівнює нулю, наприклад, $g(x) \equiv 0$.

Для обчислення в COMSOL інтегралів з нескінченними границями, в них можна зробити заміну таку, щоб межі інтегрування стали скінченними. Враховуючи, що $g(x) \equiv 0$ і, роблячи в (1) заміну $\tau = \dfrac{\xi}{1 - |\xi|}$, аналогічно до ф. (10) п. 6.2, отримуємо

$$u(x,t) = \frac{1}{\sqrt{2\pi}} \int_{-1}^{1} f\left(x - \frac{2\xi\sqrt{at}}{1-|\xi|}\right) \cdot \left(\cos\frac{\xi^2}{(1-|\xi|)^2} + \sin\frac{\xi^2}{(1-|\xi|)^2}\right) \frac{1}{(1-|\xi|)^2} d\xi \; . \quad (2)$$

Зауваження. Формула (2) не може бути застосовувана при `t=0`, оскільки тоді інтеграл буде розходитися.

Приклад. Дослідити задачу розповсюдження коливань в нескінченному стрижні початкова швидкість точок якого нуль, а початкове зміщення - всюди нуль, крім відрізка $[-0.5, 0.5]$, де вона дорівнює одиниці, тобто

$$u(x,0) = f(x) = \begin{cases} 1, -0.5 \le x \le 0.5 \\ 0, x < -0.5 \vee x > 0.5 \end{cases}.$$

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Додайте в модель вузол параметрів.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `L=3, a=1`.

Створіть в моделі одновимірний компонент.

Untitled.mph : RMB > Add Component - 1D

Його геометрія складатиметься з відрізка $[-L, L]$, який буде використовуватися при побудові графіків замість нескінченної в обидві боки прямої.

`Geometry` -> **`Primitives`**-> Interval

Geometry 1 - Interval 1:

Interval – **Left endpoint, Right endpoint** = `-L,L`

LMB > Build All Objects

Генеруйте СкЕ сітку з дуже маленькими скінченними елементами (довжиною `0.0001`).

Component 1 - Mesh 1:

Mesh Settings – **Sequence type** = `User-controlled mesh`

Component 1 - Mesh 1 – Size:

Element Size – **Custom**= `True`

Element Size Parameters – **Maximum element size**= `0.0001`

LMB > Build All

Додайте в модель вузол динамічного дослідження.

`Home` -> Study - Add Study = `Add Study` ->

Preset Studies - Time Dependent -> **x**

Задайте «обчислювані» моменти часу `t`.

Study 1 - Step 1: Time Dependent:

Study Settings –**Times** =

`range(0,0.001,0.01) 0.025 0.05 range(0.1,0.1,0.5)`

Тут для початкових моментів ми задали менші часові кроки ніж для подальших.

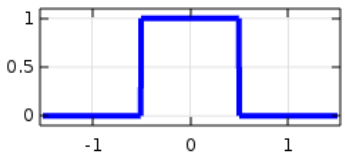
Створіть $f(x)$ у вигляді незгладженої `Rectangle`-функції.

`Home` -> Definitions -> Functions -> Local –Rectangle

Function name = `rect1`

Parameters– **Lower limit, Upper limit** = `-0.5,0.5`

Smoothing – **Size of transition zone** = `False`



Для перевірки побудуйте графік функції.

LMB> Plot

Для зсуву точок $u(x,t)$, який обчислюватимемо за формулою (2), створіть відповідну змінну моделі.

`Definitions` -> **Variables** – Local Variables

Definitions - Variables 1:

Variables – **Name** = `u1`

– **Expression** = `integrate(`

```
rect1(x-2*sqrt(a*t)*ksi/(1-abs(ksi)))*
(cos(ksi^2/(1-abs(ksi))^2)+
                sin(ksi^2/(1-abs(ksi))^2))/
(1-abs(ksi))^2,ksi,-1+eps,1-eps)/sqrt(2*pi)
```

Щоб уникнути ділення на нуль при наближенні змінної інтегрування до ± 1 ми звузили діапазон інтегрування `[-1,1]` на машинне епсілон з обох боків. Це не впливає на результат, але запобігає діленню на нуль.

Виконайте обчислення і побудуйте графік змінної `u1`. Оскільки формула (2) не може бути застосована при `t=0`, то графік для цього моменту у вигляді функції `rect1(x)` потрібно побудувати окремо.

Study 1: RMB > Compute

`Results` -> Plot Group - 1D Plot Group

1D Plot Group 1:

Data – **Data set** = `None`

`1D Plot Group 1`-> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

Data – **Data set** = `Study 1/Solution 1`

– **Time selection** = `First`

Selection – **Selection, <>** = `Manual,1` (*відрізок* [-*L*, *L*])

y-Axis Data – **Expression** = `rect1(x)`

x-Axis Data – **Parameter, Expression** = `Expression,x`

Coloring and Style – **Width** = `3`

Coloring and Style – **Color** = `Black`

Legends – **Show legends** = `True`

1D Plot Group 1 - Line Graph 1: RMB > Duplicate

1D Plot Group 1 - Line Graph 2:

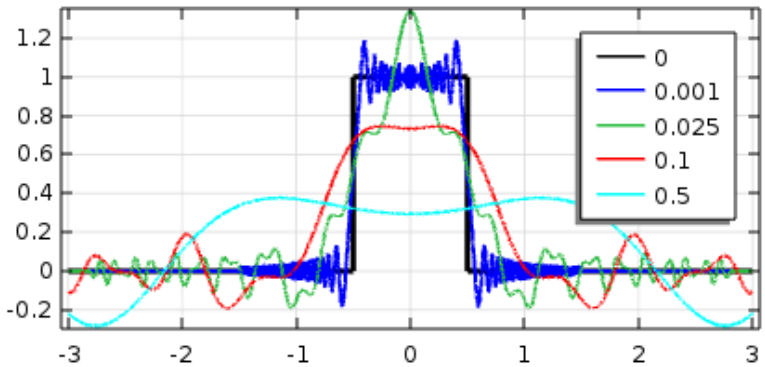
Data – **Time selection** = `From List`

– **Times** = `0.001,0.025,0.1,0.5`

y-Axis Data – **Expression** = `u1`

Coloring and Style – **Color** = `Cycle`

LMB > Plot



Оскільки осциляції, можливо, викликані розривами початкового зміщення `rect1(x)`, то має сенс це перевірити. Для цього в формулі (2) використайте згладжену Rectangle – функцію.

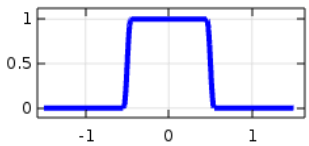
Побудуйте нову функцію $f(x)$ у вигляді згладженої `Rectangle`-функції.

Definitions - Rectangle 1(rect1): RMB > Duplicate

Function name = `rect2`

Smoothing – **Size of transition zone** = `True, 0.05`

LMB> Plot



Створіть змінну `u2` за тією ж формулою (2), що і `u1`, лише замініть в ній ім'я початкової функції на `rect2`.

Definitions - Variables 1:

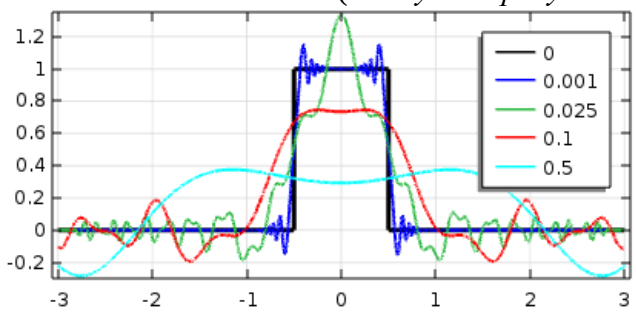
Variables – **Name** = `u2`

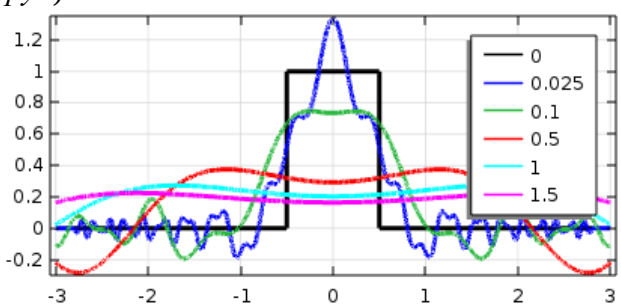
– **Expression** = `integrate(`

```
rect2(x-2*sqrt(a*t)*ksi/(1-abs(ksi)))*
```

```
(cos(ksi^2/(1-abs(ksi))^2)+
            sin(ksi^2/(1-abs(ksi))^2))/
(1-abs(ksi))^2,ksi,-1+eps,1-eps)/sqrt(2*pi)
```

Study 1: RMB > Compute
1D Plot Group 1: RMB > Duplicate
1D Plot Group 2 - Line Graph 2:
y-Axis Data – Expression = u2
LMB > Plot (*наступний рисунок ліворуч*)





Осциляції розв'язку в момент `t=0.001` зменшилися, але все одно залишилися. Також залишився стрибок амплітуди в центрі `x=0` в момент `t=0.025`.

Побудуйте самостійно графіки розв'язку для більших моментів часу, наприклад таких, що зображені на попередньому рисунку праворуч.

Зауваження. Розрив початкової функції можна інтерпретувати як злам стрижня, а осциляційні коливання, вірогідно, як те, що викликає чутний при цьому тріск.

Вправа. Розв'яжіть попередню задачу при $g(x) \equiv 0$ і початковому зміщенні у вигляді розривної (або згладженої) функцій Гевісайда. Така початкова умова краще імітує злам стрижня.

Література.

1. Тихонов А.Н., Самарский А.А. Уравнения математической физики. – М.: Наука, 1977. – 736с.
2. Кошляков Н.С., Глинер Э.Б., Смирнов М.М. Основные дифференциальные уравнения математической физики. – М.: Гос. изд. физ.-мат. лит., 1962.
3. Полянин А.Д. Справочник по линейным уравнениям математической физики. М.: ФИЗМАТЛИТ, 2001. – 576с.
4. Снеддон И. Преобразования Фурье. – М.: Изд. иностранной литературы, 1955. – 667 с.

7. Параметри і змінні.

В поточній главі розглядатимуться змінні і засоби роботи з ними, які досі докладно не обговорювалися в посібнику. Це геометричні і службові змінні, змінні систем координат користувача, змінні матеріальних властивостей, розривні змінні і змінні на межі розрахункових областей, а також змінні в «геометріях менших розмірностей».

7.1. Загальні відомості про параметри і змінні.

Операндами виразів COMSOL Multiphysics є параметри і змінні. Вони використовуються при створені і визначенні моделі, а також для візуалізації розв'язку і постпроцесорної обробки. Параметри і змінні зазвичай мають фізичні одиниці вимірювання, правила застосування яких було наведено наприкінці п. 1.7.

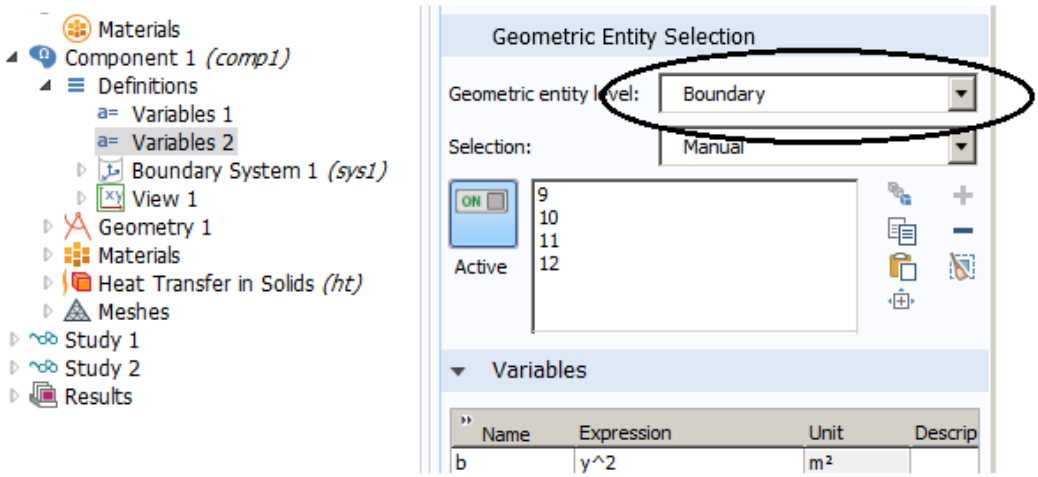
Параметри (Parameters) - це визначені користувачем сталі скаляри (константи) з глобальною зоною дії, які доступні для застосування у всіх місцях моделі, а якщо модель мультифізична, то у всіх «фізиках». Зокрема, параметри можна використовувати у гілках Geometry (Геометрія), Mesh (Сітка) і Study (Дослідження). Значення параметрів обчислюється перш, ніж починаються розрахунки в моделі. Параметри не повинні залежати від змінної часу `t`, від просторових змінних, і від змінних, відносно яких розв'язується задача. Будь-який глобальний параметр моделі, що не залежить від часу і просторових координат, може бути оформлений як іменована константа.

Змінні на відміну від параметрів, можуть залежати від часу, просторових координат та інших змінних. Як правило, вони визначаються у вузлах `Variables`, і можуть бути глобальними (доступними у всіх «фізиках» моделі) і локальними (визначеними лише для конкретного компонента). Змінні задаються за допомогою виразів. Вираз може містити числа, параметри, математичні і фізичні константи, інші змінні та функції від змінних, математичні операції і оператори. Ті змінні, що обчислюються на СкЕ сітці, називають змінними функцій форми. Інші задаються у вузлах `Variables` аналітично без прив'язки до скінчених елементів. Первинна інформація про змінні була наведена в п. 1.5.4. Тут ми обговоримо особливості застосування змінних, які не було розглянуто раніше.

Змінні, визначені в компоненті, мають локальний контекст. Але, якщо рівень геометричної сутності встановлено на весь компонент, то вони насправді є глобальними; їх можна застосовувати будь-де, використовуючи повне ім'я (тобто ім'я компонента, крапка, ім'я змінної). Наприклад, якщо ви визначаєте змінну `val` в компоненті **Component 1(comp1),** то можете посилатися на неї в іншому компоненті по імені `comp1.val`.

Окрім зони застосування (в межах компонента або глобально), локальні змінні можуть бути обмежені рівнем геометричної сутності. Змінні можуть визначатися в геометрії всього компонента, або мати рівень дії, обмежений деякою її частиною. Для цього існує налаштування вузла змінних, що визначає дію у всій геометрії або тільки в її частині (Domain, Boundaries, Edges, або

Points). Вибравши рівень геометрії і індикатори її частин, ви створюєте імена змінних, що діють лише на цих частинах.



Змінні, визначені в підобластях, доступні тільки для цих підобластей. Змінні, задані в областях, відрізняються від однойменних змінних, визначених на межі. Існує кілька типів рівнів геометрії, тому вузлів для змінних може бути багато.

Змінні не можуть використовуватися у гілках Geometry (Геометрія) та Mesh (Сітка), а у вузлі Study (Дослідження) лише у виразах, які застосовуються для зупинки обчислень.

Щоб отримати доступ до змінних, потрібно обчислити модель або поновити існуючий розв’язок.

Крім змінних, створених користувачем, кожний прикладний режим має свій власний набір змінних, які представляють величини змодельованих фізичних сутностей. Більшість змінних (властивості матеріалу, температура, координати та інші), що визначені у розрахунковій зоні, є «змінними поля» - тобто вони можуть змінюватись у просторі і часі. Наприклад, вбудована змінна густини `solid.rho` в інтерфейсі `Solid Mechanics`, являє собою функцію просторових координат і інколи часу. Ви можете візуалізувати її на ділянці поверхні або об’єму, але не можете використовувати в глобальних обчисленнях навіть якщо вона є сталою. Наприклад цю змінну при наявності оператора інтегрування `intop1` по області, можна використати у виразі `intop1(1*solid.rho)` для визначення скалярної змінної маси. Але вираз `intop1(1)*solid.rho` не визначатиме скалярну змінну – він визначатиме іншу полеву змінну, яка дорівнює густині, помноженій на об’єм. Звичайно, що набір польових змінних залежить від діючого прикладного режиму.

Існують вбудовані змінні, доступні для використання незалежно від обраного інтерфейсу. Серед них є такі, що пов'язані з геометричними властивостями об'єктів моделі. До них, наприклад, відносяться просторові координати `x`, `y`, `z`. Іншим прикладом є ідентифікатори `nx`, `ny`, `nz`, які завжди позначають компоненти вектора нормалі $\boldsymbol{n}$ до межі розрахункової області. Константи і геометричні змінні доступні завжди.

Перейдемо до більш докладного розгляду цих категорій (типів) змінних.

7.2. Геометричні і службові змінні.

В цьому параграфі будуть розглянуті основні вбудовані змінні, які характеризують властивості елементів геометрії і СкЕ сітки.

Серед змінних, які доступні для використання незалежно від обраного прикладного режиму, є змінні, пов'язані з геометрією моделі. Окрім просторових координат (x,y,z в 1D, 2D, 3D моделях або r, phi,z в Axisymmetric) можна застосовувати додаткові змінні, які визначені в граничних точках областей. Наприклад, коли ви працюєте в моделі з точками межі двовимірної області, то крім просторових координат x, y, ви можете використовувати ім'я s, яке представляє параметр кривої Безьє в рівнянні кожної ділянки межі, і де він змінюється від 0 до 1. Крім того, в граничних точках визначені одиничний дотичний $\boldsymbol{t}=\left(t_x,t_y\right)$ і нормальний $\boldsymbol{n}=\left(n_x,n_y\right)$ вектори. Ви можете використовувати символи tx, ty, nx, ny в виразах, які «працюють» на межах двовимірних областей. Щоб керувати напрямком нормалі всередину і назовні області, передбачені змінні dnx, dny, unx і uny. Вони є координатами вектора внутрішньої **un**=(unx,uny) і зовнішньої **dn**=(dnx,dny) нормалі (точніше **u**p "вгору" і **d**own "вниз"). У двовимірних моделях напрямок вліво щодо додатного напрямку кривої називається напрямком «вгору», протилежний - напрямком «вниз». Додатним напрямком руху вздовж кривої є напрям зростання параметра s. Він показується стрілками, якщо у вузлі **View** встановлено перемикач **Show edge direction arrows** і у графічному вікні обрано режим Select Boundaries (над графічним вікном натиснута кнопка Select Boundaries).

Для поверхонь тривимірних тіл визначені аналогічні змінні. Так змінні s1 і s2. доступні у всіх виразах, які відносяться до поверхонь. Крім того, на поверхнях визначені змінні - координати двох дотичних векторів $\boldsymbol{t}_1,\boldsymbol{t}_2$ і вектор нормалі $\boldsymbol{n}$. Компоненти вектора нормалі до граничної поверхні мають імена nx, ny, nz. Також передбачені змінні dnx, dny, dnz, unx, uny і unz, де префікси **u** і **d** позначають напрямок "вгору" і "вниз" щодо граничної поверхні.

На кінцях одновимірних областей (тобто в граничних точках відрізків) визначаються змінні nx, dnx і unx.

Приклад 1. Використання службових змінних і геометричних змінних, пов'язаних з межею двовимірної моделі.

Розв'язання. Створіть порожню модель.

File->New -> Blank Model

Додайте в модель двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть в моделі перевірку розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

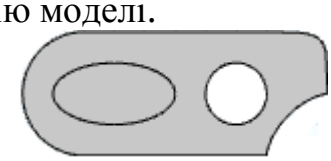
Встановіть прапорці, які вмикають відображення позначок геометричних об’єктів і напрямків зростання параметрів вздовж кривих.

Component 1 – Definitions - View 1:

View – **Show geometry labels** = True

– **Show edge direction arrows** = True

Побудуйте наступну геометрію моделі.



Geometry -> **Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1 (r1):

Size – **Width, Height** = 6,2 (*два поля*)

Position – **Base, x, y** = Corner,1,0 (*три поля*)

LMB > Build Selected

Geometry -> **Primitives** – Circle

Component 1 - Geometry 1 - Circle 1(c1):

Size and Shape – **Radius:** = 1

Position – **Base, x, y** = Center,1,1

LMB > Build Selected

Geometry -> **Booleans and Partitions** – Union

Component 1 - Geometry 1 - Union 1 (uni1):

Union – **Input Objects** = r1,c1 (*прямокутник і коло*)

– **Keep interior boundaries** = False

LMB > Build Selected

Geometry -> **Primitives** – Ellipse

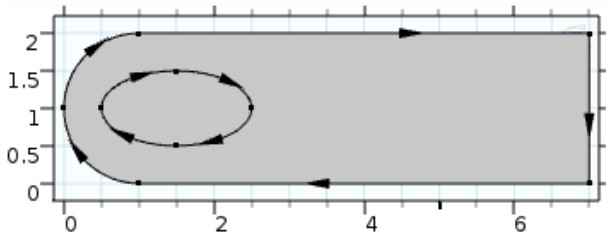
Component 1 - Geometry 1 - Ellipse 1(e1):

Size and Shape – **a-semiaxis** = 1

Size and Shape – **b-semiaxis** = 0.5

Position – **Base, x, y** = Center,1.5,1

LMB > Build Selected



Geometry -> **Primitives** – Circle

Component 1 - Geometry 1 - Circle 2(c2):

Size and Shape – **Radius:** = 0.5

Position – **Base, x, y** = Center,3.5,1

LMB > Build Selected

Geometry->**Primitives->** Parametric Curve

Geometry 1 - Parametric Curve 1 (pc1) :

Parameter – **Name, Minimum, Maximum** = s,-pi/2, 0

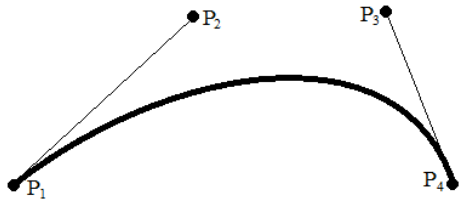
Expressions – **x, y** = 4+cos(s)^3, 1+sin(s)^3 (*дуга астроїди*)

LMB > Build Selected

Geometry -> **Pimitives** – Bézier Polygon

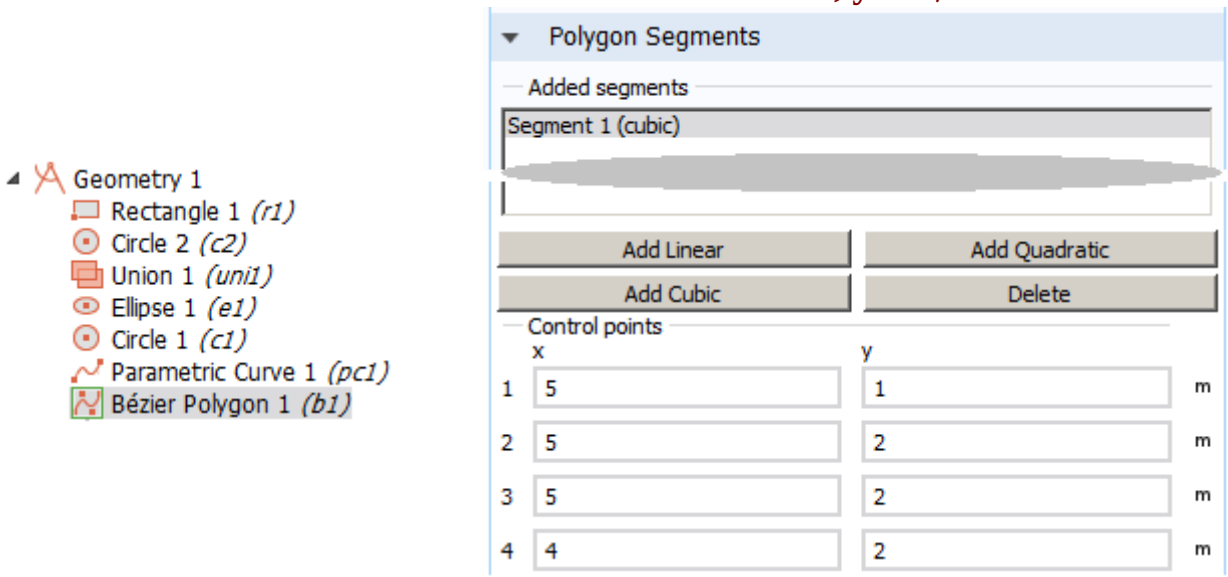
Bézier Polygon 1 : LMB > Add Cubic

Кубічний сегмент Безьє задається положенням чотирьох точок. Перша і остання являють кінцеві точки сегменту, а друга і третя, окрім іншого, визначають напрямок дотичних.

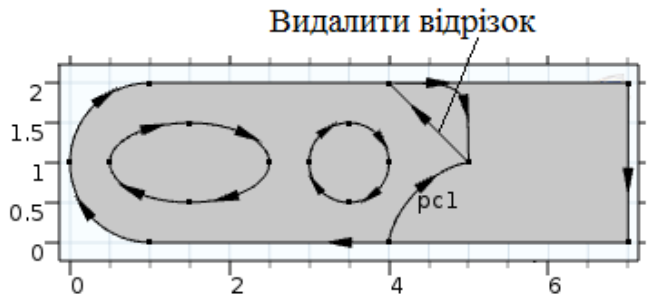


В нашому прикладі друга і третя точки співпадатимуть.

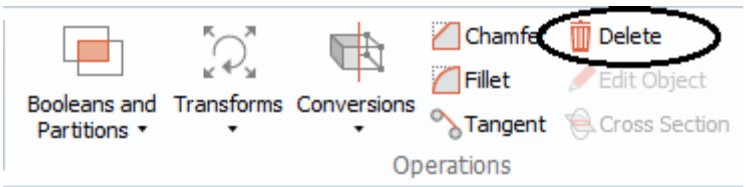
Polygon Segments – **Segment 1 (cubic)** – **1: x, y** = 5,1
– **2: x, y** = 5,2
– **3: x, y** = 5,2
– **4: x, y** = 4,2



LMB > Build Selected



Видаліть зайвий відрізок, який показано на попередньому рисунку. Для цієї операції призначений «інструмент» Delete, який розташований на панелі Geometry в розділі **Operations**.

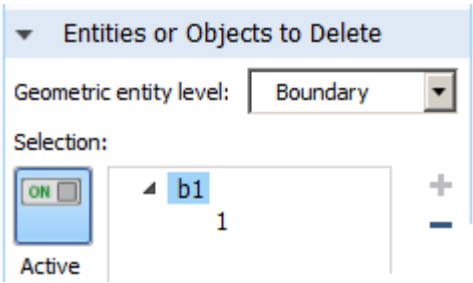


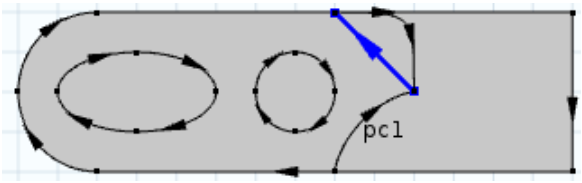
Geometry **-> Operations** – Delete

В ДМ з'явиться вузол **Delete Entities 1 (del1)**. В його секції вибору спочатку потрібно визначити рівень геометрії об'єктів, які будуть усуватися (точки, лінії або підобласті), а потім вказати (клацнувши мишею) об'єкти, які будуть видалятися.

Delete Entities 1 (del1):

Entities or Objects to Delete – **Geometric entity level**=Boundary
– **Selectin :** = b1-1





LMB > Build Selected

З геометрії віднімiть коло.

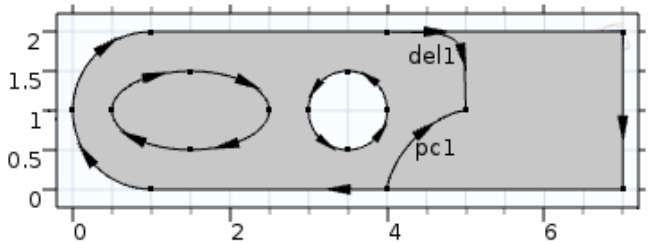
Geometry **-> Booleans and Partitions** – Difference

Component 1 - Geometry 1 - Difference 1 (dif1):

Difference – **Object to add** = uni1

Union – **Object to subtract** = c2 *(коло)*

LMB > Build Selected



Наступним кроком буде видалення правої частини зони, відокремленої кривими pc1 і del1. Для цього знадобиться інструмент «розбиття об'єктів» (Partition Objects), який створює Partition Objects вузол. Він реалізує розділення геометричних об'єктів на окремі частини. В цьому вузлі вказується розділюваний об'єкт (наприклад, область на площині) і геометричні об'єкти, які застосовуються в якості розрізу (наприклад, набір кривих). Результатом є частини первинного об'єкта, що в поєднанні являють початковий об'єкт.

Geometry **-> Booleans and Partitions** – Partition Objects

Partition Objects 1 (par1):

Partition Objects – **Objects to partition** = dif1

Partition Objects – **Tool objects** = pc1, del1

LMB > Build Selected

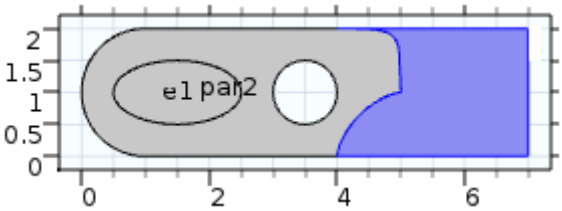
Після застосування операції Partition Objects, права частина області буде відокремлена (ви цього зразу не побачите) від лівої. Але на наступному кроці в операції видалення ви вже зможете її виділяти.

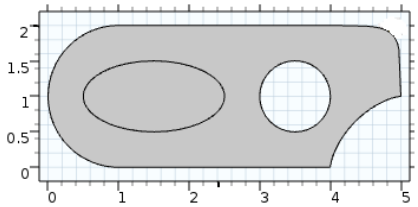
Geometry **-> Operations** – Delete

Delete Entities 2 (del2):

Entities or Objects to Delete – **Geometric entity level** = Domain
– **Selectin :** = par1-2

LMB > Build All Objects





Генеруйте СкЕ сітку з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Щоб графікам було на що посилатися (існувало джерело даних), створіть і обчисліть вузол досліджень.

Untitled.mph (root): RMB > Add Study = Add Study ->
Preset Studies – Stationary -> X

Study 1: LMB > Compute

Тепер побудуйте над всіма кривими (зовнішнім контуром, внутрішніми колом і еліпсом) графік величини s. Вона є завжди доступною геометричною змінною, яку можна використовувати у виразах на кривих.

Results **-> Plot Group ->** 2D Plot Group

2D Plot Group 1:

2D Plot Group 1-> **Add Plot ->** Line

2D Plot Group 1 – Line 1:

За допомогою першого Line вузла ми намалюємо добре помітні контури кривих розрахункової геометрії. Якщо джерелом даних вузла типу Line є весь розв'язок, тобто Study 1/Solution 1, то автоматично з цього ДД будуть вибиратися всі криві, які створені в геометрії..

Expression – **Expression** = 0

Coloring and Style – **Line type** = Tube
– **Tube radius expression** = 0.05
– **Radius scale factor** = True,1
– **Coloring, Color** = Uniform, Black

В другому Line вузлі побудуйте графіки величини s над кривими.

2D Plot Group 1-> **Add Plot ->** Line

2D Plot Group 1 – Line 2:

Expression – **Expression** = s

Coloring and Style – **Line type** = Tube
– **Tube radius expression** = 0.03
– **Radius scale factor** = True,1
– **Coloring, Color** = Color table, Rainbow

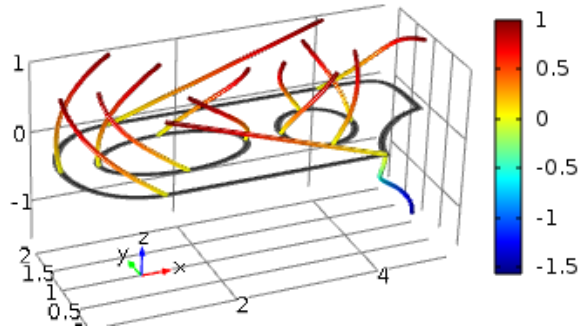
RMB> Height Expression

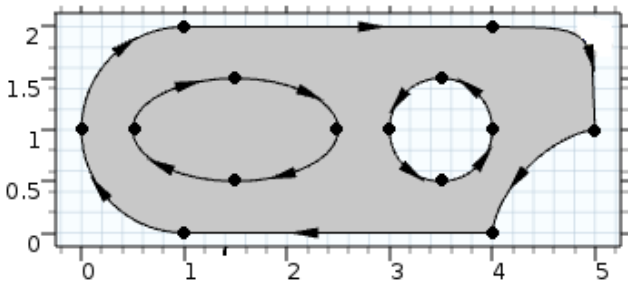
2D Plot Group 1 – Line 1 – Height Expression 1:

Axis – Scale factor = True,1

LMB>Plot (*наступний рисунок ліворуч*)

Для порівняння праворуч ми навели зображення розрахункової геометрії з напрямками зростання параметра s і точками, які розділяють прилеглі криві.





На рисунку ліворуч жовтий колір графіків відповідає близьким до нуля значенням параметра s. З графіка видно, що еліпс і коло розділені на чотири частини, вздовж кожної з яких параметр s змінюється від 0 до 1. На дузі астроїди ім'я параметра s створено нами штучно і він змінюється від $-\pi/2$ до 0. Отже графік над цією частиною контура розташований у від'ємній зоні. Сегмент Безьє теж параметризований змінною s з діапазону [0,1].

Зауваження 1. При створенні параметрично заданих кривих і поверхонь задаються змінні їх параметризації. В поточному прикладі було створено дугу астроїди, як параметричну криву з параметром s. Ім'я параметра можна обирати довільно (не обов'язково s). Такі змінні нарівні з вбудованими геометричними змінними можна застосовувати у виразах, що відносяться до цих кривих.

Зауваження 2. Параметр s можна застосовувати в будь-якому місці моделі, але лише на кривих, зокрема, його можна використати при заданні граничних умов.

Зауваження 3. В 2D моделях замість змінної s можна використовувати ім'я s1, успадковане від тривимірних позначень.

В цій же графічній групі на кривих побудуйте векторне поле $\boldsymbol{n} = (\mathrm{nx}, \mathrm{ny})$. В 2D моделях зазвичай (але не завжди) вектор $\boldsymbol{n}$ вказує назовні області.

2D Plot Group 1-> Add Plot – Arrow Line

2D Plot Group 1 - Arrow Line 1 :

Expression – x component, y component = nx,ny (*два поля*)

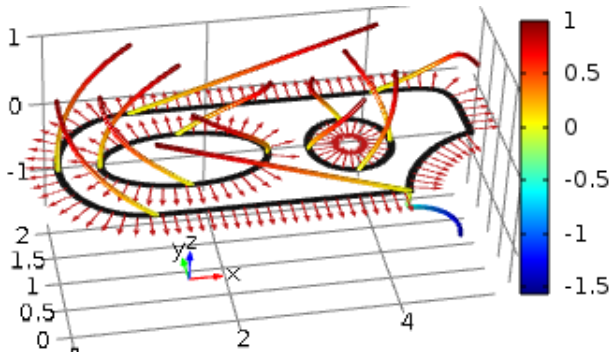
Coloring and Style – Scale factor = True,0.4

– **Placement** =Uniform

– **Number of arrows** =150

– **Color** =Red

LMB>Plot

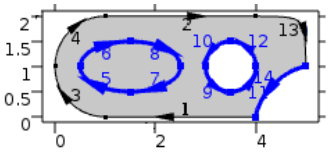


Вектори поля `dnx,dny` мають ту ж орієнтацію, а `unx,uny` – протилежну (перевірте самостійно).

В двовимірних моделях для ліній завжди визначена змінна `curv` (кривина k). Вона має знак, що залежить від напрямку кривої (вважається, що додатне значення спрямоване в бік нормалі $\boldsymbol{n} = (\mathrm{nx}, \mathrm{ny})$). Величина обернена до кривини $R = 1/k$ має сенс радіуса стичного кола (радіус кривини).

В нашому прикладі побудуємо графік залежності радіуса кривини вздовж еліпса, кола і дуги астроїди. Але спочатку створіть джерело даних, яке складається з цих дуг.

`Results` **->** More Data Sets – Edge 2D



Data Sets - Edge 2D 1 :

`Data` – **Data set** = `Study 1/Solution 1`

`Selection` – **Selection** = `Manual,`
`5,6,7,8,9,10,11,12,14`

Тепер будуйте графік радіуса кривини над цими лініями.

2D Plot Group 1: RMB > Duplicate

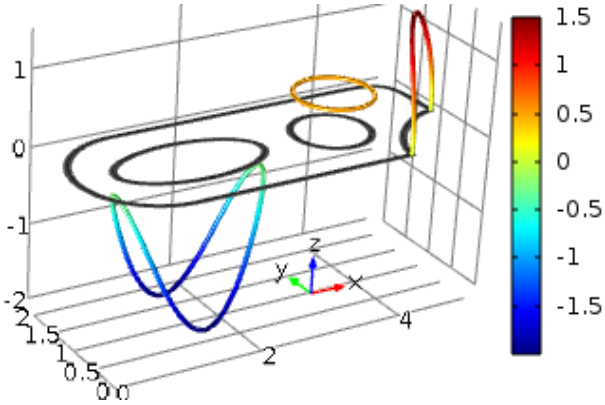
2D Plot Group 2 - Arrow Line 1: RMB > Delete

2D Plot Group 2 – Line 2:

`Data` – **Data set** = `Edge 2D 1`

`Expression` – **Expression** = `1/curv`

LMB>Plot



Оскільки кривина кола стала, то і графік його радіуса кривини розташований на фіксованій висоті над цим колом. Радіус кривини еліпса від'ємний, оскільки напрямок його обходу протилежний напрямку на колі.

В COMSOL Multiphysics завжди визначена службова змінна `dom`, яка містить індикатори (внутрішній номер) геометричних об'єктів (точок, ділянок кривих і поверхонь, підобластей). Вона може приймати тільки цілочисельні

значення. Для тренування побудуємо її графік над розрахунковою областю і над граничними кривими.

2D Plot Group 1: RMB > Duplicate

2D Plot Group 3 - Arrow Line 1: RMB > Delete

2D Plot Group 3 - Line 2: RMB > Delete

2D Plot Group 3-> **Add Plot ->** Contour

2D Plot Group 3 – Contour 1:

Expression – **Expression** = dom

Levels – **Entry method; Levels** = Levels;1,2

Coloring and Style – **Contour type** = Filled

RMB> Height Expression

2D Plot Group 3 – Line 1 – Height Expression 1:

Axis – **Scale factor** = True,0.5

LMB>Plot (*наступний рисунок ліворуч*)

2D Plot Group 1: RMB > Duplicate

2D Plot Group 4 - Arrow Line 1: RMB > Delete

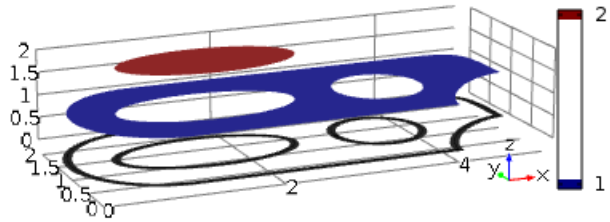
2D Plot Group 4– Line 2:

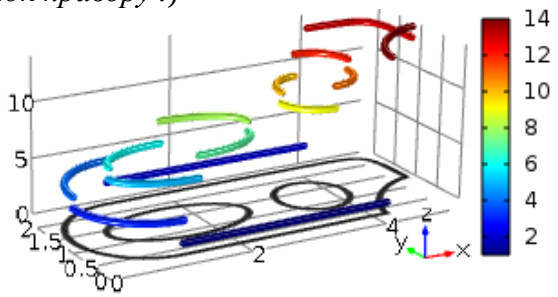
Expression – **Expression** = dom

2D Plot Group 4 – Line 2 – Height Expression 1:

Axis – **Scale factor** = True,0.15

LMB>Plot (*наступний рисунок праворуч*)





Щоб дізнатися значення змінної dom конкретних точок, його можна надрукувати у вузлі Point Evaluation, обравши точки в секції **Selection** цього вузла (зробіть це самостійно).

Є ще допоміжна змінна h. Вона доступна тільки в областях і представляє в двовимірних моделях максимальний розмір сторін трикутних елементів і максимальний розмір ребер тетраедрів в тривимірних моделях. Її теж можна візуалізувати.

2D Plot Group 3: RMB > Duplicate

2D Plot Group 5 - Contour 1: RMB > Delete

2D Plot Group 5-> **Add Plot ->** Surface

2D Plot Group 5 - Surface 1:

Expression – **Expression** = h

Quality – **Smoothing** = None

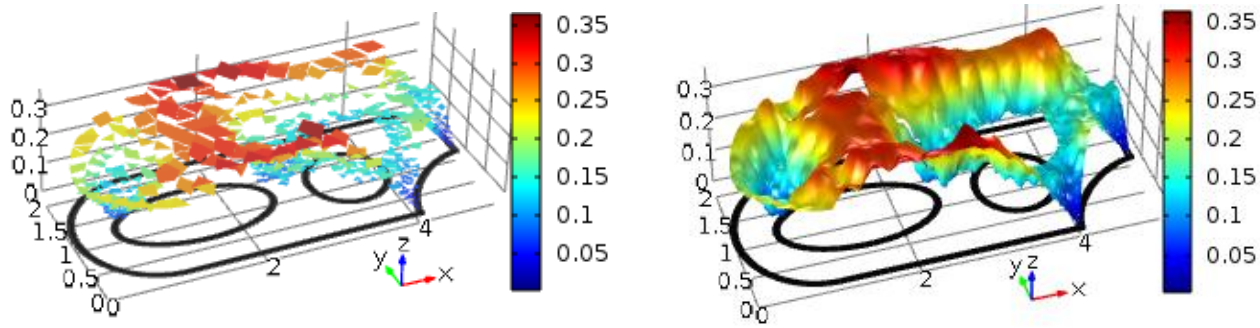
Оскільки змінна h має стале значення в кожному скінченному елементі то, щоб не було згладжування графіка між сусідніми елементами, знадобилося відключити опцію **Smoothing**.

RMB> Height Expression

2D Plot Group 5– Surface 1 – Height Expression 1:

`Axis` – Scale factor = `True,5`

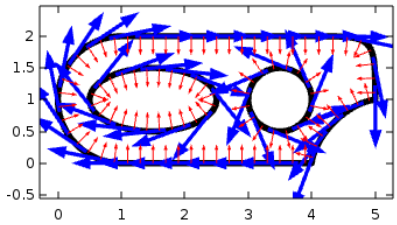
LMB>Plot (*наступний рисунок ліворуч*)



Графік, наведений праворуч, отримано при ввімкненій опції **Smoothing**= `Internal`.

Зауваження. Щоб не було конфлікту імен, не застосовуйте до змінних користувача ім'я `h`.

Вправа. На контурах області з поточного прикладу побудуйте векторне поле одиничних дотичних векторів (`tx`,`ty`) і поле внутрішніх нормалей (`unx`,`uny`).



Приклад 2. Побудувати еквідистанту і еволюту кривої на площині. В заданій точці (x_0, y_0) зобразити дотичний і нормальний одиничні вектори.

Еволютою називають геометричне місце центрів кривини, побудованих для всіляких точок кривої. В якості кривої оберемо еліпс. Схожу задачу ми розглядали в п. 2.3.7. Але зараз, побудувавши в геометрії іншу криву, ви зможете побудувати її еволюту, не виконуючи ніяких додаткових обчислень. Для цього застосовуватимуться вбудовані змінні `curv`,`tx`,`ty`,`nx`,`ny`, які в COMSOL моделях визначаються автоматично.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Додайте в модель двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Встановіть прапорці, які вмикають відображення позначок геометричних об'єктів і напрямків зростання параметрів вздовж кривих.

Component 1 – Definitions - View 1:

`View` – Show geometry labels = `True`

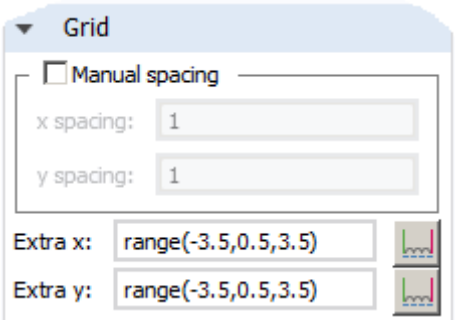
– Show edge direction arrows = `True`

У дочірньому підвузлі **View 1 – Axis** задайте положення додаткових зарубок на координатних осях.

Component 1 – Definitions - View 1 – Axis:

Grid – **Extra x:** = `range(-3.5,0.5,3.5)`

– **Extra y:** = `range(-3.5,0.5,3.5)`



Побудуйте геометрію.

`Geometry->`**`Primitives->`** Square

Geometry 1 - Square 1 (sq1):

Size – **Size length** = `7`

Position – **Base, x, y** = `Center, 0, 0`

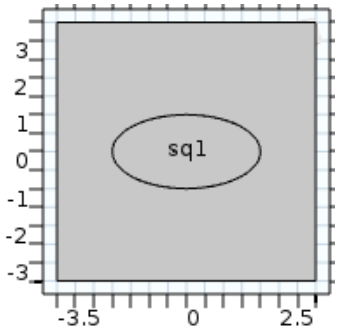
`Geometry->`**`Primitives->`** Ellipse

Component 1 - Geometry 1 - Ellipse 1(e1):

Size and Shape – **a-semiaxis, b-semiaxis** = `2,1`

Position – **Base, x, y** = `Center,0,0`

LMB > Build All Objects



Створіть змінні – координати точки на еліпсі, координати дотичного і нормального векторів в цій точці.

`Definitions` **`->Variables`** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = `x0,1`

Variables – **Name, Expression** = `y0, sqrt(3)/2`

Variables – **Name, Expression** = `Lx, at1(x0,y0,tx)`

Variables – **Name, Expression** = `Ly, at1(x0,y0,ty)`

Variables – **Name, Expression** = `Nx, at1(x0,y0,nx)`

Variables – **Name, Expression** = `Ny, at1(x0,y0,ny)`

Рівнем дії залишимо всю модель, але насправді формули типа `at1(x0,y0,tx)` (отже і змінні `Lx,Ly,Nx,Ny`) можна буде застосовувати лише в точках еліпса.

Генеруйте СкЕ сітку з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Щоб графікам було на що спиратися (існувало джерело даних), створіть і обчисліть вузол досліджень.

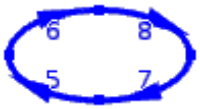
Untitled.mph (root): RMB > Add Study = `Add Study` ->
Preset Studies – Stationary -> X

Study 1: LMB > Compute

Побудуйте джерело даних, яке містить лише точки еліпса.

`Results` -> More Data Sets – Edge 2D

Data Sets - Edge 2D 1 :



`Data` – **Data set** = `Study 1/Solution 1`

`Selection` – **Selection** = `Manual,5,6,7,8`

Створіть графічну групу, в якій почніть по черзі будувати потрібні графіки.

`Results` -> **`Plot Group`** -> 2D Plot Group

2D Plot Group 1:

`2D Plot Group 1`-> **`Add Plot`** -> Line

У створеному Line вузлі побудуйте графік еволюти.

2D Plot Group 1 – Line 1:

`Data` – **Data set** = `Edge 2D 1`

`Expression` – **Expression** = `0`

`Coloring and Style` – **Line type** = `Tube`

– **Tube radius expression** = `0.03`

– **Radius scale factor** = `True,1`

– **Coloring, Color** = `Uniform, Magenta`

RMB> Deformation

2D Plot Group 1 – Line 1 – Deformation 1:

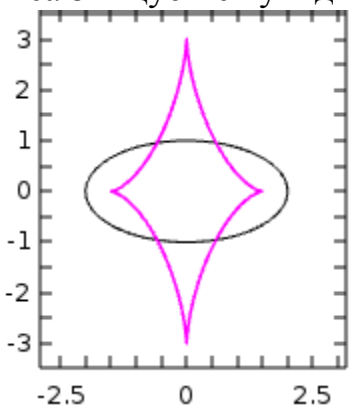
`Expression` – **x component** = `nx/curv`

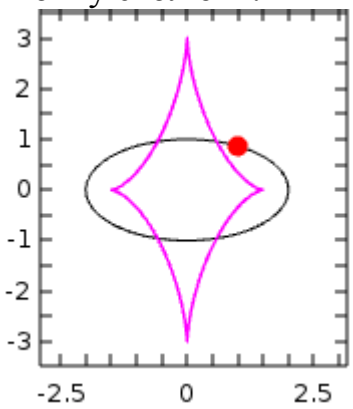
`Expression` – **y component** = `ny/curv`

`Scale`– **Scale factor** = `True,1`

LMB>Plot (*наступний рисунок ліворуч*)

Тут для побудови центрів кривини використано вузол `Deformation`. Фактично, кожна точка входової кривої зміщується на вектор $\boldsymbol{n}\cdot R$, де $\boldsymbol{n}$=(nx,ny) – одиничний вектор нормалі, а R=1/curv – радіус кривини. Внаслідок цього кожна точка еліпса зміщується у відповідну точку еволюти.





Домалюйте точку дотику, в якій зображатимуться дотичний і нормальний вектори. Це можна зробити за допомогою вузла `Point Trajectories`, відключивши в ньому відображення кривої і залишивши відображення точки.

`2D Plot Group 1` -> Add Plot - More Plots - Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

Trajectory Data – x-expression, y-expression = x0,y0

Coloring and Style – Line style –Type = None

Останнім відключається відображення кривої. Наступні інструкції налаштовують зображення точки дотику.

Coloring and Style – Point style –Type = Point

– Point style –Point radius expression = 0.1

– Point style – Radius scale factor = True,1

– Point style – Color = Red

LMB > Plot (*попередній рисунок праворуч*)

Для зображення окремого вектора в системі не існує відповідного інструмента. Та можна застосувати вузли типу Arrow Line (або Arrow Surface), які будують поле векторів вздовж кривої (або поверхні), а кількість стрілок векторного поля вказується в налаштуваннях вузла. Завжди можна наказати будувати лише одну стрілку поля. Але тут виникають проблеми – де буде намальована ця стрілка. Зазвичай її зображення нам потрібно в деякій точці. Коли стрілок багато, то вони розподіляються рівномірно вздовж ДД відповідного вузла (Arrow Line або Arrow Surface). Якщо **Number of arrows**=1, то стрілка малюється в центрі симетрії джерела даних. Отже перш, ніж зображати стрілку, нам знадобиться ДД, центр симетрії якого буде розташовано в точці дотику. Це може бути короткий відрізок дотичної, заданої параметричним рівнянням. Створіть таке ДД для дотичного відрізка.

Results – Data Sets: RMB > More Data Sets - Parameterized Curve 2D

Data Sets – Parameterized Curve 2D 1:

Parameter – Name, Minimum, Maximum = tau,-0.5,0.5

Expressions – x:, y: = x0+tau*Lx, y0+tau*Ly

Останній рядок задає параметричне рівняння дотичного відрізка, а попередня інструкція задає невеликий діапазон зміни параметра tau, визначаючи довжину відрізка дотичної. Довжина не має значення, але важливо, щоб діапазон зміни параметра tau був симетричним.

Тепер можна малювати вектори.

2D Plot Group 1-> Add Plot – Arrow Line

2D Plot Group 1 - Arrow Line 1 :

Data – Data set = Parameterized Curve 2D 1

Expression – x component, y component = Nx,Ny

Coloring and Style – Scale factor = True,1.5

– Placement =Uniform

– Number of arrows =1

– Color =Red

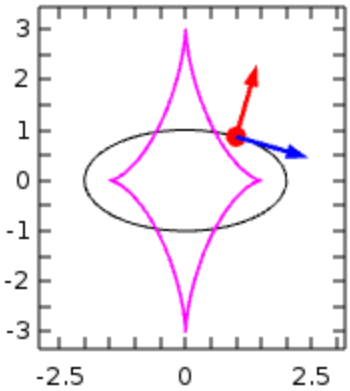
2D Plot Group 1 - Arrow Line 1 : RMB > Duplicate

2D Plot Group 1 - Arrow Line 2 :

Expression – x component, y component = Lx,Ly

Coloring and Style – Color =Blue

LMB>Plot



Застосовуючи геометричні змінні nx, ny, можна будувати еквідистантні криві (криві, які розташовані на фіксованій відстані одна від одної в напрямку нормалі). Для цього достатньо у вузлі, який зображує еліпс, задати деформацію на вектор $\pm \boldsymbol{n} \cdot H$, де $\boldsymbol{n}$ – одиничний вектор нормалі, а H – відстань між кривими.

Побудуйте еквідистанти нашого еліпса. Для наочності будуватимемо також векторне поле нормалей вздовж нього.

Results **-> Plot Group ->** 2D Plot Group

2D Plot Group 2:

Data – **Data set** = Edge 2D 1

2D Plot Group 2-> Add Plot – Arrow Line

2D Plot Group 2 - Arrow Line 1 :

Expression – **x component, y component** = nx,ny

Coloring and Style – **Scale factor** = True,1

– **Placement** =Uniform

– **Number of arrows** =40

– **Color** =Blue

2D Plot Group 2-> **Add Plot ->** Line

2D Plot Group 2 – Line 1:

Expression – **Expression** = 0

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.03

– **Radius scale factor** = True,1

– **Coloring, Color** = Uniform, Red

RMB> Deformation

2D Plot Group 2 – Line 1 – Deformation 1:

Expression – **x component, y component** = nx,ny (*H=1*)

Scale – **Scale factor** = True,1

LMB>Plot (*наступний рисунок ліворуч*)

Тепер побудуйте внутрішню еквідистанту з відступом на 0.6 одиниць.

2D Plot Group 2 - Arrow Line 1 : RMB > Duplicate

2D Plot Group 2 - Arrow Line 2 :

Expression – **x component, y component** = -nx*0.6,-ny*0.6

Coloring and Style – **Color** =Magenta

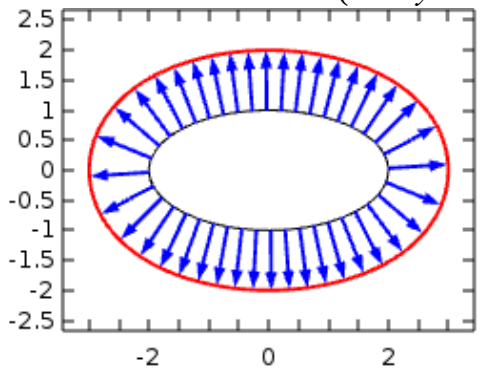
2D Plot Group 2 - Line 1 : RMB > Duplicate

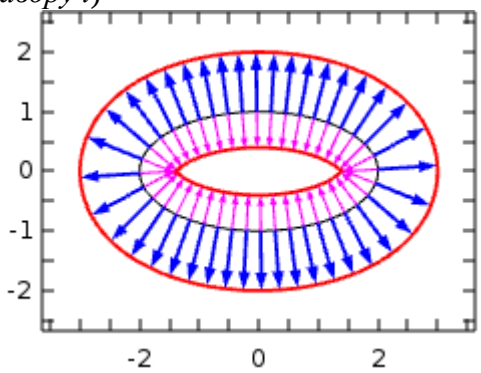
2D Plot Group 2 – Line 2 – Deformation 1:

Expression – x component = `-nx*0.6`

Expression – y component = `-ny*0.6`

LMB>Plot (*наступний рисунок праворуч*)

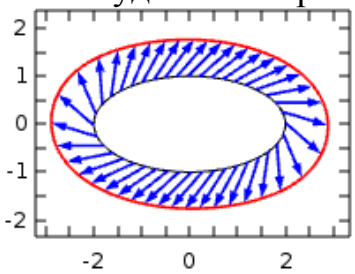




Зауваження 1. Застосований спосіб побудови еквідистант працює лише для гладких кривих і «правильно» обраних відстаней *H*. Наприклад, при наявності на кривій гострих кутів ділянки відрізків «зовнішніх» еквідистант, які їм відповідають, повинні бути дугами кіл, але вони системою не будуються. Навпаки, «внутрішні» еквідистанти, які відповідають гострим вершинам, повинні мати гострі кути, але це не спостерігається (перевірте самостійно).

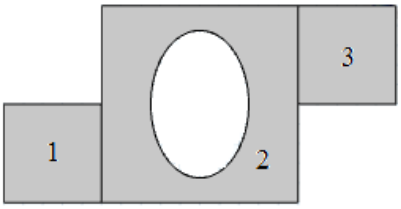
Зауваження 2. Еквідистанти вздовж нормалі можна замінити «еквідистантами» вздовж дотичного напрямку, або будь якого іншого, наприклад, вздовж напрямку вектора $\boldsymbol{l} = \alpha \boldsymbol{n} + \beta \boldsymbol{t}$, де $\boldsymbol{n}$ і $\boldsymbol{t}$ – нормальний і дотичний вектори.

Вправа. Побудуйте «еквідистанту» вздовж вектора $\boldsymbol{l} = \boldsymbol{n} + \boldsymbol{t}$, розташовану зовні еліпса на одиничній відстані. Не забудьте вектор $\boldsymbol{l}$ нормувати.

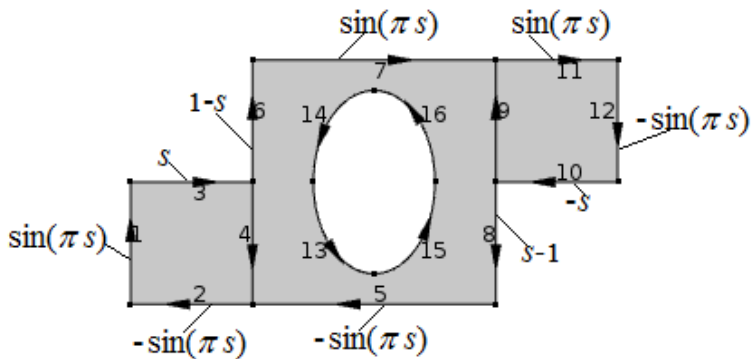


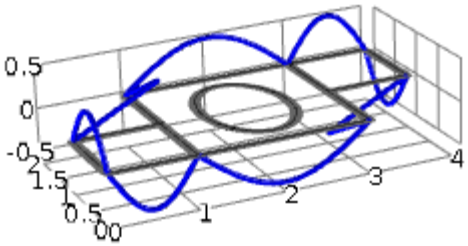
Приклад 3. Застосування геометричних і службових змінних в умовах стаціонарної задачі дифузії.

Розв'язати стаціонарне рівняння дифузії в двовимірній області з наступною геометрією.



Коефіцієнт дифузії сталий. В зонах 1 і 3 діють джерела дифундуючої речовини зі сталою потужністю Q_0. В зоні 2 діє внутрішнє джерело з потужністю $\alpha \cdot u$, пропорційною концентрації u дифундуючої речовини. На зовнішній межі задана концентрація u_0, яка розподілена наступним чином





Графік концентрації на зовнішній межі.

де s - параметр ($0 \le s \le 1$) вздовж відповідних відрізків зовнішньої межі. На контурі еліптичного отвору потік дифундуючої речовини дорівнює $q = q_0 \cdot k$, де k – кривина еліпса.

Отже потрібно розв'язати наступну задачу.

$$\nabla \cdot (-c \nabla u) + \alpha u = Q,$$

де

$$\alpha = \begin{cases} 0 & \text{в зонах 1 і 3} \\ \alpha_0 & \text{в зоні 2} \end{cases}, \quad Q = \begin{cases} Q_0 & \text{в зонах 1 і 3} \\ 0 & \text{в зоні 2} \end{cases},$$

з граничними умовами

$$u\big|_{\text{на зовнішній межі}} = u_0,$$

$$-c \frac{\partial u}{\partial n}\bigg|_{\text{на внутрішній межі}} = q_0 k \quad (k - \text{кривина еліпса}).$$

В прикладі оберіть $c = 1, \alpha_0 = 1, Q_0 = 0.5, \; q_0 = 0.2$.

Це штучний приклад, призначений для демонстрування можливостей застосування геометричних і службових змінних COMSOL Multiphysics.

Р о з в ' я з а н н я . За допомогою майстра створіть 2D модель задачі.

File -> New -> Model Wizard -> 2D -> Mathematics – Classical PDEs – Helmholtz Equation (hzeq) -> Stationary

Вимкніть в моделі перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Встановіть прапорці, які вмикають відображення позначок геометричних об'єктів і напрямків зростання параметрів вздовж кривих.

Component 1 – Definitions - View 1:

`View` – **Show geometry labels** = True
– **Show edge direction arrows** = True

Задайте параметри моделі.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = a0, 1
– **Name, Expression** = Q0, 0.5
– **Name, Expression** = q0, 0.2

Побудуйте геометрію.

Geometry->**Primitives->** Square

Geometry 1 - Square 1 (sq1):

Size – **Size length** = 1

Position – **Base, x, y** = Corner, 0, 0

Geometry->**Primitives->** Square

Geometry 1 - Square 2 (sq2):

Size – **Size length** = 2

Position – **Base, x, y** = Corner, 1, 0

Geometry->**Primitives->** Square

Geometry 1 - Square 3 (sq3):

Size – **Size length** = 1

Position – **Base, x, y** = Corner, 3, 1

Geometry **->Booleans and Partitions** – Union

Component 1 - Geometry 1 - Union 1 (uni1):

Union – **Input Objects** = sq1,sq2,sq3

Geometry **->Primitives** – Ellipse

Component 1 - Geometry 1 - Ellipse 1(e1):

Size and Shape – **a-semiaxis** = 0.5

Size and Shape – **b-semiaxis** = 0.75

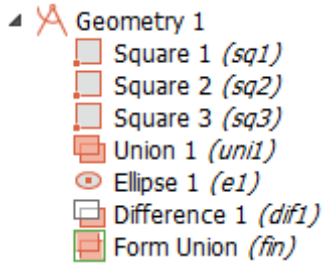
Position – **Base, x, y** = Center,2,1

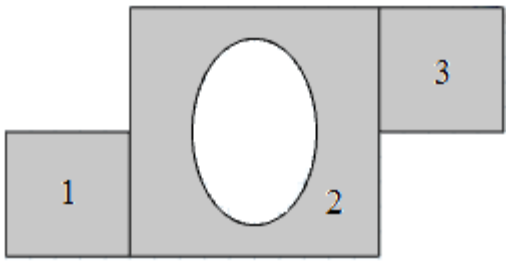
Component 1 - Geometry 1 - Difference 1 (dif1):

Difference – **Object to add** = uni1

Union – **Object to subtract** = e1 (*еліпс*)

LMB > Build All Objects





Генеруйте СкЕ сітку з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Для подальшої роботи буде зручно створити вибірку. Нагадаємо, що вузли вибірок (Selections) застосовуються для іменування множини геометричних об'єктів, яким дається спільне ім'я. Це ім'я використовують при заданні на всіх елементах групи однакової умови (наприклад, граничної). Вузли вибірок (Selections) створюються з випадного меню гілки Definitions командою Selections > спосіб вибору. Якщо виділення елементів буде виконуватися за допомогою інструментів вибору окремих геометричних сутностей (клацанням миші), то застосовується вибірка типу Explicit.

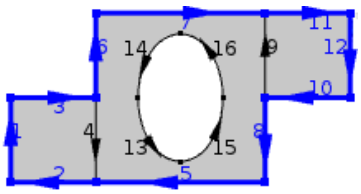
Component 1 – Definitions : RMB > Selections – Explicit

Definitions - Explicit 1 :

Label = SelOuterContour

Input Entities – **Geometric entity level** = Boudary

– **Active** = 1,2,3,5,6,7,8,10,11,12



Створіть також дві допоміжні змінні, які будуть визначені лише на побудованій вибірці SelOuterContour.

Definitions **->Variables** – Local Variables

Definitions – Variables 1:

Geometric Entity Selection – **Geometric entity level** = Boudary

– **Selection** = SelOuterContour

Variables – **Name, Expression** = vs,sin(pi*tx*s)*abs(y-1)+ sin(pi*ty*s)*(abs(x-2)-1)

Variables – **Name, Expression** = vd, (dom==3)*s+ (dom==6)*(1-s)-(dom==8)*(1-s)-(dom==10)*s

Перш ніж продовжувати моделювання задачі, проаналізуйте сенс створених змінних. Для цього побудуйте їх графіки на зовнішній межі розрахункової області. Але спочатку відкомпілюйте змінні, наприклад, командою

Study -> **Study** – Get Initial Value

Буде створено графічну групу **2D Plot Group 1**, яку ми перебудуємо.

2D Plot Group 1 – Surface 1: RMB > Delete

Для наочності за допомогою Line вузла намалюйте добре помітні контури кривих розрахункової геометрії.

2D Plot Group 1-> **Add Plot ->** Line

2D Plot Group 1 – Line 1:

Expression – **Expression** = 0

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.05

– **Radius scale factor** = True,1

– **Coloring, Color** = Uniform, Yellow

В другому Line вузлі над зовнішнім контуром фігури побудуйте графік змінної vs. Але спочатку створіть ДД відповідне кривим зовнішнього контуру.

Results **->** More Data Sets – Edge 2D

Data Sets - Edge 2D 1 :

Data – **Data set** = Study 1/Solution 1

Selection – **Selection** = SelOuterContour

Тут використана іменована вибірка.

Тепер в графічну групу **2D Plot Group 1** додайте і налаштуйте другий Line вузол.

2D Plot Group 1-> **Add Plot ->** Line

2D Plot Group 1 – Line 2:

Data – **Data set** = Edge 2D 1

Expression – **Expression** =vs

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = `0.03`
– **Radius scale factor** = `True,1`
– **Coloring, Color** = `Uniform, Black`

RMB > Height Expression

2D Plot Group 1 – Line 2 – Height Expression 1:

`Axis` – **Scale factor** = `True,1`

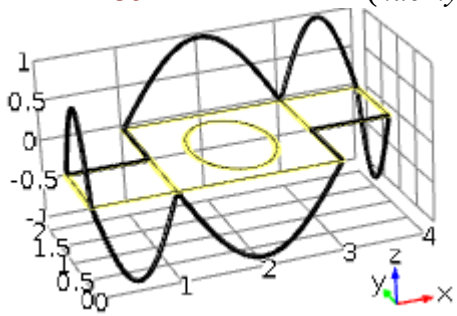
LMB>Plot (*наступний рисунок ліворуч*)

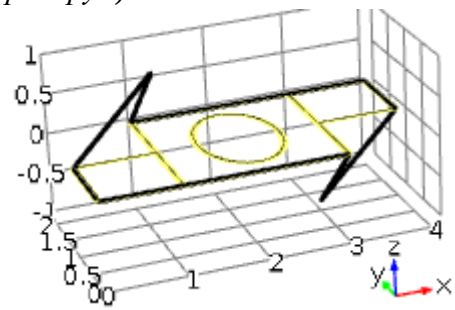
Замініть вираз на ім'я змінної `vd` і знову побудуйте графік.

2D Plot Group 1 – Line 2:

`Expression` – **Expression** =`vd`

LMB>Plot (*наступний рисунок праворуч*)





З графіків видно, що сума змінних `vs+vd` представлятиме на зовнішній межі граничну умову нашої задачі.

Проаналізуйте формули змінних `vs` і `vd`. В них використовуються декартові координати (`x`,`y`), параметр `s` вздовж відрізків контура, а також компоненти дотичного вектора `tx`, `ty` до відрізків межі. Наприклад, в змінній `vd` перевіряється значення індикатора відрізка, і якщо рівність виконується, то відповідний логічний вираз дорівнює одиниці, і потім множиться на вираз `s` або `1-s`. Коли рівність хибна, відповідний додаток дорівнює нулю.

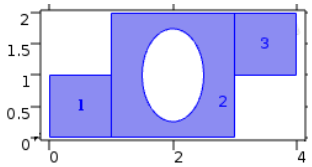
Налаштуйте фізичну постановку задачі.

Helmholtz Equation (hzeq) - Helmholtz Equation 1:

`Diffusion Coefficient` – **c** = `1`

`Source Term` – **f** = `(dom!=2)*Q0`

`Absorption Coefficient` – **a**=`-a0*(dom==2)`



Нагадаємо, що логічні вирази в арифметичних операціях мають значення 0 або 1 в залежності від їх хибності або істинності. Наприклад, вираз **f** буде дорівнювати Q_0 в зонах з індикатором 1 і 3.

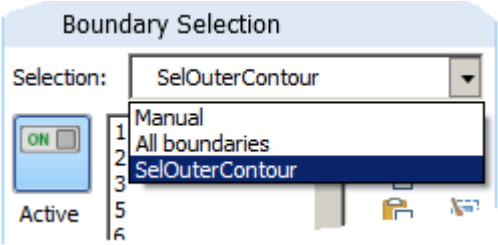
Задайте умову Діріхле на зовнішній межі.

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

`Boundary Selection` – **Selection** = `SelOuterContour`

Для виділення ділянок межі тут застосована іменована вибірка, ім'я якої присутнє у випадному списку **Selection**.



Dirichlet Boundary Condition – **r** =vs+vd

Задайте потік речовини на межі еліптичного отвору, який пропорційний кривині його контура.

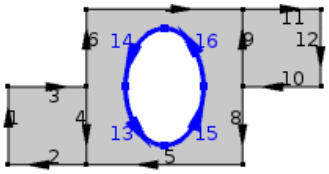
Physics -> **Boundaries** – Flux/Source

Helmholtz Equation (hzeq) - Flux/Source 1:

Boundary Selection – **Selection** = Manual,13,14,15,16

Boundary Flux/Source – **g** = -q0*curv

Boundary Absorption/Impedance Term – **q** =0



Нарешті виконайте обчислення моделі.

Study 1 : LMB > Compute

Після їх завершення створіть графічну групу, в якій побудуйте графік розв'язку.

Results **-> Plot Group ->** 2D Plot Group

2D Plot Group 2:

2D Plot Group 2-> **Add Plot ->** Surface

2D Plot Group 2 – Surface 1:

RMB> Height Expression

2D Plot Group 2 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = True,1

2D Plot Group 1 – Line 1: RMB > Copy

2D Plot Group 2: RMB > Past Line

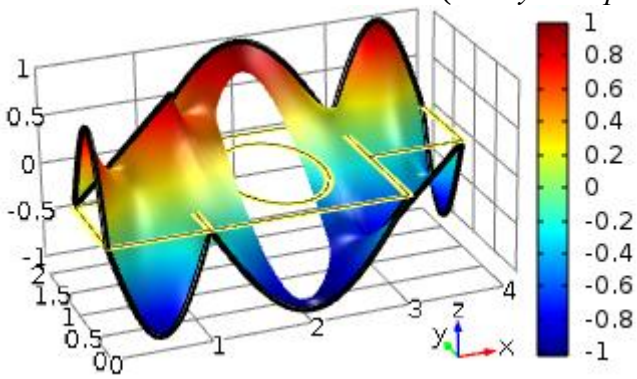
2D Plot Group 1 – Line 2: RMB > Copy

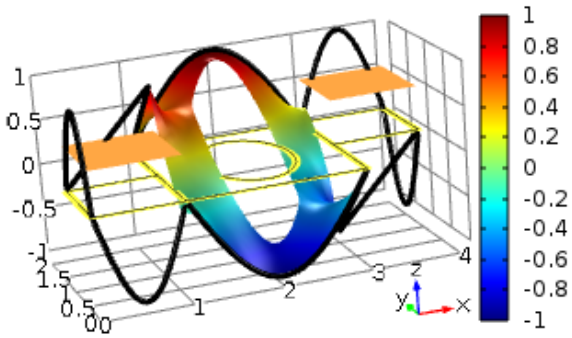
2D Plot Group 2: RMB > Past Line

2D Plot Group 2 – Line 2:

Expression – **Expression** =vs+vd

LMB>Plot (*наступний рисунок ліворуч*)





Для тренування, побудуйте графік густини внутрішніх джерел дифундуючої речовини. Нагадаємо, що в зонах з індикатором 1 і 3 вона стала, а в зоні 2 – пропорційна концентрації u.

2D Plot Group 2: RMB > Duplicate

2D Plot Group 3 – Surface 1:

Expression – **Expression** = (dom!=2)*Q0+a0*(dom==2)*u

LMB>Plot (*попередній рисунок праворуч*)

Зауваження. Між джерелами даних **Study 1/Solution 1** і **Edge 2D 1** спостерігається деяка різниця при використанні геометричних змінних в околі точок з різкою зміною напрямку границі. Щоб її побачити, збільшимо розміри скінченних елементів, а потім вздовж межі побудуємо графіки поля нормальних векторів ***n***=(nx,ny) на базі джерел даних **Study 1/Solution 1** і **Edge 2D 1**.

Mesh 1: Mesh Settings –**Element size** = Coarser

Study 1 : LMB > Compute

Results **-> Plot Group ->** 2D Plot Group

2D Plot Group 4:

2D Plot Group 4-> Add Plot – Mesh

2D Plot Group 4-> Add Plot – Arrow Line

2D Plot Group 4 - Arrow Line 1 :

Expression – **x component, y component** = nx,ny (*два поля*)

Coloring and Style – **Scale factor** = True,0.4

– **Placement** =Uniform

– **Number of arrows** =150

– **Color** =Red

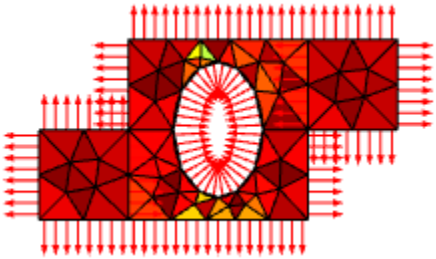
LMB>Plot (*наступний рисунок ліворуч*)

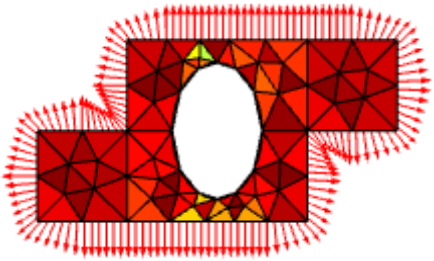
Тепер у вузлі **Arrow Line 1** змініть ДД.

2D Plot Group 4 - Arrow Line 1 :

Data – **Data set** = Edge 2D 1

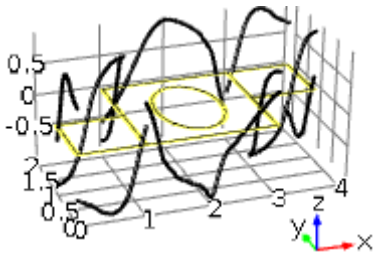
LMB>Plot (*наступний рисунок праворуч*)





Як бачите, в кутових точках поведінка векторного поля ***n***=(nx,ny) на ДД **Study 1/Solution 1** і **Edge 2D 1** відрізняється. Якщо кількість векторів велика, то над ДД **Edge 2D 1** їх зображення повертається в околі кутів, бо компоненти nx,ny в проміжних точках обчислюються по інтерполяції. Поворот виконується в границях кутових елементів, а у граничної вершини в точці стикування елементів, вірогідно, обчислюється середнє значення зліва і справа. Те ж саме спостерігається при побудові дотичних векторів ***t***=(tx,ty).

Вправа. Для розглянутої задачі побудуйте графік косинуса кута між градієнтом розв'язку u і нормаллю до межі тіла.



■

Розглянемо особливості застосування геометричних змінних в тривимірному випадку. Звичайно, що в 3D завжди визначені просторові координати `x,y,z` (або `r,phi,z`). Окрім них є вбудовані геометричні змінні, дійсні лише на межі тіл: на гранях і ребрах. На гранях (поверхнях) визначені змінні `s1,s2`. Їх можна вважати криволінійними координатами. Для «одновимірних» ребер визначена змінна `s1`, яка, на відміну від двовимірного випадку, приблизно еквівалентна довжині дуги.

На гранях/поверхнях визначені одиничні дотичні вектори $\boldsymbol{t}_1, \boldsymbol{t}_2$ і вектор нормалі $\boldsymbol{n}$. Їх компоненти позначаються `t1x,t1y,t1z,t2x,t2y,t2z,nx,ny,nz`. Вектори $\boldsymbol{t}_1, \boldsymbol{t}_2$ часто бувають ортогональними. Ортогональність може порушуватися при масштабуванні геометричних об'єктів. Для ребер змінні `t1x,t1y,t1z` позначають одиничний дотичний вектор до кривої. Коли використовуються внутрішні межі, то для розрізнення напрямку нормалі зручно використовувати вектори $\boldsymbol{un}$ і $\boldsymbol{dn}$ з компонентами `dnx,dny,dnz,unx,uny,unz`. Префікси **u** і **d** позначають «вгору» і «вниз» граничної поверхні. Напрямок «вгору» визначається як $\boldsymbol{un} = \boldsymbol{t}_1 \times \boldsymbol{t}_2$, а $\boldsymbol{dnx} = -\boldsymbol{unx}$.

В 3D геометріях на поверхнях в кожній неособливій точці визначені дві кривини: `curv1` і `curv2`, де `curv1` менша за `curv2`. Вони відповідають мінімальній і максимальній кривинам кривих, які утворюються при пересіченні поверхні нормальними площинами. Додатний напрямок кривини вказує нормаль $\boldsymbol{n} = (\mathtt{nx,ny,nz})$, яка на зовнішній межі спрямована назовні тіла. Наслідком цієї домовленості є те, що величини `curv1` і `curv2`, наприклад, для сфери і еліпсоїда будуть від'ємними. В диференціальній геометрії мінімальну і максимальну кривину нормальних перерізів поверхні в точці називають головними кривинами і часто позначають k_1 і k_2. Враховуючи домовленість про знаки, матимемо $k_1 = -\mathrm{curv2}$, $k_2 = -\mathrm{curv1}$, і $k_1 \le k_2$.

Напрямки нормальних перерізів на поверхні, які відповідають кривинам `curv1` і `curv2`, визначаються одиничними векторами з компонентами (`tcurv1x,tcurv1y,tcurv1z`) і (`tcurv2x,tcurv2y,tcurv2z`). Вони розташовані в дотичній площині і завжди ортогональні.

Добуток головних кривин в точці поверхні називається повною або гаусовою кривиною $K = k_1 \cdot k_2 = \mathrm{curv1} \cdot \mathrm{curv2}$. Напівсума головних кривин в даній точці поверхні називається середньою кривиною $H = (k_1 + k_2)/2 = -(\mathrm{curv1} + \mathrm{curv2})/2$.

Аналогічно до двовимірних моделей в 3D застосовуються службові змінні `dom` і `h`.

Приклад 4. Геометричні змінні на еліпсоїді.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Додайте в модель тривимірний компонент.

Untitled.mph (root) : RMB > Add Component ->3D

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Задайте параметри: півосі еліпсоїда a,b,c і координати x0,y0,z0 точки на його поверхні.

Global Definitions : RMB > Parameters

Parameters – **Name, Expression** = a, 3
– **Name, Expression** = b, 2
– **Name, Expression** = c, 1
– **Name, Expression** = x0,1
– **Name, Expression** = y0,1
– **Name, Expression** = z0,c*sqrt(1-x0^2/a^2-y0^2/b^2)

Побудуйте геометрію.

Geometry->**Primitives – More Primitives ->** Ellipsoid

Size – **a-semiaxis, b-semiaxis, c-semiaxis** = a,b,c

Для подальшої демонстрації нам знадобиться крива на поверхні. Побудуйте її. Параметричне рівняння створеного еліпсоїда має вигляд $x = a\sin\theta\cos\tau$, $y = b\sin\theta\sin\tau$, $z = c\cos\theta$. Підставивши сюди, наприклад, вираз $\tau = 2\theta$, ми отримаємо параметричне рівняння кривої на його поверхні.

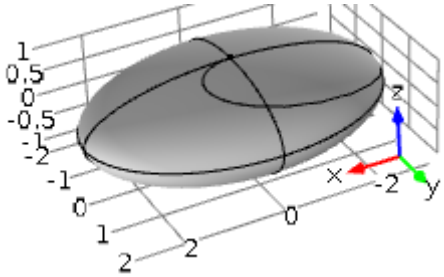
Geometry->**Primitives – More Primitives ->** Parametric Curve

Geometry 1 - Parametric Curve 1 (pc1) :

Parameter – **Name, Minimum, Maximum** = th,0, pi/2

Expressions – **x** = a*sin(th)*cos(2*th)
– **y** = b*sin(th)*sin(2*th)
– **z** = c*cos(th)

LMB > Build All Objects



Щоб в подальшому графікам було на що спиратися (джерело даних), створіть і обчисліть вузол досліджень.

Untitled.mph (root): RMB > Add Study = Add Study ->
Preset Studies – Stationary -> X

Study 1: LMB > Compute

Побудуйте графіки змінних s1 і s2 на поверхні еліпсоїда, причому графік змінної s2 зобразіть на зміщеному вбік еліпсоїді (щоб графіки не перекривалися).

Results **-> Plot Group ->** 3D Plot Group

3D Plot Group 1-> Add Plot – Surface

3D Plot Group 1 - Surface 1 :

Expression – **Expression** = s1

3D Plot Group 1-> Add Plot – Surface

3D Plot Group 1 - Surface 2 :

Expression – **Expression** = s2

Coloring and Style – **Color legend**= False

RMB> Deformation

3D Plot Group 3 – Surface 2 – Deformation 1:

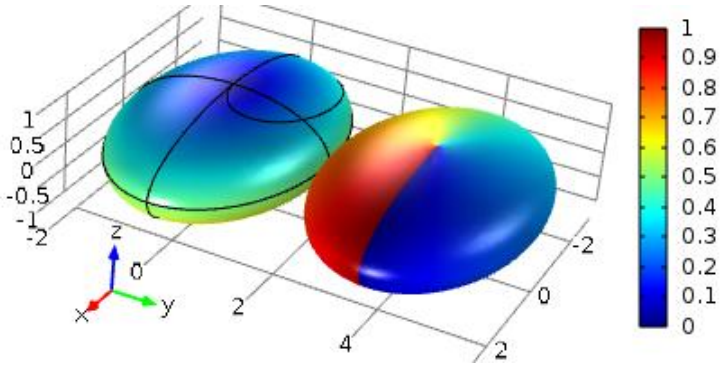
Expression – **x component** = 0

– **y component** = 4

– **y component** = 0

Scale– **Scale factor** = True,1

LMB > Plot



Тут на поверхні правого еліпсоїда зображено величину s2, а на лівому – s1.

Зауваження. Змінні s1 і s2 можна застосовувати будь-де на поверхнях, наприклад, в граничних умовах. Але в більшості випадків використовувати їх важко, бо зв'язки між ними і просторовими координатами складні. Спробуйте самостійно додати до попереднього тіла циліндричний отвір або циліндричну надбудову, і зрозуміти розподіл координат s1 і s2 буде вже важко.

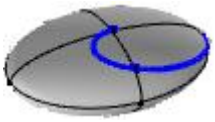
Для кривої (наприклад, ребра на поверхні) змінна її параметризації теж називається s1. У тривимірному режимі, параметр кривої s1 приблизно еквівалентний довжині дуги. Порівняємо графіки значень параметра s1 в точках кривої і поверхні, на якій крива розташована. Для цього спочатку створіть ДД, що співпадає з параметричною кривою, побудованою нами раніше на поверхні.

Results **->** More Data Sets – Edge 3D

Data Sets - Edge 3D 1 :

Data – **Data set** = Study 1/Solution 1

Selection – **Selection** = Manual,4,12



Тепер побудуйте графіки.

Results **-> Plot Group ->** 3D Plot Group

3D Plot Group 2-> Add Plot – Surface

3D Plot Group 2 - Surface 1 :

Expression – **Expression** = s1

3D Plot Group 2-> Add Plot – Line

3D Plot Group 2 - Line 1 :

Data – **Data set** = Edge 2D 1

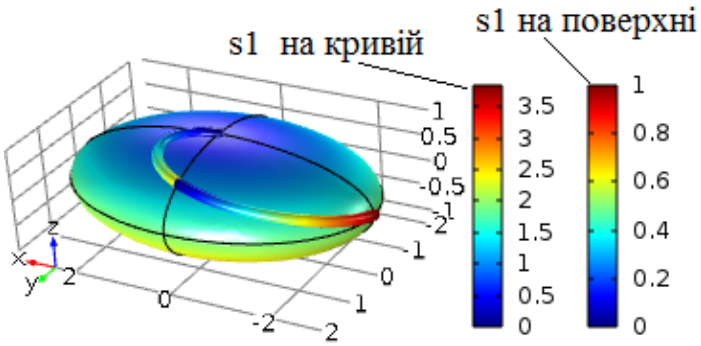
Expression – **Expression** =s1

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.1

– **Radius scale factor** = True,1

LMB > Plot



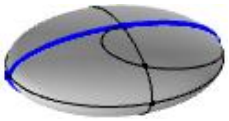
З кольорових палітр видно, що змінна s1, яка віднесена до поверхні, і змінна s1 вздовж кривої, ніяк не пов'язані одне з одним.

Для додаткової перевірки порівняємо значення параметрів s1 на поверхні і вздовж кривих, отриманих в перерізі еліпсоїда площинами x=0 і y=0. Спочатку для цих кривих створіть «власні» джерела даних.

Results **->** More Data Sets – Edge 2D

Data Sets - Edge 2D 2:

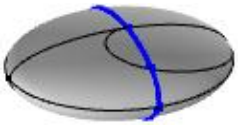
Selection – **Selection** = Manual,2,3,10,13



Results **->** More Data Sets – Edge 2D

Data Sets - Edge 2D 3:

Selection – **Selection** = Manual,6,7,9,11,14



Тепер побудуйте графіки.

3D Plot Group 2: RMB > Duplicate

3D Plot Group 3 - Line 1 :

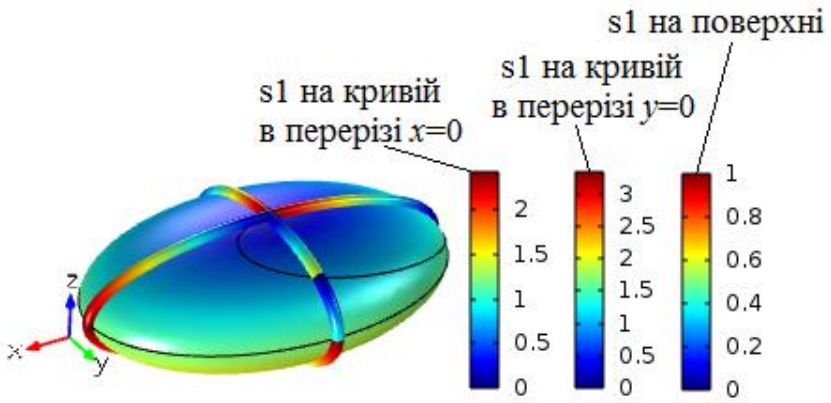
Data – **Data set** = Edge 3D 2

3D Plot Group 3 - Line 1 : RMB > Duplicate

3D Plot Group 3 - Line 2 :

Data – **Data set** = Edge 3D 3

LMB > Plot



Бачимо, що на кожній ділянці кривих параметри `s1` змінюються у власному діапазоні і ніяк не пов'язані з параметром `s1` поверхні.

Перевіримо, як співвідносяться змінні `curv1` на поверхні і кривій.

3D Plot Group 2: RMB > Duplicate

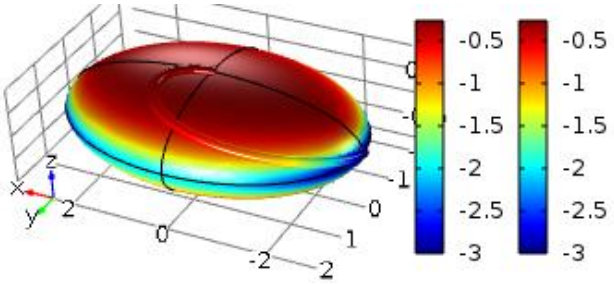
3D Plot Group 4 - Surface 1 :

`Expression` – **Expression** = `curv1`

3D Plot Group 4 - Line 1 :

`Expression` – **Expression** = `curv1`

LMB > Plot



З графіка видно, що кольорові палітри співпадають. Це означає, що на кривій параметр `curv1` «береться з поверхні». Отже в точках кривої на поверхні параметр `s1` відрізняється від параметра `s1` поверхні в тих же точках, а параметр `curv1` однаковий. З останнього графіка також видно, що величина `curv1` від'ємна на всій поверхні еліпсоїда (бо центри кривини протилежні нормалі ***n*** до поверхні).

Побудуйте графік гаусової кривини $K = \text{curv1} \cdot \text{curv2}$ еліпсоїда у вигляді деформації вздовж нормалі початкового еліпсоїда.

3D Plot Group 2: RMB > Duplicate

3D Plot Group 5 :

`Color Legend` – **Show maximum and minimum values** = `True`

3D Plot Group 5 - Surface 1 :

`Expression` – **Expression** = `0`

`Coloring and Style` – **Coloring, `Color`** = `Uniform, Gray`

3D Plot Group 5 - Line 1 : RMB > Delete

`3D Plot Group 5`-> Add Plot – Surface

3D Plot Group 5 - Surface 2 :

`Expression` – **Expression** = `curv1*curv2`

RMB> Deformation

2D Plot Group 5 – Surface 2 – Deformation 1:

Expression – **x component** = `curv1*curv2*nx`

– **y component** = `curv1*curv2*ny`

– **z component** = `curv1*curv2*nz`

Scale– **Scale factor** = `True,1`

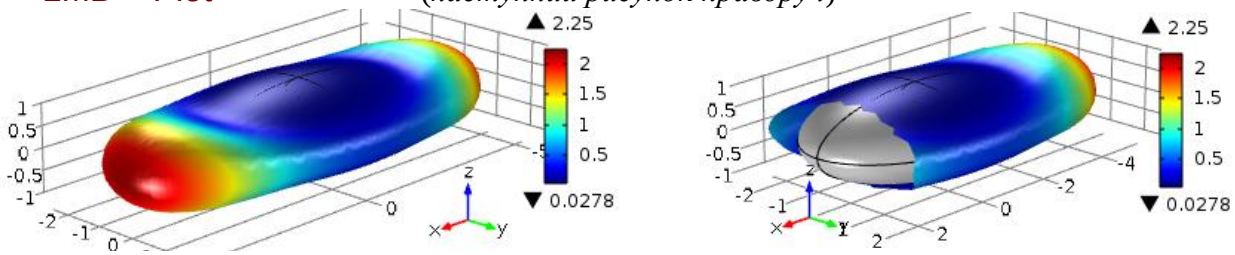
LMB > Plot (*наступний рисунок ліворуч*)

Для кращого розглядання поверхні додайте фільтр, зобразивши її частину з x<=2.

2D Plot Group 5 – Surface 2: RMB> Filter

Element Selection – **Logical expression for inclusion** = `x<=2`

LMB > Plot (*наступний рисунок праворуч*)



Тепер в точці (x_0, y_0, z_0) побудуйте дотичну площину, дотичні прямі вздовж векторів $\boldsymbol{t}_1, \boldsymbol{t}_2$, вектор нормалі $\boldsymbol{n}$, та дотичні прямі вздовж головних напрямків поверхні в цій точці. Спочатку для цього потрібно визначити напрямні вектора дотичних і нормалі в обраній точці. Обчислимо їх як змінні моделі. Створіть вузол `Variables` і заповніть його наступним чином.

`Definitions` -> **`Variables`** – Local Variables

Component 1 - Definitions – Variables 1:

Variables – **Name, Expression** = `Nx, at2(x0,y0,z0,nx)`

– **Name, Expression** = `Ny, at2(x0,y0,z0,ny)`

– **Name, Expression** = `Nz, at2(x0,y0,z0,nz)`

– **Name, Expression** = `st1x, at2(x0,y0,z0,t1x)`

– **Name, Expression** = `st1y, at2(x0,y0,z0,t1y)`

– **Name, Expression** = `st1z, at2(x0,y0,z0,t1z)`

– **Name, Expression** = `st2x, at2(x0,y0,z0,t2x)`

– **Name, Expression** = `st2y, at2(x0,y0,z0,t2y)`

– **Name, Expression** = `st2z, at2(x0,y0,z0,t2z)`

– **Name, Expression** = `ct1x, at2(x0,y0,z0,tcurv1x)`

– **Name, Expression** = `ct1y, at2(x0,y0,z0,tcurv1y)`

– **Name, Expression** = `ct1z, at2(x0,y0,z0,tcurv1z)`

– **Name, Expression** = `ct2x, at2(x0,y0,z0,tcurv2x)`

– **Name, Expression** = `ct2y, at2(x0,y0,z0,tcurv2y)`

– **Name, Expression** = `ct2z, at2(x0,y0,z0,tcurv2z)`

Щоб змінні моделі стали доступними, потрібно оновити обчислення.

Study 1: RMB > Update Solution

Для побудови дотичних знадобляться допоміжні джерела даних типу `Parameterized Curve 3D`.

Results -> Data Set – More Data Sets - Parameterized Curve 3D

Data Sets – Parameterized Curve 3D 1:

Parameter – **Name, Minimum, Maximum** = s,-1,1

Expressions – **x:, y: , z:** = x0+st1x*s, y0+st1y*s, z0+st1z*s

Expressions – **Only evaluate globally defined expressions** = True

Остання опція потрібна, щоб вирази в рівняннях параметричної прямої можна було обчислювати глобально.

Data Sets – Parameterized Curve 3D 1: RMB > Duplicate

Parameterized Curve 3D 2:

Expressions – **x:, y: , z:** = x0+st2x*s, y0+st2y*s, z0+st2z*s

Parameterized Curve 3D 1: RMB > Duplicate

Parameterized Curve 3D 3:

Expressions – **x:, y: , z:** = x0+ct1x*s, y0+ct1y*s, z0+ct1z*s

Parameterized Curve 3D 2: RMB > Duplicate

Parameterized Curve 3D 4:

Expressions – **x:, y: , z:** = x0+ct2x*s, y0+ct2y*s, z0+ct2z*s

Для побудови дотичної площини знадобиться джерело даних типу Parameterized Surface.

Results -> Data Set – More Data Sets - Parameterized Surface

Data Sets – Parameterized Surface 1:

Parameters – **First parameter** – **Name:** = s1

– **Minimum, Maximum** = -1 ,1

Parameters – **Second parameter** – **Name:** = s2

– **Minimum, Maximum** = -1 ,1

Expressions – **x:** = x0+st1x*s1+st2x*s2

– **y:** = y0+st1y*s1+st2y*s2

– **z:** = z0+st1z*s1+st2z*s2

– **Only evaluate globally defined expressions** = True

LMB > Plot

Тепер можна починати будувати зображення дотичних відрізків, дотичної площини і нормалі до поверхні в обраній точці.

Results -> **Plot Group** -> 3D Plot Group

3D Plot Group 6 -> Add Plot – Surface

3D Plot Group 1 - Surface 1 :

Expression – **Expression** = 0

Coloring and Style – **Coloring, Color** = Uniform, Gray

Для зображення точки дотику використаємо вузол Point Trajectories.

3D Plot Group 6 -> Add Plot - More Plots - Point Trajectories

2D Plot Group 6 - Point Trajectories 1 :

Trajectory Data – **x-expression, y-expression, z-expression** = x0,y0,z0

Coloring and Style – **Line style** –**Type** = None

Останнім відключається відображення кривої, а наступні інструкції налаштовують зображення точки.

Coloring and Style – **Point style** –**Type** = Point

– **Point style** –**Point radius expression** = 0.15

– **Point style** – **Radius scale factor** = True,1

– **Point style** – **Color** = Yellow

Намалюйте дотичну площину.

3D Plot Group 6-> Add Plot – Surface

3D Plot Group 6 - Surface 2 :

Data – **Data set** = Parameterized Surface 1

Expression – **Expression** = 0

Coloring and Style – **Coloring, Color** = Uniform, Cyan

LMB > Plot (*наступний рисунок ліворуч*)

Намалюйте дотичні відрізки вздовж векторів $\boldsymbol{t}_1, \boldsymbol{t}_2$.

3D Plot Group 6-> **Add Plot ->** Line

3D Plot Group 6 – Line 1:

Data – **Data set** = Parameterized Curve 3D 1

Expression – **Expression** = 0

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.04

– **Radius scale factor** = True,1

– **Coloring, Color** = Uniform, Red

3D Plot Group 6 – Line 1: RMB > Duplicate

3D Plot Group 6 – Line 2:

Data – **Data set** = Parameterized Curve 3D 2

Зобразіть вектор нормалі.

3D Plot Group 6-> Add Plot – Arrow Line

3D Plot Group 6 - Arrow Line 1 :

Data – **Data set** = Parameterized Curve 3D 1

Expression – **x component, y component, z component** = Nx,Ny,Nz

Coloring and Style – **Scale factor** = True,1.5

– **Placement** =Uniform

– **Number of arrows** =1

– **Color** =Blue

Накресліть в обраній точці дотичні відрізки вздовж головних напрямків.

3D Plot Group 6 – Line 1: RMB > Duplicate

3D Plot Group 6 – Line 3:

Data – **Data set** = Parameterized Curve 3D 3

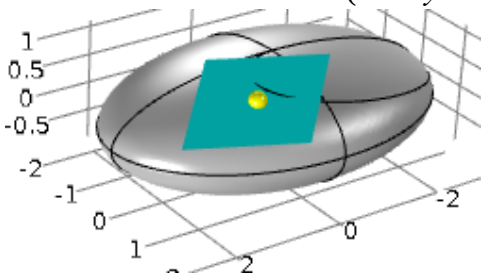
Coloring and Style – **Coloring, Color = Uniform, Green**

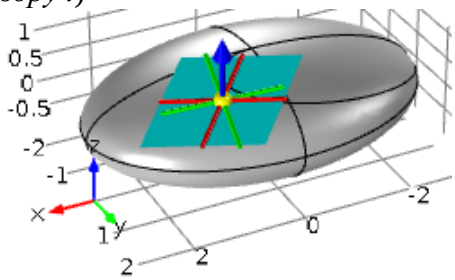
3D Plot Group 6 – Line 3: RMB > Duplicate

3D Plot Group 6 – Line 4:

Data – **Data set** = Parameterized Curve 3D 4

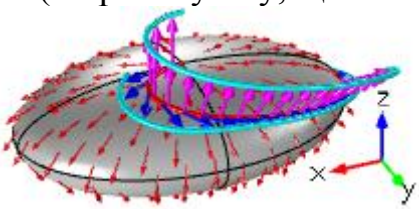
LMB > Plot (*наступний рисунок праворуч*)





З останнього рисунка видно, що напрямки дотичних вздовж векторів $\boldsymbol{t}_1, \boldsymbol{t}_2$ (червоні відрізки), не співпадають з напрямками головних кривин поверхні (зелені відрізки).

Вправа. 1. Побудуйте еквідистантну поверхню до еліпсоїда.
2. Побудуйте графік середньої кривини еліпсоїда.
3. Побудуйте еквідистанту до кривої на поверхні еліпсоїда (вздовж нормалі до поверхні і вздовж дотичного вектора кривої).
4. Побудуйте поле дотичного вектора $\boldsymbol{t}_1$ до поверхні еліпсоїда і до кривої на його поверхні (зверніть увагу, що вони відрізняються).



5. Перевірте, що в поточному прикладі $\boldsymbol{dn} = \boldsymbol{n}$

Підведемо невеликий підсумок параграфа у вигляді таблиці імен розглянутих геометричних і службових змінних із зонами їх застосування.

Рівень геометрії	Розмірність розрахункової області		
	1D	2D	3D
Ребро (Edge)			`s1,t1x,t1y,t1z`
Межа (Boundary)	`nx,` `dnx,` `unx`	`s(s1),nx,ny,` `tx,ty,dnx,dny,` `unx,uny,curv`	`s1,s2,nx,ny,nz,t1x,t1y,t1z,` `t2x,t2y,t2z,dnx,dny,dnz,` `unx,uny,unz,curv1,curv2,` `tcurv1x,tcurv1y,tcurv1z,` `tcurv2x,tcurv2y,tcurv2z`
Область (Domain)	`h`	`h`	`h`
Будь який	`x,dom`	`x,y,dom`	`x,y,z,dom`

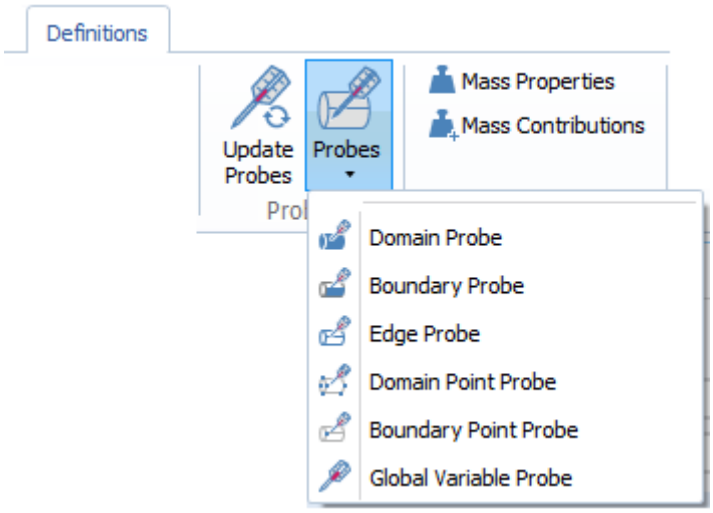
7.3. Зонди, фізичні властивості матеріалів і змінні маси.

Сутності, перелічені в заголовку параграфа, представляють специфічні змінні системи COMSOL Multiphysics. Вони створюються іншим способом, ніж величини вузлів `Variables`, але можуть застосовуватися однаково зі звичайними змінними. Ми вже використовували їх в різних прикладах і тут наведемо короткий огляд.

Дослідники природничих процесів часто розміщують датчики в визначених місцях експериментальної установки, і за їх допомогою стежать за

еволюцією фізичного явища у часі. В COMSOL Multiphysics при дослідженні моделей ви можете робити те саме за допомогою зондів. Зонд – це віртуальний датчик, реалізований в COMSOL Multiphysics у вигляді спеціальної скалярної змінної. Його первісне призначення – контроль значень полевих змінних в точках під час обчислення. Але зонди COMSOL мають більш гнучкі можливості порівняно з давачами, наприклад, можуть обчислювати середнє значення полевої змінної по всьому тілу. Як і фізичний давач, що підключений до зовнішніх інтерфейсів виведення, зонд «друкує» дані в таблицю і по ній будує графік відповідної величини (в динамічних задачах – в залежності від часу, а в статичних - в залежності від параметра). Для свого функціювання він автоматично будує в ДМ власні джерела даних.

Зонди створюються у вузлах **Definitions** компонентів. Список доступних типів зондів можна побачити, якщо розкрити випадне меню Probes (зонди) на стрічці `Definitions`.



Для контролю значень полевої змінної в точках призначені `Domain Point` і `Boundary Point` зонди. Ці «точкові» зонди імітують роботу давачів в фізичному експерименті. Вони являють змінні, які містять значення «полевих» величин в точках, розташованих всередині об'єму (`Domain point`) або на його межах (`Boundary point`).

Просторові зонди (`Domain Probe`), поверхневі зонди (`Boundary Probe`) і реберні зонди (`Edge Probe`) дозволяють відслідковувати середнє, мінімальне, максимальне або інтегральне значення полевої величини по об'єму, поверхні або вздовж ребра. Інакше кажучи, вони відслідковують інтегральні характеристики величин по відповідним елементам геометрії.

Зонд типу `Global Variable Probe` не використовує геометрію і відноситься до всього компоненту. Він обчислює вираз, який не залежить від геометрії або координат точки.

З точки зору користувача, зонди це змінні, які не є функціями просторових координат, але, зазвичай, бувають функціями часу або параметру. Всі зонди мають імена і відрізняються від звичайних змінних лише тим, що під час обчислення моделі вони автоматично «створюють» кілька додаткових вузлів ДМ. Зонди можна використовувати в моделі як будь-які інші змінні,

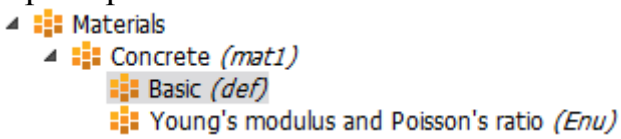
наприклад, в рівняннях, граничних умовах або в умовах зупинки обчислень. Опис зондів було надано в п. 1.5.4.

Інший тип величин – змінні властивостей матеріалів – зберігає числові значення фізичних характеристик речовин (Material Properties). Вони надаються вузлами матеріалів в гілці **Materials**, і їх відмінність від звичайних змінних полягає, здебільшого, в способі іменування.

COMSOL Multiphysics використовує простір імен з ієрархічною структурою, яка реалізується за допомогою точкової нотації. Для звернення до змінних верхнього рівня дерева моделей використовується тег `root`. Для посилання на змінні компонента використовується його ім'я, наприклад, `comp1`. Для посилання на змінні у фізичному інтерфейсі використовується його назва, наприклад, `solid`. Так для доступу до змінної переміщення `disp` в інтерфейсі `Solid Mechanics` з тегом `solid` слід використовувати ім’я `solid.disp`. Посилаючись на змінну, потрібно вказати лише ту частину повного імені, яка робить змінну унікальною. Наприклад, в тому ж інтерфейсі `Solid Mechanics (solid)` з єдиним в моделі компонентом достатньо використати назву `solid.disp`, але `comp1.solid.disp` та повне ім'я `root.comp1.solid.disp` також є правильними. Щоб отримати доступ до цієї змінної з іншого компонента, застосовуйте `comp1.solid.disp` або `root.comp1.solid.disp`. Для доступу до глобального параметра, можна використати його ім’я `param1` безпосередньо, або `root.param1`. Залежні змінні (змінні поля) в межах компонента є унікальними, і вам непотрібно ім’я компонента для доступу до них. Але, наприклад, для доступу з першого компонента моделі до шуканої змінної `U` другого компонента `comp2` потрібно використовувати ім’я `comp2.U`.

При роботі з властивостями матеріалів потрібно враховувати наступні правила іменування:

- При додаванні вузла матеріалу в ДМ йому надається «внутрішнє» ім’я. Як правило, перший матеріал в гілці **Materials** має тег `mat1`. Властивості матеріалу розподіляються по групам. Вузол основної групи властивостей називається `Basic`, і для доступу до змінних цього вузла використовується простір імен з тегом `def`.



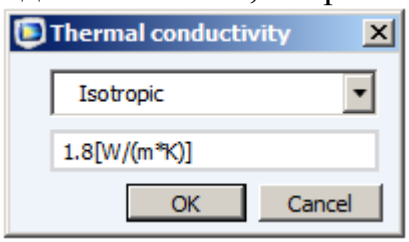
  Самі назви властивостей відображаються в ПВ вузлів груп в таблицях **`Output properties`** в стовпці `Variable`.

Output properties

Property	Variable	Expression	Unit	Size
Coefficient of thermal expansion	alpha ; alphaii = alpha, alphaij = 0	10e-6[1/K]	1/K	3x3
Density	rho	2300[kg/m^3]	kg/m³	1x1
Thermal conductivity	k ; kii = k, kij = 0	1.8[W/(m*K)]	W/(m·K)	3x3
Heat capacity at constant pressure	Cp	880[J/(kg*K)]	J/(kg·K)	1x1
Electrical conductivity	sigma ; sigmaii = sigma, sigmaij...	123[S/m]	S/m	3x3

Для доступу до імен властивостей використовується точкова нотація. Наприклад, синтаксис посилання на густину `rho` матеріалу `mat1` матиме вигляд `mat1.def.rho` (або повне ім'я `root.comp1.mat1.def.rho`). Для властивостей у групах, відмінних від групи `Basic(def)`, слід використовувати повний простір імен, включаючи ім'я групи властивостей.

- Для анізотропних властивостей, таких як теплопровідність, значення доступні як `mat1.def.k11`, `mat1.def.k12`, тощо. Якщо така властивість встановлена як ізотропна, то ці значення будуть однаковими і можна використовувати одне означення, наприклад, `mat1.def.k11`



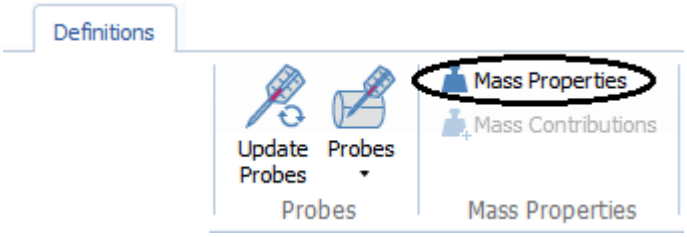
Для певного матеріалу ім'я `mat1.def.k_iso` представляє це ізотропне значення. Але майте на увазі, що в COMSOL 5.1 використання змінних типу `k_iso` декларується, але не працює. Це виправлено в наступних версіях програми.

- Якщо в моделі використовується кілька матеріалів то, щоб отримати доступ до якоїсь властивості у всій моделі використовується спеціальний контейнер `root.material`. Наприклад, `root.material.rho` - це густина ρ всіх матеріалів, яка складається з щільності речовин всіх частин геометрії.
- Багато фізичних інтерфейсів визначають власні змінні властивостей матеріалів. Наприклад, `solid.rho` - це густина в `Solid Mechanics` інтерфейсі. Якщо ви використовуєте густину по всій в моделі, то `solid.rho` і `material.rho` ідентичні.

Зауваження. У вузлах матеріалів існує можливість визначати властивості, як функції деякої величини, наприклад, як функції температури `T`. Тоді всі матеріальні характеристики будуть не змінними, а функціями, і до них потрібно буде звертатися, як до функцій, наприклад, `mat1.def.rho(T)`.

Існує група величин, які називаються змінними маси (`Mass Properties`). Вони обчислюють і зберігають такі розповсюджені властивості тіл, як об'єм, площа, маса, тощо. Для цього в гілці **Component – Definitions** створюється вузол **Mass Properties 1 (mass1)**, який представляє властивості компонента,

пов'язані з його геометрією. Він додається в ДМ кнопкою Mass Properties інструментальної стрічки `Definitions`.



Для обчислення маси в ПВ цього вузла є розділ, в якому задається густина речовини. Можна обчислювати наступні характеристики: об'єм та масу тіла; центр мас; тензор інерції та його головні осі. Які характеристики обчислювати і зберігати в змінних моделі вказується відповідними прапорцями, що розташовані внизу ПВ вузла **Mass Properties 1(mass1)**. Назви змінних за замовчуванням наведено в наступній таблиці

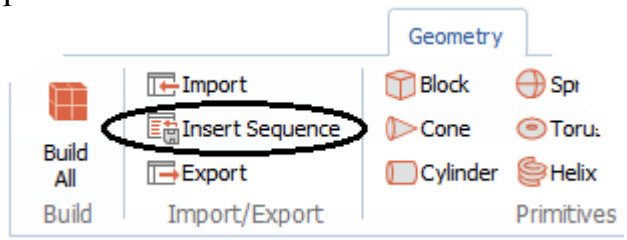
Величина	**Ім'я змінної за замовчуванням**
Об'єм	`mass1.volume` (`mass1.area` в 2D)
Маса	`mass1.mass`
Центр мас	`mass1.CMX, mass1.CMY, mass1.CMZ`
Момент інерції	`mass1.IXX, mass1.IXY, mass1.IXZ ...`
Головні осі інерції	`mass1.Ip1X, mass1.Ip2Y ...`
Головні моменти інерції	`mass1.Ip1, mass1.Ip2, mass1.Ip3`

Вузлів **Mass Properties** в моделі може бути багато, і в кожному можна обирати різні рівні геометрії і різні значення густини речовини. Тоді змінні маси таких вузлів будуть представляти інтеграли від густини по відповідним сутностям. Отже ці змінні надають ще одну можливість обчислювати інтеграли.

Для демонстрування можливостей зондів, змінних фізичних характеристик матеріалів і змінних маси ми розглянемо задачу розв'язання рівняння теплопровідності в тілі у формі куба з трьома ортогональними циліндричними отворами. Форма геометрії досить складна, і знаходити об'єм, масу і площі граней тіла клопітно, якщо не застосовувати інструменти системи, призначені для обчислення цих параметрів, наприклад, змінні маси.

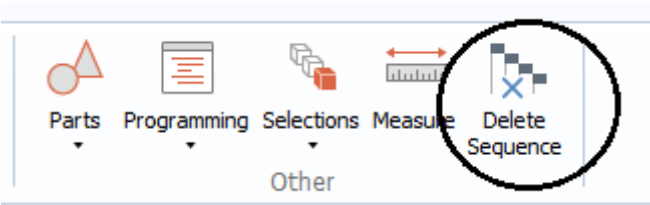
В п. 1.5.4 пояснення сенсу зондів відбувалося на прикладі дослідження впливу на схоже тіло зовнішнього навантаження, і модель була збережена в файлі з ім'ям `cube_with_holes.mph`. Тут ми використаємо ту саму геометрію, але досліджуватимемо іншу фізичну задачу.

В COMSOL Multiphysics існує можливість копіювати геометрію з одного `mph` файлу в інший. Для цього на панелі `Geometry` є кнопка Insert Sequance, яка завантажує послідовність інструкцій побудови геометрії з обраного `mph` файлу в гілку геометрії поточного.



Якщо просторові розмірності геометрій співпадають, послідовність «геометричних» інструкцій обраного файлу завантажується після поточної «геометричної» інструкції. Коли в іншому файлі є кілька компонентів, то відкривається діалогове вікно вибору потрібної геометрії. Якщо завантажувана геометрія містить посилання на функції або параметри, то вони також завантажуються в модель поточного файлу. Параметри і глобальні функції завантажуються у відповідні глобальні гілки, а локальні функції – в гілку **Definitions** робочого компонента. Функції і параметри завантажуються навіть тоді, коли в моделі вже є однойменні об'єкти. Тоді, перш ніж будувати геометрію, ви зобов'язані вручну розв'язати конфлікти імен.

Зауваження. Інколи потрібно видалити всю послідовність інструкцій побудови розрахункової геометрії. Для цього призначена кнопку Delete Sequance панелі `Geometry`.



Наступний навчальний приклад призначений для того, щоб продемонструвати використання згаданих вище інструментів.

Приклад. Дослідити процес розповсюдження тепла в кубічному тілі з циліндричними отворами. Потім спростити задачу, розглянувши тіло, як суцільне із заданими масою m, теплоємністю c і зовнішнім тепловим потоком q (обчисленими в загальній моделі). Порівняти середню температуру кубічного тіла з розв'язком ЗДР $mc\dfrac{dT}{dt}=q$, яке описує процес нагрівання/остигання тіла, розглядуваного як матеріальна точка.

Розв'язання. Створіть тривимірну `Heat Transfer in Solids` модель.

File –> New –> Model Wizard –> 3D –>

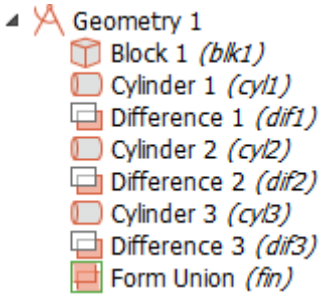
Heat Transfer - Heat Transfer in Solids (ht) –>Time Dependent

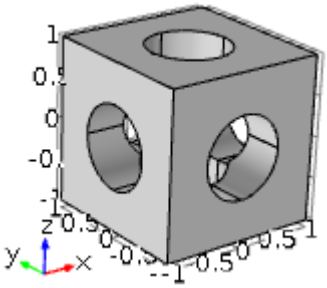
Щоб вдруге не створювати ту ж саму геометрію, використайте інструкції з файлу `cube_with_holes.mph`.

`Geometry` **–> Import/Export** – Insert Sequance

У відкрившемуся вікні вибору знайдіть і оберіть файл `cube_with_holes.mph`.

Geometry 1: LMB > Build All Objects



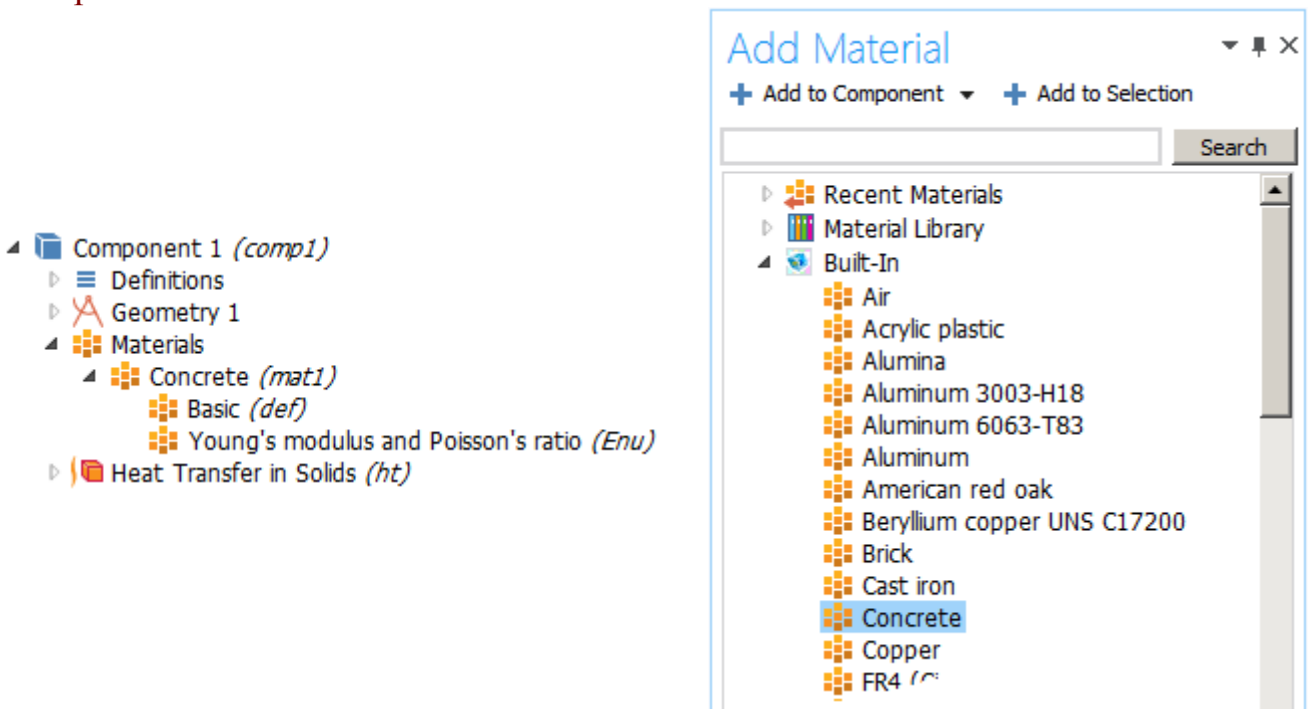


Використана тут кнопка Insert Sequance завантажує послідовність інструкцій створення геометрії з обраного/іншого COMSOL Multiphysics файлу в гілку геометрії поточного файлу. Якщо потрібного вам файлу немає, то поверніться до параграфу 1.5.4, і з його прикладу повторіть послідовність команд створення геометрії.

Додайте в модель вузол матеріалу.

`Materials` **-> Materials** – Add Material

Праворуч від графічного вікна відкриється вікно `Add Material` (наступний рисунок праворуч). У вікні `Add Material` в дереві доступних матеріалів виберіть `Built-In – Concrete`.



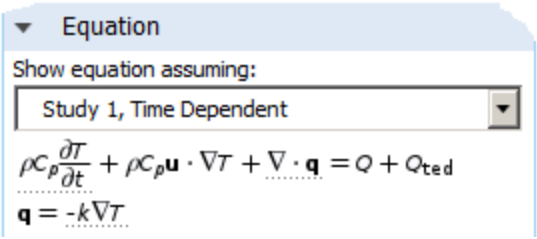
Клацніть по кнопці +Add to Component, яка розташована вгорі цього вікна, потім закрийте вікно «хрестиком» в правому верхньому куті. В ДМ з'явиться вузол **Materials - Concrete (mat1)** з двома підвузлами. В таблицях ПВ цих підвузлів в стовпці **`Variables`** можна побачити позначки для фізичних характеристик матеріалу (бетону): `rho` – густина, `k11,k22,...` – компоненти тензора теплопровідності, `Cp` – теплоємність, тощо.

Output properties

Property	Variable	Expression	Unit	Size
Coefficient of thermal expansion	alpha ; alphaii = alpha, alphaij = 0	10e-6[1/K]	1/K	3x3
Density	rho	2300[kg/m^3]	kg/m³	1x1
Thermal conductivity	k ; kii = k, kij = 0	1.8[W/(m*K)]	W/(m·K)	3x3
Heat capacity at constant pressure	Cp	880[J/(kg*K)]	J/(kg·K)	1x1

Головні властивості розташовані в ПВ вузла **Concrete (mat1) – Basic(def)**. Тому звертатися до них потрібно по повному імені, наприклад, `mat1.def.Cp` або `mat1.def.k11`.

Тепер налаштуйте «фізику». Клацніть по вузлу **Heat Transfer in Solids (ht)**. В його ПВ ви побачите рівняння, які описують обране фізичне явище.

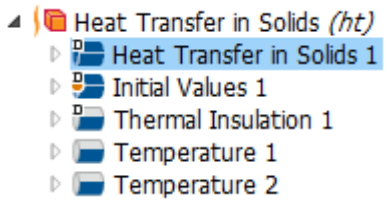


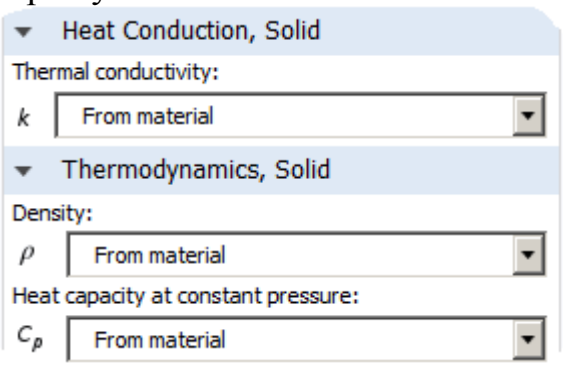
Інтерфейс **Heat Transfer in Solids (ht)** використовується для дослідження розповсюдження тепла в твердих тілах. Цей процес описується рівнянням

$$\rho C_{\text{P}} \frac{\partial T}{\partial t} + \rho C_{\text{P}} \boldsymbol{u} \cdot \nabla T + \nabla \cdot \boldsymbol{q} = Q + Q_{ted},$$

де T – шукане поле температури, $\boldsymbol{q} = -k\nabla T$ – тепловий потік, ∇ – оператор градієнта. В рівняння входять матеріальні коефіцієнти: густина ρ, теплоємність C_{P}, і коефіцієнт теплопровідності k (скалярний або тензорний). В рівняння входять також вектор швидкості руху $\boldsymbol{u}$ речовини і джерела тепла Q різної природи. Поле швидкостей використовується тоді, коли частини моделі рухаються одна відносно іншої. В нашому випадку цього немає. Відсутні також джерела тепла. Отже відповідні члени рівняння вважаються нульовими.

Коефіцієнти рівняння задаються в ПВ вузла **Heat Transfer in Solids (ht) - Heat Transfer in Solids 1**. Якщо в цій панелі нічого не змінювати, то вони автоматично будуть взяті з параметрів матеріалу.





Задайте на верхній грані температуру `1000[K]`, на нижній - `293.15[K]`, а початкову температуру вважатимемо в тілі повсюди однаковою `400[K]`.

`Physics` **-> Boundariy -** Boundaries ->Temperature

Heat Transfer in Solids(ht) - Temperature 1:

Boundary Selection – **Selection** = `Manual,4` (*верхня грань*)

Temperature – **Temperature:** $\mathbf{T_o}$ = `1000[K]`

`Physics` **-> Boundariy -** Boundaries ->Temperature

Heat Transfer in Solids(ht) - Temperature 2:

Boundary Selection – **Selection** = `Manual,3` (*нижня грань*)

Temperature – **Temperature:** $\mathbf{T_o}$ = `293.15[K]`

На інших боках тіла ми задавати нічого не будемо. Це означатиме, що залишаться налаштування вузла **Thermal Insulation 1** (теплоізоляція).

Задайте початкову температуру.

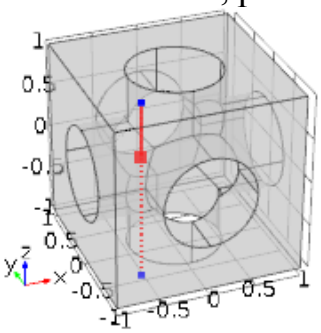
Heat Transfer in Solids(ht) - Initial Values 1:

Initial Values – **Temperature:** **T** = `400[K]`

Створіть зонди. Спочатку – точковий зонд. Він імітуватиме давач, який вимірює температуру у внутрішній точці тіла.

Definitions **-> Probes** - Probes -> Domain Point Probe

Існує кілька способів вказівки точок в зонді Domain Point Probe. Спосіб вибирання задається в списку **Line entry method**. Нехай це буде Point and surface normal. Тоді вибір точки складається з наступних дій. Ви повинні клацнути по поверхні тіла. В графічному вікні на зображенні тіла з'явиться зображення нормалі до поверхні, яка проходить через точку клацання. Потім в ПВ за допомогою повзуна потрібно задати глибину розташування давача, положення якого позначається червоною точкою на лінії нормалі. Координати точки автоматично відображаються в полях, розташованих нижче повзуна.



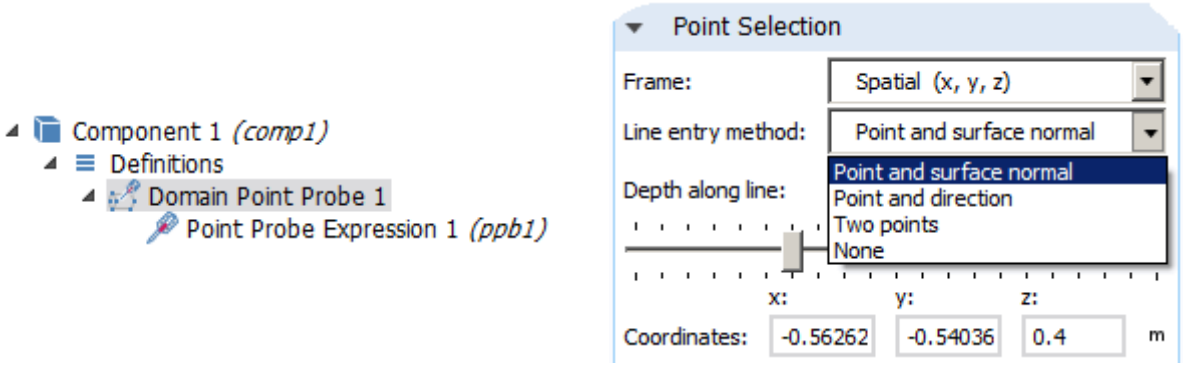
Якщо в списку **Line entry method** обрати None, то в розділі **Point Selection** залишаться лише поля для введення координат точки.

Definitions - Domain Point Probe 1:

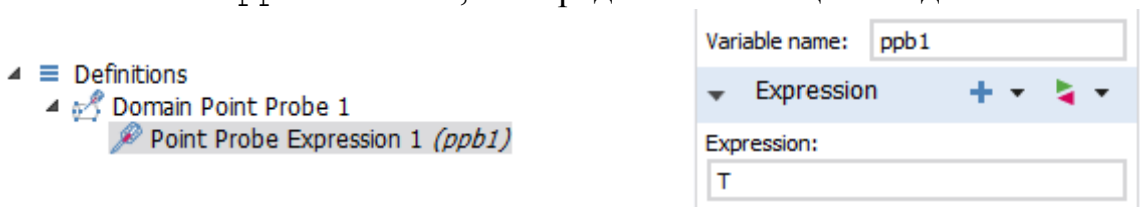
Point Selection – **Line entry method**=Point and surface normal

Клацніть мишею по якій-небудь точці на поверхні і за допомогою повзуна задайте її відстань від поверхні (всередину тіла).

Point Selection – **Depth along line** = 0.3



Координати точки зображено на рисунку (у вас будуть власні числа). Величина, яка обчислюватиметься зондом (у нас температура T), задається в ПВ дочірнього вузла **Point Probe Expression 1**. В ньому ж у полі **Variable name** визначається ім'я ppb1 змінної, яка представлятиме цей зондом.



Тепер створіть «просторовий» `Domain Probe` зонд, який відслідковуватиме середню температуру по всьому тілу.

`Definitions` **-> Probes -** Probes -> Domain Probe

Definitions - Domain Probe 1:

`Probe Type` – **Type** = `Average`

`Expression` – **Expression** = `T`

Зоною його дії залиште всі області (`All domains`), тобто середня температура обчислюватиметься по всьому тілу. Також зверніть увагу на ім'я змінної в полі **`Variable name`**, яка представлятиме цей зонд – `dom1`.

В подальшому в цьому прикладі нам знадобляться значення теплових потоків через верхню і нижню грані. Для цього можна створити «поверхневі» `Boundary Probe` зонди. Для тренування читача один з них ми створимо таким, що обчислює середнє значення густини теплового потоку через верхню грань, а другий відразу обчислюватиме сумарний (інтегральний) тепловий потік через нижню грань.

`Definitions` **-> Probes -** Probes -> Boundary Probe

Component 1 – Definitions - Boundary Probe 1 (bnd1):

`Probe Type` – **Type** = `Average`

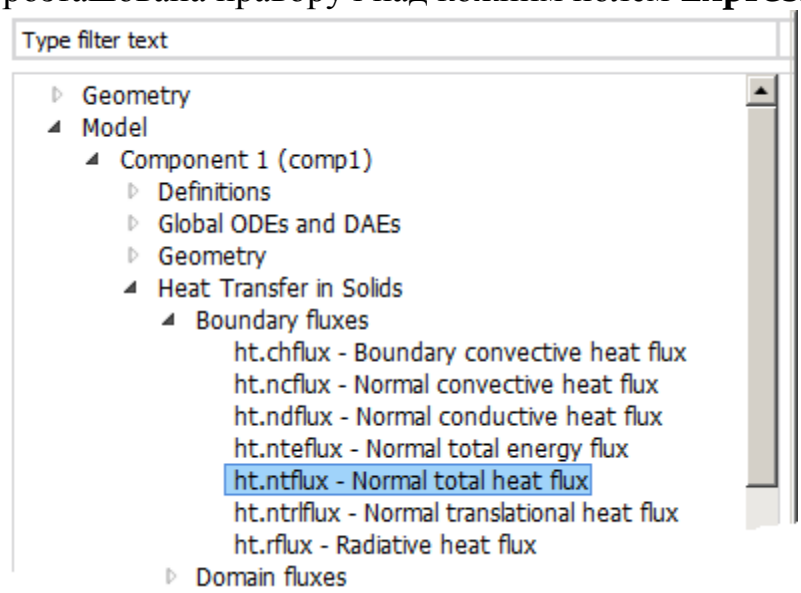
Очистіть поле вибору кнопкою Clear Selection, і клацніть мишею по верхній грані.

`Source Selection` – **Selection** = `Manual,4` (*верхня грань*)

Потім задайте обчислюваний вираз.

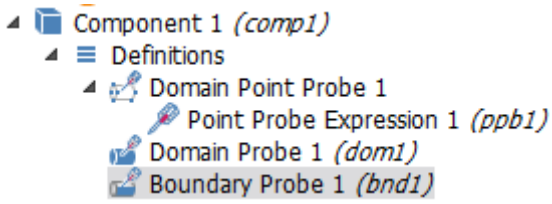
`Expression` – **Expression** = `ht.ntflux`

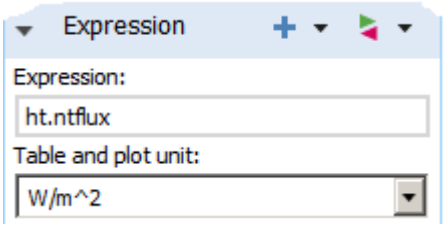
Нагадаємо, що кожний прикладний інтерфейс створює власні змінні. Їх назви можна побачити, якщо розкрити вікно готових величин кнопкою Replace Expression, що розташована праворуч над кожним полем **Expression**.



Ми розкрили це вікно і знайшли змінну `ht.ntflux`, по якій виконали подвійне клацання мишею. Ця змінна представляє густину теплового потоку по нормалі до поверхні, на якій вона обчислюватиметься. Зонд обчислюватиме це значення

в різні моменти часу. Одиниці вимірювання W/m^2 цієї величини вказано в списку **Table and plot unit**.





Другий Boundary Probe зонд обчислюватиме сумарний (інтегральний) тепловий потік через нижню грань.

Definitions **->** **Probes** **-** Probes **->** Boundary Probe

Component 1 – Definitions - Boundary Probe 2 (bnd2):

Probe Type – **Type** = Integral

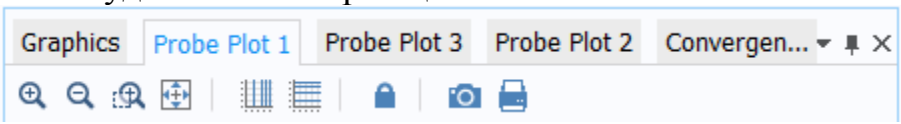
LMB > Clear Selection (*очистити зону вибору*)

Source Selection – **Selection** = Manual,3 (*нижня грань*)

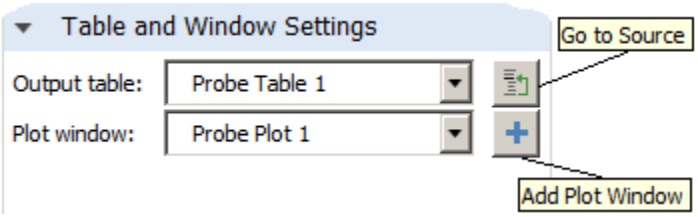
Expression – **Expression** = ht.ntflux

Одиниці вимірювання W змінної bnd2 відрізняються від попереднього зонду тому, що ця величина представляє сумарний тепловий потік через грань.

Налаштуйте таблиці і графічні вікна, які COMSOL використовуватиме для відображення значень зондів і їх графіків. За замовчуванням COMSOL використовує спільну таблицю зондів (як правило з назвою **Probe Table 1**) та будує спільний графік зондових змінних (як правило у вузлі **Probe Plot Group X - Probe Table Graph 1**). «Зондові» графіки з'являються в графічній зоні в окремому вікні (або в кількох вікнах) під час обчислення. І у кожного графічного вікна буде власний корінець.



Щоб згрупувати кілька зондів по різних таблицях і графіках, керуйте назвами таблиць та вікон графіків. Вони визначаються в розділах **Table and Window Settings** ПВ всіх зондів.



Зі списку **Output table** виберіть Default, New table або будь-яку існуючу таблицю зондів. Якщо бажано використати і обрати нову таблицю зондів, натисніть кнопку Go to Source, щоб створити її ім'я. У списку **Plot window** виберіть Default, New window або будь-яке існуюче «зондове» графічне вікно. Можна також натиснути кнопку Add Plot Window, щоб створити ім'я корінця нового «зондового» графічного вікна.

В нашій моделі для всіх зондів в полі **Output table** залиште значення Default. Це означатиме що значення всіх зондів друкуватимуться в спільну таблицю. Але назви зондових графічних вікон змінимо. Для перших двох

зондів, які друкують температуру в точці і середню температуру по об'єму, в полі **`Plot window`** залиште значення `Default`. Для граничних зондів **Boundary Probe 1 (bnd1)** і **Boundary Probe 2 (bnd2)** оберіть інше графічне вікно, бо вони обчислюють не температуру, а теплові потоки. Для цього в гілці **Definitions** в ПВ вузла **Boundary Probe 1 (bnd1)** в розділі **`Table and Window Settings`** клацніть два рази по кнопці Add Plot Window, встановивши у випадному списку назву `Prob Plot 2`. У вузлі **Boundary Probe 2 (bnd2)** з випадного списку **`Plot window`** оберіть те саме ім'я. Так ви встановите назву корінця графічного вікна, призначеного для виведення графіків цих двох зондів (назви вузлів графічних груп будуть іншими).

Задайте діапазон і інтервал зміни часу `t`.

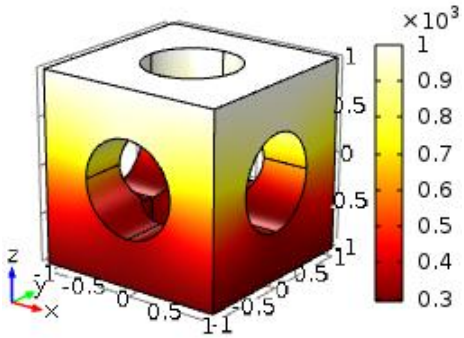
Study 1 - Step 1: Time Dependent:

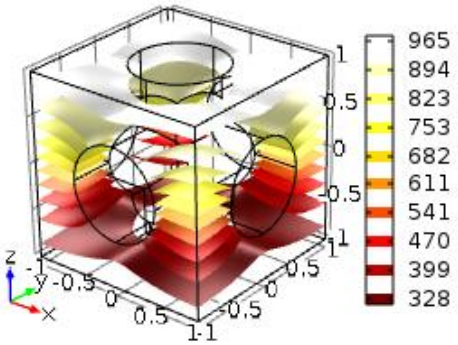
`Study Settings` –**Times** = `range(0,10000,1000000)` (*в секундах*)

Виконайте обчислення.

Study 1: RMB > Compute

По завершенні обчислень в гілці **Results** з'явиться поверхневий графік температури **3D Plot Group 1** (наступний рисунок ліворуч) і графік **Isothermal Contours (ht)** поверхонь рівня температури (наступний рисунок праворуч).





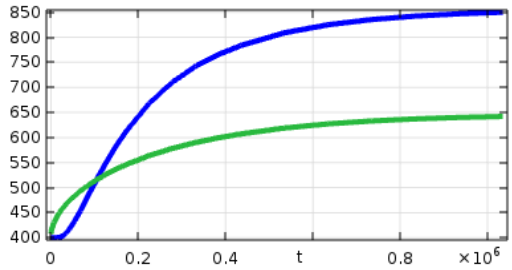
Коли обчислення триватимуть, в інформаційній зоні вікна COMSOL буде поступово друкуватися таблиця **Probe Table 1** зі значеннями створених зондів. Якщо таблиці не видно, активуйте її, клацнувши мишею по відповідному корінцю.

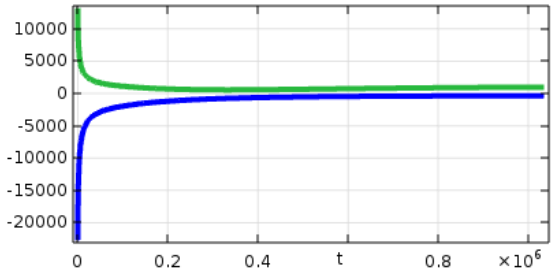
Probe Table 1

t	Temperature (K)	Temperature (K), [	Normal total he	Normal total heat
4096.7	400.01	423.16	-9581.8	5488.0
5735.4	400.00	427.28	-8236.4	4715.0
7374.1	400.00	430.87	-7330.4	4195.5
9012.8	400.00	434.09	-6673.6	3819.5
12290	400.07	439.82	-5714.3	3270.6
15568	400.27	444.82	-5085.2	2910.8
18845	400.71	449.29	-4637.1	2654.4
22122	401.52	453.38	-4294.3	2458.3
28677	404.71	460.81	-3769.3	
35232	410.00			
41787				

По завершенні обчислень по цій таблиці буде створено два графічних вузлі. У вузлі **Probe Plot Group 3** (наступний рисунок ліворуч) будуть намальовані

графіки температури в залежності від часу (в точці і середня по об'єму). А у вузлі **Probe Plot Group 4** будуть зображені графіки середньої густини теплового потоку на верхній грані і сумарного теплового потоку на нижній грані (наступний рисунок праворуч).





Щоб з густини теплового потоку на верхній грані отримати сумарний тепловий потік, її ще потрібно буде помножити на площу грані.

Перш ніж продовжувати роботу далі, проаналізуємо множини вузлів ДМ, які створені зондами.

Для зонду **Domain Point Probe 1** створено джерело даних **Probe Solution 2** (копія ДД **Study 1/Solution 1**, змінено лише фрейм координат) і з нього, як з базового, згенеровано власне джерело даних **Domain Point Probe 1** типу `Cut Point 3D`. Окрім того, в гілці залежних величин **Derived Values** створено вузол **Point Probe Expression 1** (типу `Point Evaluation`), з якого дані друкувалися в таблицю **Probe Table 1**.

Definitions	Data Sets	Derived Values
Definitions Domain Point Probe 1 Point Probe Expression 1 *(ppb1)*	Data Sets Study 1/Solution 1 Probe Solution 2 Domain Point Probe 1	Derived Values Point Probe Expression 1

Це загальна особливість – точкові зонди (`Domain Point Probe` і `Boundary Point Probe`) створюють власне (спільне) джерело даних, з якого генеруються залежні набори даних `Domain Point Probe` і `Boundary Point Probe`.

Зауваження. В одній точці можна створювати кілька зондових змінних. Наприклад, на наступному рисунку показано, що точковий граничний зонд має дві зондові змінні `ppb1` і `ppb2`..

Boundary Point Probe 1
Point Probe Expression 1 *(ppb1)*
Point Probe Expression 2 *(ppb2)*

Тобто точка спільна, але в ній можна вимірювати кілька фізичних величин. При чому в гілці **Derived Values** для кожної змінної створюватиметься власний вузол залежної величини.

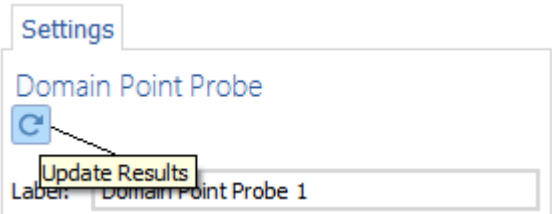
Для зондів `Domain Probe` і `Boundary Probe` (а також `Edge Probe`, якщо він є) створюється спільне джерело даних (в нашому прикладі **Probe Solution 3**). З нього, як з базового, утворюються «звужені» набори даних **Domain Probe 1**, **Boundary Probe 1**, **Boundary Probe 2**. Відповідно до цього в гілці **Derived Values** з'являються вузли залежних величин.

Definitions	Data Sets	Derived Values
Domain Probe 1 *(dom1)* Boundary Probe 1 *(bnd1)* Boundary Probe 2 *(bnd2)*	Probe Solution 3 Domain Probe 1 Selection Boundary Probe 1 Selection Boundary Probe 2 Selection	Derived Values Point Probe Expression 1 Domain Probe 1 Boundary Probe 1 Boundary Probe 2

Якщо було б створено глобальний зонд `Global Variable Probe` , то він записав би свою інформацію в загальне джерело даних, створене «інтегральними» зондами (в нашому прикладі в **Probe Solution 3**) і не генерував би залежний набір даних. Але подібно іншим зондам, в гілці **Derived Values** з'явився би вузол **Global Variable Probe 1**.

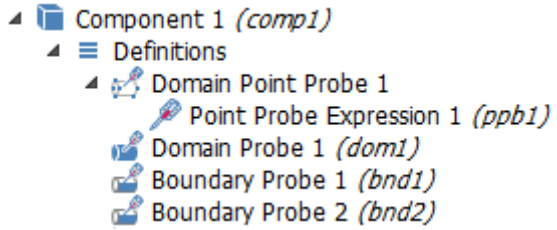
Definitions	Data Sets	Derived Values
Global Variable Probe 1 *(Rho1)*	Probe Solution 3	Global Variable Probe 1

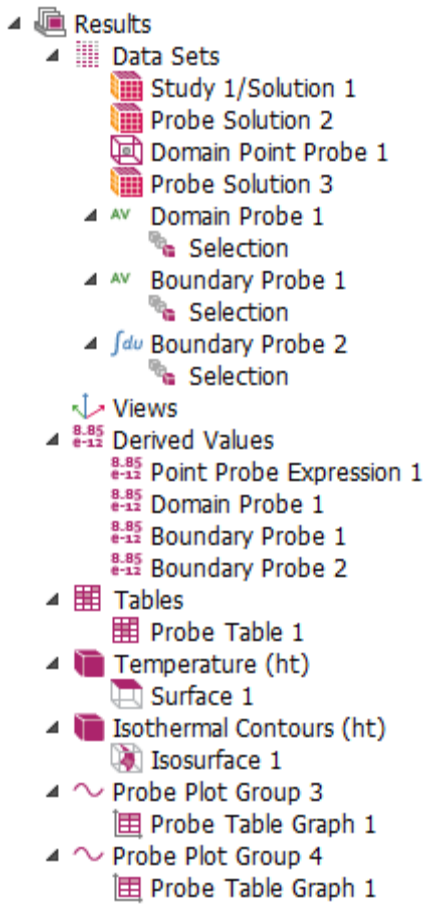
Обговорюючи зонди, треба звернути увагу на різницю дії команд Update Solution і Update Results. Кнопка останньої розташована вгорі ПВ всіх зондів.



Команда Update Solution лише дає можливість використовувати зондові змінні. А команда Update Results призводить до оновлення (або створення) вузлів джерел даних, вузлів залежних змінних, «зондової» таблиці (або до додавання стовпців в таблицю) і до побудови «зондових» кривих.

На наступному рисунку праворуч зображено гілку **Results** в поточний момент моделювання задачі.





Окремо побудуйте графіки теплових потоків через верхню і нижню грані, а також їх суму. Оскільки всі боки тіла теплоізольовані, то останнє буде представляти сумарний тепловий потік, який надходить в тіло. При чому тепловий потік через нижню грань представляється зондовою змінною `bnd2`, а для обчислення потоку зверху нам знадобиться добуток змінної `bnd1` на площу грані. Для її обчислення створіть, наприклад, змінну маси, яка чисельно буде дорівнювати цій площі. Звичайно, що площу і масу в COMSOL можна обчислити різними способами, та ми обрали інструмент Mass properties.

`Definitions` **`->Mass properties`** – Mass properties

Definitions - Mass Properties 1 (mass1):

`Source Selection` – **Geometric entity level** = `Boundary`

– **Selection** = `Manual,4` (*верхня грань*)

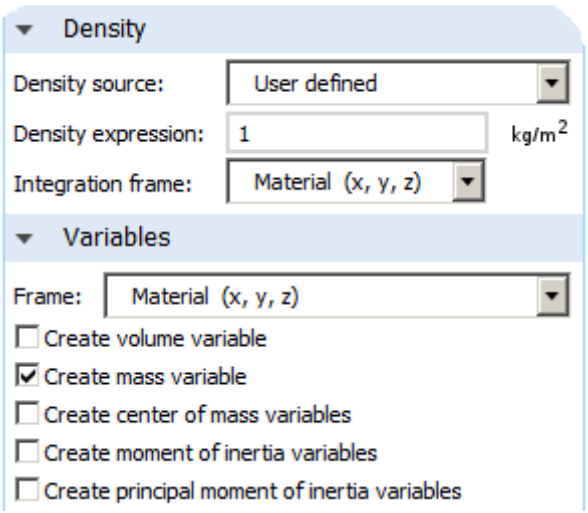
`Density` – **Density source** = `User defined`

– **Density expression** = `1`

Зверніть увагу на одиниці вимірювання поверхневої густини (`kg/m^2`). Це призводить до того, що змінна `mass1.mass` вимірюватиметься в кілограмах (а не в `m^2`, як хотілося б), хоча вона чисельно співпадатиме з площею грані. Для застосування цієї змінної у вигляді площі потрібно буде використовувати запис `mass1.mass[m^2/kg]`.

`Variables` – **Create mass variable** = `True`

Решту прапорців розділу **`Variables`** зніміть.



Змінні маси стають доступними після оновлення обчислень.

Study 1: RMB > Update Solution

Тепер можна будувати графіки теплових потоків

`Results` -> Plot Group - 1D Plot Group

`1D Plot Group 5` -> Add Plot – Global

1D Plot Group 5 - Global 1:

`y-Axis Data` – **Expression, Description** =
`bnd2, Flux from below` (*потік знизу*)

`y-Axis Data` – **Expression, Description** = `bnd1*mass1.mass[m^2/kg], Flux from above` (*потік зверху*)

Остання формула представляє добуток середньої густини теплового потоку на площу грані, тобто буде повним тепловим потоком через верхню грань.

`y-Axis Data` – **Expression, Description** =
`bnd2 + bnd1*mass1.mass[m^2/kg], Total flux` (*сумарний*)

y-Axis Data

Expression	Unit	Description
bnd2	W	Flux from below
bnd1*mass1.mass[m^2/kg]	W	Flux from above
bnd2 + bnd1*mass1.mass[m^2/kg]	W	Total flux

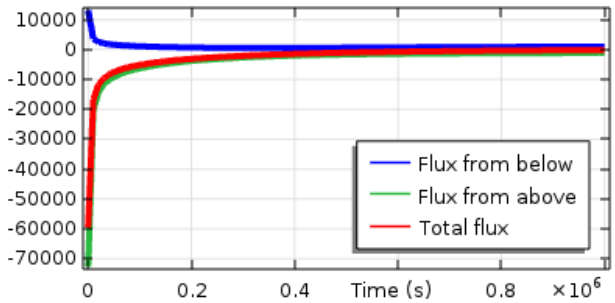
В таблиці зображуваних величин ми ще заповнили стовпець **`Description`**. Його зміст відображатиметься на легенді.

`Coloring and Style` – **`Line style`** – **Width** = 2

1D Plot Group 5 :

`Legends` – **Show legends, Legends** = `True,Manual`
– **Position** = `Lower right`

LMB > Plot



Виконаємо додаткову перевірку потоків, а саме, обчислимо тепловий потік через верхню грань S за формулою $\iint\limits_S \left(-k\frac{dT}{dz}\right)d\sigma$. Для цього в гілці **Derived Values** створіть і обчисліть вузол `Surface Integration`.

Results - Derived Values: RMB > Integration - Surface Integration

Results - Derived Values – Surface Integration 1:

`Data` – **Data set** = `Study 1/Solution 1`

`Selection` – **Selection** = `Manual,4` (*верхня грань*)

`Expression` – **Expression** = `-mat1.def.k11*Tz`

LMB > Evaluate – New Table

З'явиться таблиця **Table 2** з двома стовпцями – моментами часу і відповідними значеннями теплових потоків.

Time (s)	-mat1.def.k11*Tz (W)
0.0000	-78155
10000	-21530
20000	-14980
30000	-12130
40000	-10460
50000	-9297.1
60000	-8429.1
70000	-7772.8
80000	-7230.7
90000	-6768.9
1.0000E5	-6362.1
1.1000E5	

Для побудови по табличних даних графіка теплового потоку натисніть кнопку Table Graph, яка розташована над таблицею. В створену графічну групу **1D Plot Group 6** додайте `Global` вузол з графіком теплового потоку, обчисленим за допомогою **Boundary Probe 1** зонду.

1D Plot Group 6:

`Legends` – **Show legends,** = `False`

1D Plot Group 6: RMB > Global

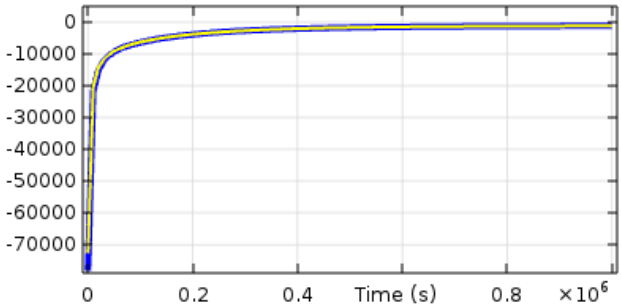
1D Plot Group 6 – Global 1:

`Data` – **Data set** = `Study 1/Solution 1`

`y-Axis Data` – **Expression** = `bnd1*mass1.mass[m^2/kg]`

`Coloring and Style` – **`Line style`** – **Color, Width** = `Yellow,2`

LMB > Plot



Як бачите, криві співпадають.

Для дослідження спрощеної моделі нагрівання/остигання тіла, додайте в поточний файл `Global ODEs and DAEs` інтерфейс.

Component 1: RMB > Add Physics = `Add Physics` -> Mathematics – ODE and DAE Interfaces - Global ODEs and DAEs (ge)->X

В спрощеній моделі $mc\dfrac{dT}{dt}=q$, $T(0)=T_0$ для обчислення теплоємності c можна використати змінну `mat1.def.Cp`, а тепловий потік q дорівнюватиме сумі потоків через верхню і нижню грані (боки тіла теплоізольовані), тобто `q=bnd2+bnd1*mass1.mass[m^2/kg]`. Масу тіла m слід обчислити. Це можна зробити за допомогою змінної маси.

`Definitions` **->Mass properties** – Mass properties

Definitions - Mass Properties 2 (mass2):

Source Selection – **Geometric entity level** = `Domain`
– **Selection** = `All domains,1`

Density – **Density source** = `User defined`
– **Density expression** = `comp1.mat1.def.rho`

Variables – **Create mass variable** = `True`

Решту прапорців розділу **Variables** зніміть.

В ПВ вузла **Global Equations 1** заповніть таблицю рівнянь наступним чином.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name** = `w`
– **f(u, ut, utt, t)** = `mass2.mass*mat1.def.Cp*wt+(bnd2+bnd1*mass1.mass[m^2/kg])`
– **Initial value** = `400[K]`

Global Equations

$f(u,u_t,u_{tt},t)=0,\ u(t_0)=u_0,\ u_t(t_0)=u_{t0}$

Name	f(u,ut,utt,t) (W)	Initial value (u_0) (K)	Initial value (u_t0) (K/s)
w	mass2.mass*mat1.def.Cp*wt+(bnd2+bnd1*mass1.mass[m^2/kg])	400[K]	0

Не звертайте уваги, якщо колір рівняння буде помаранчевим. Після встановлення фізичних одиниць вимірювання в розділі **Units** це повинно зникнути.

Задайте фізичну розмірність шуканої величини `w`, а також рівняння.

Units – **Dependent variable quantity** = `Temperature (K)` (*розмірність w*)
– **Source term quantity** = `None`
– **Unit** = `W` (*розмірність рівняння*)

Units

Dependent variable quantity

Temperature (K)

Source term quantity

None

Unit:

W

Study 1: LMB > Compute

Залишилося побудувати порівняльні графіки розв'язку величини w, яка представляє температуру тіла в спрощеній моделі, і середню температуру тривимірного тіла, значення якої зберігається в змінній dom1.

Results -> Plot Group - 1D Plot Group

1D Plot Group 7:

Data – **Data set** = Study 1/Solution 1

– **Time selection** = All

Legends – **Position** = Lower right

1D Plot Group 7-> Add Plot – Global

1D Plot Group 7 - Global 1:

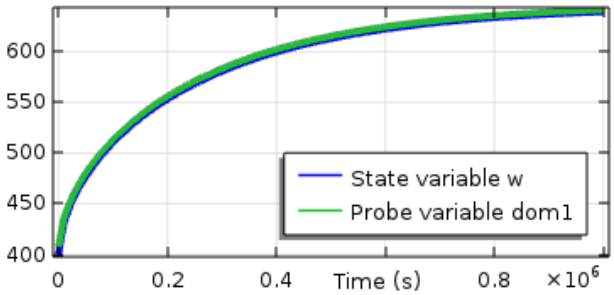
y-Axis Data – **Expression** = w

y-Axis Data – **Expression,** = dom1

x-Axis Data – **Parameter** = Time

Coloring and Style – **Width** = 3

LMB > Plot



Отже спрощена модель дає величину w, яка близька до середньої температури розглядуваного тіла.

7.4. Робота з розривними величинами.

Інколи в COMSOL потрібно обчислювати значення величин по один бік від геометричної границі. Це важливо, коли ви працюєте з тонкими шарами та поверхнями, які розділяють різні фізики або різні матеріали, чи просто маєте справу з розривними величинами або розривними похідними неперервних розв'язків. Також буває необхідним явно виводити результати з різних боків розділювальних примітивів. Для цього можна використовувати вбудовані оператори up і down. Вони призначені для роботи з величинами на примітивах, які мають розмірність на одиницю меншу ніж геометрія моделі. Для розривних величин їх значення різняться з боків стрибка, в той час як COMSOL зазвичай повертає середнє значення виразу в цих місцях. Якщо один з боків розташований зовні геометрії або змінна не визначена на якомусь з боків, то ці оператори повертають 0. Оператор up вибирає значення з боку зовнішньої нормалі до примітива, яка на зовнішніх границях спрямована назовні розрахункової області, а down – з протилежного. Для розрізнення зовнішніх і внутрішніх нормалей в системі визначені вектори (unx,uny,unz) і (dnx,dny,dnz). Їх можна відобразити на лініях та поверхнях за допомогою стрілкового графіка. В двовимірному випадку напрямок «назовні» визначається

як ліворуч відносно напрямку вздовж кривої. Його можна простежити, якщо в ПВ вікна `View` встановити прапорець `Show edge direction arrows`.

Ще одним оператором, призначеним для вибору значення розривної величини з боків стрибка, є оператор `side(num,expr)`. Він теж використовується для отримання значень виразів на примітивах розмірності меншої ніж розрахункова область, і його першим аргументом `num` є індикатор зони, з боку якої буде обчислюватися вираз `expr`. Навпаки, оператор `mean()` обчислює середнє з боків будь якого елементу меншої розмірності, ніж геометрія.

Для демонстрування більшості можливостей операторів, які діють на «розривну» змінну, не має значення, яким чином вона створена – з розв'язку ДРЧП, чи по аналітичним формулам. Тому для спрощення пояснення роботи операторів `up` і `down` буде розглянуто модель, в якій замість розв'язання рівняння будується «розривна» змінна, і на її прикладі розглядаються особливості використання цих операторів.

Приклад 1. Робота з розривним змінними в одновимірній моделі.

Розв'язання. Створіть порожню модель.

File->New -> Blank Model

Додайте в модель одновимірний компонент.

Untitled.mph (root) : RMB > Add Component ->1D

Вимкніть в моделі перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Геометрія моделі складатиметься з об'єднання відрізків $[0,1]\cup[1,2]$.

`Geometry` -> **`Primitives->`** Interval

В ПВ створеного вузла **Interval 1** залиште значення налаштувань за замовчуванням і ще раз повторіть команду.

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 2 (i2):

`Interval` – **Left endpoint, Right endpoint** = `1,2`

Виконайте операцію логічного об'єднання відрізків.

`Geometry` -> **`Booleans and Partitions`** -> Union

Geometry 1 - Union (uni1):

`Union` – **Input objects** = `i1,i2`

– **Keep interior boundaries** = `True`

Закінчіть конструювання геометрії натисканням на кнопку Build All Objects .

Активуйте вузол створеного компонента і виконайте команду:

Component 1:

`Definitions` -> **`Variables`** – Local Variables

В ПВ створеного вузла **Variables 1** введіть:

Definitions - Variables 1:

`Geometric Entity Selection` – **Geometric entity level** = `Domain`

– **Selection - <>** = `1` (*відрізок* [0,1])

Заповніть таблицю змінних так, як зображено на наступному рисунку.

Variables

Name	Expression
U	x^2
U1	d(U,x)

Вибір індикатора 1 означає, що змінні цього вузла діють на лівому відрізку [0,1]. Повторіть команду створення вузла змінних.

`Definitions` -> **`Variables`** – Local Variables

В ПВ створеного вузла **Variables 2** в таблиці змінних введіть:

Definitions - Variables 2:

`Geometric Entity Selection` – **Geometric entity level** = `Domain`

– **Selection - <>** = `2` (*відрізок* [1,2])

Заповніть таблицю змінних так, як зображено на рисунку.

Variables

Name	Expression
U	2*x*(2-x)
U1	d(U,x)

В обох таблицях створені змінні з однаковими іменами, але вони визначені на різних відрізках. Як наслідок, у виразах можна застосовувати одне (спільне) ім'я. Система буде розпізнавати ділянку дії і використовувати відповідну формулу.

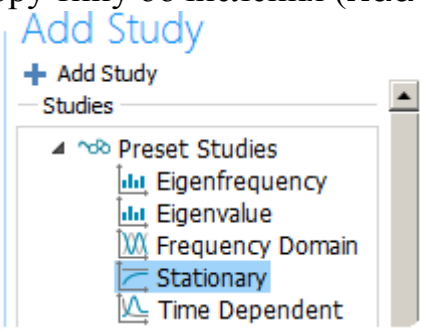
Створіть СкЕ сітку з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Щоб змінні моделі працювали, потрібно виконати обчислення, а для цього треба мати відповідний вузол. Виконайте команду

Untitled.mph (root): RMB > Add Study

Система відкриє вікно вибору типу обчислення (`Add Study`).



Виконайте подвійне клацання миші по рядку `Stationary` і закрийте вікно хрестиком в правому верхньому куті. В дереві моделі з'явиться вузол **Study 1**. Виконайте обчислення моделі командою

`Home` -> Compute

Побудуйте графіки змінних `U` і `U1` на відрізку [0,2]. Для цього виконайте інструкції:

`Results` -> **`Plot Group`** – 1D Plot Group

Results - 1D Plot Group 1:

`Data` – **Data set** = `Study 1/Solution 1`

`1D Plot Group 1` -> **`Add Plot`** – Line Graph

1D Plot Group 1 - Line Graph 1:

Selection – **Selection -** < > = 1,2

y-Axis Data – **Expression** = U

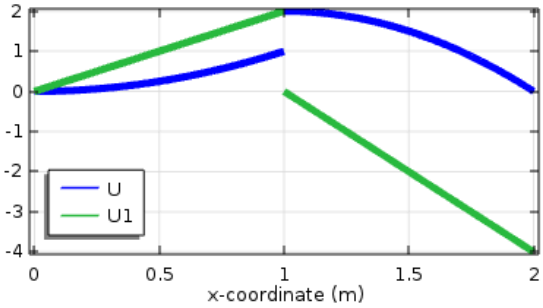
x-Axis Data – **Parameter, Expression** = Expression,x

RMB > Duplicate

1D Plot Group 1 - Line Graph 2:

y-Axis Data – **Expression** = U1

Задайте легенду, налаштуйте стилі оформлення графіків на свій розсуд, і виконайте команду LMB > Plot.



Тепер, використовуючи оператори up і down, надрукуйте значення змінних U і U1 в точці x=1. Для цього застосуйте вузли Global Evaluation. Створіть один такий вузол наступними інструкціями

Results -> **Derived Values** – Global Evaluation

Derived Values - Global Evaluation 1:

Expression – **Expression** = at0(1,U)

LMB > Evaluate

Потім ще 5 разів виконайте команду

Global Evaluation 1 : RMB > Duplicate

Заповніть поля виразів **Expression** створених вузлів **Global Evaluation X** так, як зображено в таблиці. Джерелом даних всіх вузлів буде Study 1/Solution 1.

Вузол	**Expression**	**Пояснення**	**Таблиця**
Global Evaluation 1	at0(1,U)	Середнє значення U	**Table 1**
Global Evaluation 2	at0(1,down(U))	Ліва межа U	**Table 1**
Global Evaluation 3	at0(1,up(U))	Права межа U	**Table 1**
Global Evaluation 4	at0(1,down(U1))	Ліва межа U'_x	**Table 2**
Global Evaluation 5	at0(1,up(U1))	Права межа U'_x	**Table 2**
Global Evaluation 6	at0(1,U1)	Середнє значення U'_x	**Table 2**

Нагадаємо, що оператор at0(1,...) застосовується для вибору значення у вузлі (в точці x=1).

Перші 3 значення надрукуйте в таблиці **Table 1** (ми не описуємо послідовність інструкцій, бо вона очевидна).

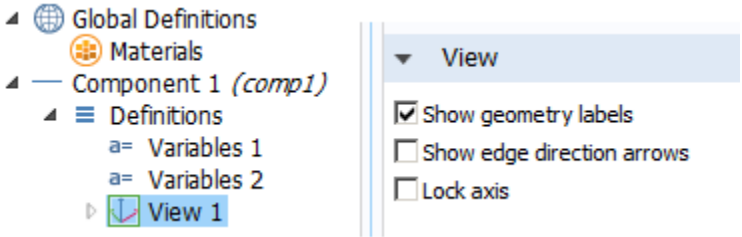
at0(1,U)	at0(1,down(U))	at0(1,up(U))
1.5000	1.0000	2.0000

Останні 3 значення надрукуйте в таблиці **Table 2**.

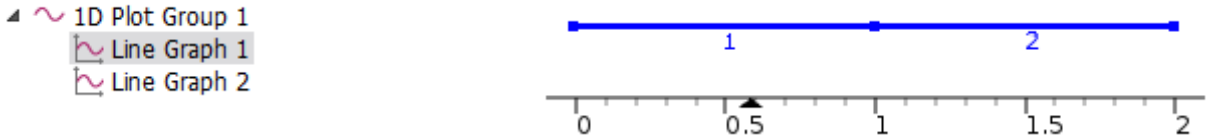
at0(1,down(U1))	at0(1,up(U1))	at0(1,U1)
2.0000	0.0000	1.0000

Як бачите, оператори `up(U)` і `down(U)` вибирають граничні значення полевої змінної `U` (або похідної `U1`) по один та інший бік межі (точка `x=1`). Без використання цих операторів ви отримуєте середнє між лівим і правим значеннями.

Схожим чином працює оператор `side(num,expr)`. Його першим аргументом є номер зони, з боку якої буде обчислюватися вираз `expr`. В нашому прикладі індикатори зон з боків від розділювальної точки можна дізнатися, якщо встановити в ПВ вузла **Component 1 – Definitions – View 1** опцію Show geometry labels = `True`.



Тоді, наприклад, графічне вікно, активоване разом з вузлом **1D Plot Group 1 - Line Graph 1** при включені кнопки вибору Activate, буде містити індикатори зон.



Для друкування значення `U` ліворуч і праворуч від точки `x=1` у вузлах `Global Evaluation` замість виразів `at0(1,down(U))` і `at0(1,up(U))` можна використати вирази `at0(1,side(1,U))` і `at0(1,side(2,U))`. Крім того, замість вузлів типу `Global Evaluation` та оператора `at0` вибору значення в точці, можна застосувати вузли типу `Point Evaluation`, які заміняють виклик оператора `at0`.

Створіть два вузли `Point Evaluation` і заповніть поля виразів так, як показано в наступній таблиці. В їх розділах **`Selection`** вкажіть індикатор `2` (тобто точку `x=1`).

Вузол	**Expression**	**Пояснення**	**Таблиця**
. . .	. . .	. . .	. . .
Point Evaluation 1	`side(1,U)`	Ліва межа U	**Table 3**
Point Evaluation 2	`side(2,U)`	Права межа U	**Table 3**

Надрукуйте їх значення в новій таблиці **Table 3**.

side(1,U), Point: 2	side(2,U), Point: 2
1.0000	2.0000

Вони такі ж самі, що отримані раніше за допомогою вузлів **Global Evaluation 2** та **Global Evaluation 3** з виразами `at0(1,down(U))` і `at0(1,up(U))`. ■

Оператори up і down працюють на геометричних об'єктах розмірність яких на одиницю нижче, ніж розмірність моделі. Наприклад, в тривимірній моделі вирази up(expr) і down(expr) можуть бути використані тільки на двовимірних примітивах, тобто на поверхнях. Оператор side може використовуватися на будь яких примітивах меншої розмірності ніж геометрія моделі. В тривимірній геометрії він може бути використаний для обчислення виразу на одному боці поверхні, що обмежує тіло. Отриманий за його допомогою вираз може бути обчислений на ребрі цієї поверхні, який в свою чергу, може бути обчислений в вершині суміжній до цього ребра.

В наступному прикладі ми, аналогічно до попереднього, створимо змінну, яка визначена різними виразами в різних частинах геометрії, і на її прикладі розглянемо особливості оператора side.

Приклад 2. Дано квадратну область складену з чотирьох менших квадратів.

В кожній з підобластей задано свій вираз для змінної u.

$$u=\begin{cases}(1-x)(1-y), & 0\le x\le 1, 0\le y\le 1\\ (x^2+y^2)/2, & -1\le x\le 0, 0\le y\le 1\\ (1+x^2)(1+y^2)/4, & -1\le x\le 1, -1\le y\le 0\end{cases} \qquad (1)$$

Побудувати графіки граничних значень цієї змінної на лініях розриву і обчислити граничні значення *u* в точці O.

За допомогою оператора side можна отримати граничні значення на внутрішніх ребрах з боку будь якої суміжної зони. Для цього придатні оператори up і down, але за допомогою оператора side також обчислюються граничні значення у вершині O з боку будь-якого з чотирьох внутрішніх ребер.

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Додайте двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Геометрію моделі створіть з об'єднання чотирьох квадратів

$$D=D_1\cup D_2\cup D_3\cup D_4,$$

де $D_1=[0,1]\times[0,1]$, $D_2=[-1,0]\times[0,1]$,

$D_3=[-1,0]\times[-1,0]$, $D_4=[0,1]\times[-1,0]$.

Geometry->**Primitives->** Square

Geometry 1 - Square 1 (sq1) : RMB > Duplicate

Geometry 1 - Square 2 (sq2) :

Position – **x:** = -1

RMB > Duplicate

Geometry 1 - Square 3 (sq3) :

Position – **x:, y:** = -1,-1

RMB > Duplicate

Geometry 1 - Square 4 (sq4) :

Position – **x:, y:** = 0,-1

Виконайте операцію логічного об'єднання квадратів зі збереженням внутрішніх границь.

Geometry -> **Booleans and Partitions** -> Union

Union – **Input objects** = sq1,sq2,sq3,sq4

– **Keep interior boundaries** = True

Імена квадратів (sq1,sq2,sq3,sq4), як об'єктів ДМ, відображаються в графічному вікні під час вибору зон в ПВ створеного вузла **Union 1 (uni1)** при активній кнопці Active.

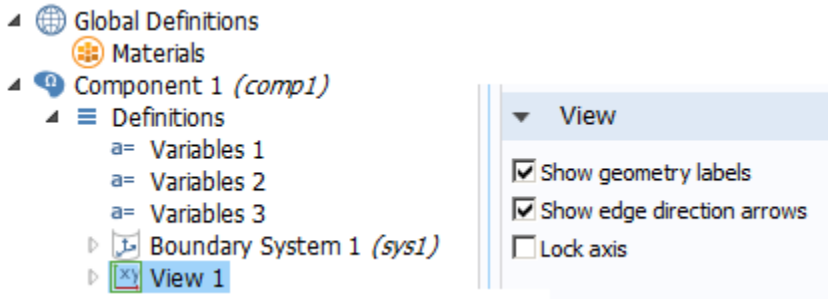
Закінчіть конструювання геометрії натисканням на кнопку Build All Objects .

Щоб номери зон та напрямки на ребрах з'являлися на схемах, в ПВ вузла **View 1** встановіть наступні прапорці.

Component 1 – Definitions – View 1:

Show geometry labels = True

Show edge direction arrows = True



Активуйте вузол компонента **Component 1** і виконайте команду:

Definitions -> **Variables** – Local Variables

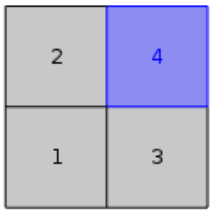
В ПВ створеного вузла **Variables 1** введіть:

Component 1 - Definitions -Variables 1:

Geometric Entity Selection – **Geometric entity level** = Domain

– **Selection** - <> = 4 (*індикатор правого верхнього квадрата*)

Індикатори зон, з яких складається геометрія, відрізняються від номерів квадратів як об'єктів ДМ попереднього рисунка. Індикатори видно в графічному вікні під час вибору квадратів (при активній кнопці Active).



В таблиці змінних вузла **Variables 1** введіть:

Variables – **Name, Expression** = u, (1-x)*(1-y) (*введення в таблиці*)

Вузол **Variables 1** містить значення змінної u у правому верхньому квадраті (sq1). Створіть ще один вузол змінних.

Definitions -> **Variables** – Local Variables

Definitions - Variables 2:

Geometric Entity Selection – **Geometric entity level** = Domain

– **Selection** - <> = 2 (*індикатор лівого верхнього квадрата*)

Variables – **Name, Expression** = u, (x^2+y^2)/2 (*введення в таблиці*)

Вузол **Variables 2** містить значення змінної u у лівому верхньому квадраті (sq2). Створіть третій вузол змінних.

Definitions -> **Variables** – Local Variables

Definitions - Variables 3:

Geometric Entity Selection – **Geometric entity level** = Domain

Geometric Entity Selection – **Selection** - <> = 1,3 (*нижні квадрати*)

Variables – **Name, Expression** = u, (1+x^2)*(1+y^2)/4

Вузол **Variables 3** містить значення змінної u в нижніх двох квадратах.

Змінна u, яка визначена в трьох різних зонах, в результаті задається формулою (1).

Щоб змінні моделі працювали і графікам було на що посилатися (існували джерела даних), створіть і обчисліть вузол досліджень.

Untitled.mph (root): RMB > Add Study = Add Study ->

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Побудуйте графіки змінної u. Для цього виконайте інструкції:

Results -> **Plot Group** ->2D Plot Group

2D Plot Group 1:

2D Plot Group 1-> **Add Plot** -> Surface

2D Plot Group 1 – Surface 1:

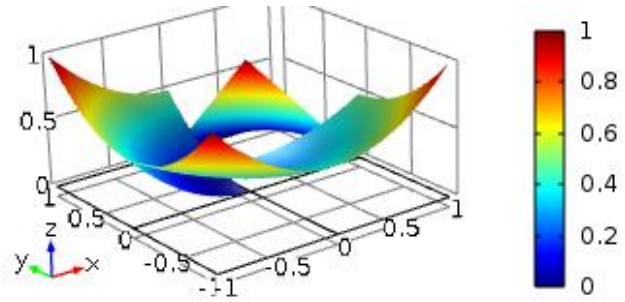
Expression – **Expression** = u

RMB> Height Expression

2D Plot Group 1 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = True,1

LMB>Plot



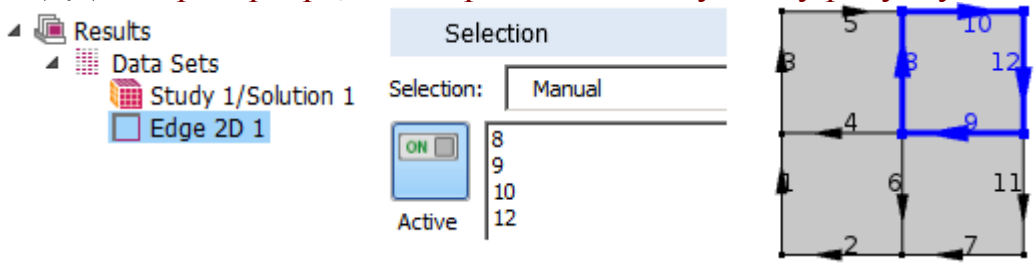
Змінна u задана кусково і зазнає розривів на внутрішніх ребрах геометрії. Спробуємо побудувати графіки однобічних значень вздовж цих ребер. Для цього спочатку потрібно створити реберні джерела даних.

Results – Data Sets : RMB > Edge 2D

Data Sets - Edge 2D 1**:**

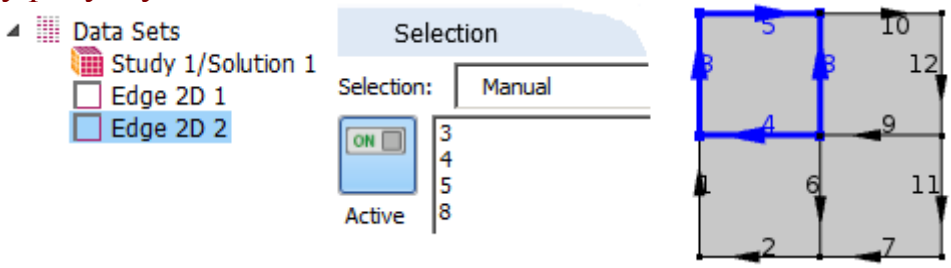
`Data` – **Data set** = `Study 1/Solution 1`

Для цього ДД виберіть ребра, які зображені на наступному рисунку.



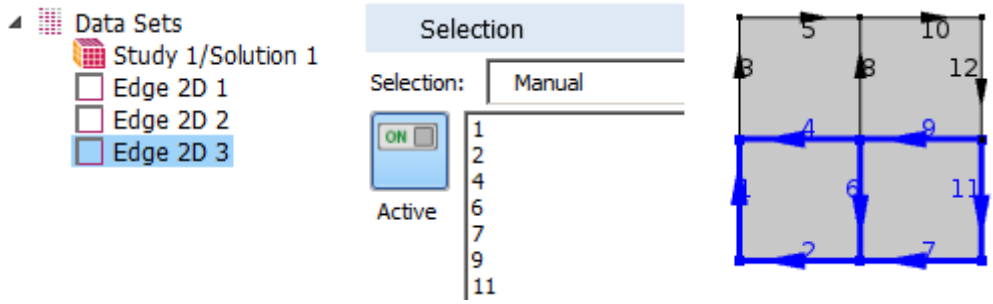
Data Sets - Edge 2D 1: RMB > Duplicate

Для створеного ДД **Data Sets - Edge 2D 2** виберіть ребра, які зображені на наступному рисунку.



Data Sets - Edge 2D 2: RMB > Duplicate

Для ДД **Data Sets - Edge 2D 3** виберіть ребра, які зображені на рисунку.



В двовимірну групу **2D Plot Group 1** додайте і налаштуйте лінійний графік.

2D Plot Group 1:

`2D Plot Group 1`**->** Line

2D Plot Group 1 – Line 1:

`Data` – **Data set** = `Edge 2D 1`

`Expression` – **Expression** = `1`

Coloring and Style – **Line type** = Tube
– **Tube radius expression** = 0.03
– **Radius scale factor** = True,1
– **Coloring, Color** = Uniform, Red

RMB > Height Expression

2D Plot Group 1 – Line 1 – Height Expression 1:

Expression – **Height data** = Expression
– **Expression** = side(4,u)

Axis – **Scale factor** = True, 1

LMB>Plot

2D Plot Group 1 – Line 1: RMB > Duplicate

2D Plot Group 1 – Line 2:

Data – **Data set** = Edge 2D 2

2D Plot Group 1 – Line 2 – Height Expression 1:

Expression – **Expression** = side(2,u)

LMB>Plot

2D Plot Group 1 – Line 2: RMB > Duplicate

2D Plot Group 1 – Line 3:

Data – **Data set** = Edge 2D 3

2D Plot Group 1 – Line 3 – Height Expression 1:

Expression – **Expression** = side(1,u)

LMB>Plot

2D Plot Group 1 – Line 3: RMB > Duplicate

2D Plot Group 1 – Line 4 – Height Expression 1:

Expression – **Expression** = side(3,u)

LMB>Plot

2D Plot Group 1 – Line 4: RMB > Duplicate

2D Plot Group 1 – Line 5:

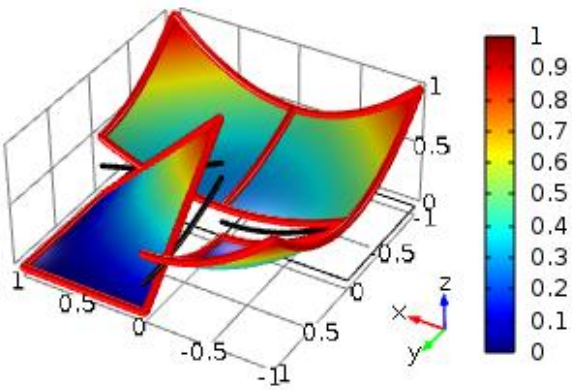
Data – **Data set** = Study 1/Solution 1

Coloring and Style – **Color** = Black

2D Plot Group 1 – Line 5 – Height Expression 1:

Expression – **Expression** = u (*середнє значення на внутрішніх ребрах*)

LMB>Plot



Лінійний графік **Line 5** будує вираз u на всіх ребрах (внутрішніх і зовнішніх). Але на внутрішніх ребрах, де змінна u зазнає розриву, будуються середні значення. На рисунку вони зображені чорними відрізками, які не примикають до сусідніх внутрішніх шматків поверхні. Зовні значення u співпадає з відповідними виразам side(...,u).

Нагадаємо, що можна замінити вирази з оператором side на оператори up і down. Але пам'ятайте, як різняться «верх» і «низ». В двовимірних геометріях верх до кривої спрямований ліворуч від напрямку вздовж неї. На малюнках джерел даних Edge 2D в режимі вибору відрізків цей напрямок відображається в графічному вікні. До того ж, для побудов графіків граничних значень вам доведеться використати більше виразів з up і down операторами, і отже, доведеться будувати більше вузлів типу Line.

Використовуючи оператор side, ми змогли виділити однобічні граничні значення змінної u. Побудуйте ще лінійні графіки однобічних граничних значень змінної u, використовуючи оператори up і down , наприклад, вздовж прямої y=0.

Results -> **Plot Group** – 1D Plot Group

Results - 1D Plot Group 2:

Data – **Data set** = Study 1/Solution 1

1D Plot Group 1 -> **Add Plot** – Line Graph

1D Plot Group 2 - Line Graph 1:

Selection – **Selection** - < > = 9,4 (*індикатори відрізків прямої y*=0)

y-Axis Data – **Expression** = u

x-Axis Data – **Parameter, Expression** = Expression, x

RMB > Duplicate

1D Plot Group 2 - Line Graph 2:

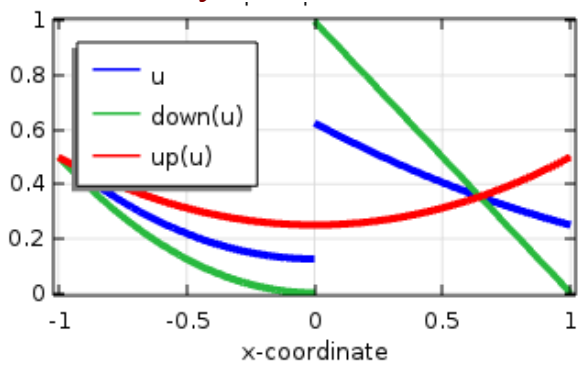
y-Axis Data – **Expression** = down(u)

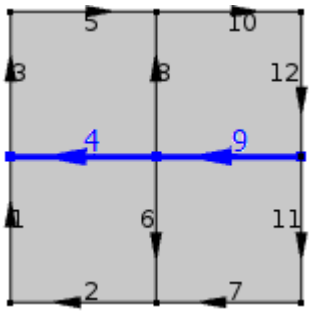
RMB > Duplicate

1D Plot Group 2 - Line Graph 3:

y-Axis Data – **Expression** = up(u)

Задайте легенду, налаштуйте стилі оформлення графіків на свій розсуд, і виконайте команду LMB > Plot.





Синя крива зображає змінну u, яка представляє середнє значення зліва і справа від ребра 4 – 9 (зверху і знизу на правому рисунку). Червона (неперервна крива)

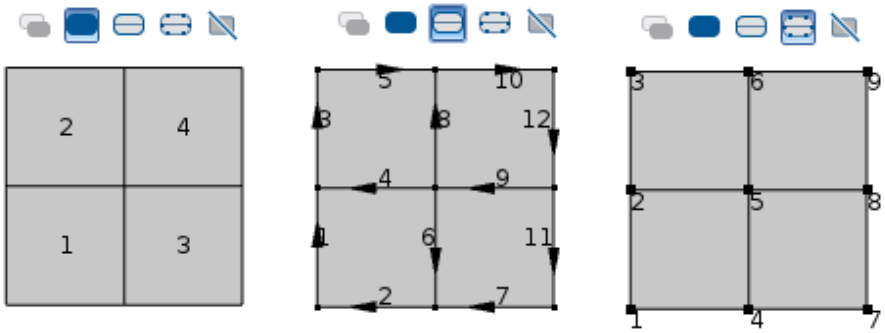
зображає значення `up(u)`, яке на правому рисунку є граничним знизу. Зелена – зображає значення `down(u)`, яке на правому рисунку є граничним зверху.

Тепер обчисліть значення в центральній точці, яке залежить від того, з якого боку ми до неї наближаємось. В цьому випадку використання операторів `up` і `down` неможливо. Тому будемо користуватися лише оператором `side`, для якого знадобляться номери зон, ребер та точок. При встановлених налаштуваннях у ПВ вузла **View 1**

Show geometry labels = `True`

Show edge direction arrows = `True`

нумерацію геометричних примітивів можна відобразити в графічному вікні, коли активний один з вузлів `Global Evaluation`. Створіть такий вузол самостійно (якщо його ще немає в моделі). Потім клацніть мишею по одному з вузлів `Variables`, а потім знову по вузлу `Global Evaluation`. Тоді якщо на панелі інструментів графічного вікна активувати кнопку Select Domains, то будуть показані індикатори зон (наступний рисунок ліворуч). Якщо активувати кнопку Select Boudries, то будуть показані індикатори одновимірних примітивів і їхні напрямки (рисунок всередині). Якщо активувати кнопку Select Points, то будуть показані індикатори вершин (рисунок праворуч).



При активації інших вузлів (не `Global Evaluation`) часто доступна для відображення лише одна з зазначених можливостей нумерації. Наприклад, при активному вузлі `Point Evaluation` можна побачити лише нумерацію точок.

Обчисліть границю в центральній вершині з боку зони 4. З формули (1) маємо $\lim\limits_{x\to+0}\lim\limits_{y\to+0} u = \lim\limits_{x\to+0}\lim\limits_{y\to+0}(1-x)(1-y)=1$. Зробіть ці обчислення в COMSOL.

`Results` -> **`Derived Values`** – Point Evaluation

Derived Values - Point Evaluation 1:

`Selection` – **Selection** = `Manual,5` (*індикатор центральної точки*)

`Expression` – **Expression** = `side(9,side(4,u))`

LMB > Evaluate

В таблиці **Table 1** друкується значення 1. Тут ми наближалися до точки (0,0) вздовж ребра 9 з боку зони 4. Таке ж значення ви отримаєте, якщо будете наближатися до вершини вздовж ребра 8 з боку тієї ж зони 4. Спробуйте замінити вираз на `side(8,side(4,u))` і ви повинні отримати той же результат. Зауважимо, що у виразі `side(9,side(4,u))` зовнішня цифра `9` є індикатором внутрішнього ребра, а внутрішня цифра `4` є індикатором квадрата.

Обчисліть границю в тій же вершині вздовж ребра 9 з боку зони 3.

Derived Values - Point Evaluation 1: RMB > Duplicate
Derived Values - Point Evaluation 2:

Expression – **Expression** = `side(9,side(3,u))`

LMB > Evaluate - Table 1

В таблиці **Table 1** буде надруковано значення `0.25`. Дійсно маємо

$$\lim_{x\to+0}\lim_{y\to-0} u = \lim_{x\to+0}\lim_{y\to-0}\left(1+x^2\right)\left(1+y^2\right)/4 = 1/4.$$

Обчисліть границю в вершині `(0,0)` вздовж ребра 4 з боку зони 2.

Derived Values - Point Evaluation 2: RMB > Duplicate
Derived Values - Point Evaluation 3:

Expression – **Expression** = `side(4,side(2,u))`

LMB > Evaluate - Table 1

В таблиці **Table 1** буде надруковано значення `0.0`. Дійсно маємо

$$\lim_{x\to-0}\lim_{y\to+0} u = \lim_{x\to-0}\lim_{y\to+0}\left(x^2+y^2\right)/2 = 0.$$

Таким чином, в нашій моделі змінна `u` може мати 3 різних граничних значення в вершині (`0,0`): `1`, `0.25` і `0`. До речі, це видно з графіка поверхні змінної `u`, яку ми побудували раніше. Якщо не подвоювати оператор `side`, то можна отримати середнє граничне значення вздовж ребер. Наприклад, якщо для тієї ж вершини `5` використати вираз `side(9,u)`, то буде повернуто результат `0.625`, який є середнім (зверху і знизу) вздовж ребра 9 при наближенні до вершини справа. Дійсно маємо

`1.0` (границя вздовж ребра 9 збоку зони 4) +
`0.25` (границя вздовж ребра 9 збоку зони 3) =`1.25`

Оператор `side(9,u)` повертає середнє `1.25/2=0.625`.

Якщо в вершині 5 просто надрукувати значення `u`, то буде отримано середнє значення.

Derived Values - Point Evaluation 3: RMB > Duplicate
Derived Values - Point Evaluation 4:

Expression – **Expression** = `u`

LMB > Evaluate - Table 1

В таблиці `Table 1` буде надруковано число `0.375`, яке може буде обчислено як середнє між граничними значеннями, що отримані при наближенні до вершини з усіх чотирьох зон `(1+0+0.25+0.25)/4=0.375`.

Інший спосіб отримати середнє полягає у використанні оператора `mean(expr)`, який обчислює середнє значення виразу `expr` в точці, розглядаючи її як вершину графічних примітивів, що з нею межують.

Derived Values - Point Evaluation 4: RMB > Duplicate
Derived Values - Point Evaluation 5:

Expression – **Expression** = `mean(u)`

LMB > Evaluate - Table 1

Отримане значення `0.375` співпадає зі значенням `u` в цій вершині, яке також обчислюється як середнє.

Збережіть модель в файл, наприклад з ім'ям `Demo_sides2D.mph`.

7.5. Полеві змінні на межі розрахункових областей.

В COMSOL полевими називаються змінні, які залежать від просторових координат. Ними є самі просторові координати `x,y,z`, шукані величини, а також будь-які змінні, що отримані з них шляхом диференціювання і виконання інших математичних операцій.

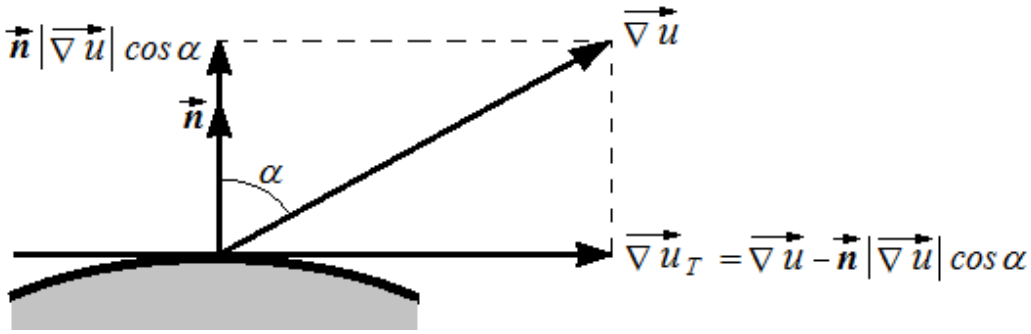
Шукані величини в МСкЕ розробники називають змінними функцій форми. Функції форми – це поліноміальні функції, які застосовуються в скінчених елементах для представлення розв'язків. Змінні функцій форми це залежні величини, які апроксимуються наборами функцій форми. Інакше кажучи, змінні функцій форми будуються на СкЕ сітці. Змінні, що задаються аналітичними виразами у вузлах Variables з використанням просторових координат, не є такими. Протягом всіх попередніх глав ми використовували різноманітні полеві змінні (які були функціями форми або ні) і багато говорити про них не будемо. Ми лише торкнемося питання обчислення похідних на зовнішніх межах областей.

Змінні функцій форми поза розрахунковою геометрією зазвичай невизначені. Тому обчислювати їх похідні на межі областей неможливо. Але вздовж межі вони змінюються і похідні в дотичних напрямках існують. Крім того, бувають ситуації, коли величина взагалі визначена лише на межі, але швидкість її зміни потрібно обчислювати. Для всіх цих випадків призначені змінні «дотичних похідних» `uTx`, `uTy`, `uTz` (`u` – ім'я шуканої змінної) і оператор `dtang(expr,xi)`, який всередині областей співпадає із звичайним оператором `d(expr,xi)`. Другий аргумент оператора `dtang` завжди є однією з просторових координат.

В 3D моделях вектор `(dtang(f,x),dtang(f,y),dtang(f,z))` є дотичним до поверхні в точці, де він обчислюється. Аналогічно в 2D моделях `(dtang(f,x),dtang(f,y))` є дотичним вектором до граничної кривої. «Дотичне диференціювання» підкоряється більшості загальних правил диференціювання, наприклад, при обчисленні похідних суми або добутку виразів. Однак слід зазначити, що `dtang(x,x)` на криволінійній межі зазвичай не дорівнює `1`.

Оператор `dtang` можна застосовувати до змінних функцій форми і до полевих змінних, які ними не є. Похідні `uTx`, `uTy`, `uTz` визначаються лише для змінних функцій форми.

Пояснимо сенс тангенціальних/дотичних похідних. Тангенціальні похідні $\nabla u_T = (uTx, uTy, uTz)$ змінної u функцій форми являють собою координати проекції вектора її градієнта ∇u на дотичну площину (або пряму). Наступний рисунок це ілюструє.



На рисунку $\vec{\nabla u}$ позначає градієнт величини u, $\vec{n}$ – одиничний вектор нормалі до межі. Оскільки $|\nabla u|\cos\alpha = (\nabla u, \boldsymbol{n})$, де $(\nabla u, \boldsymbol{n})$ – скалярний добуток векторів ∇u і $\boldsymbol{n}$, то

$$\nabla u_T = \nabla u - (\nabla u, \boldsymbol{n})\boldsymbol{n}. \qquad (1)$$

Координати вектора $\nabla u_T = (uTx, uTy, uTz)$ називають дотичними похідними величини u (на межі тіла).

Розглянемо приклад, в якому знайдемо координати вектора (1), і порівняємо їх з дотичними похідними (uTx,uTy,uTz), обчислюваними системою. Звичайно, що коли все зробити правильно, то вони співпадуть. Для наочності оберемо двовимірну модель, в якій розв'яжемо рівняння Пуассона в еліпсі відносно шуканої величини u, яка буде змінною функцій форми, і побудуємо графіки її «інших похідних» на зовнішній межі, зокрема (ux,uy), (d(u,x),d(u,y)), (dtang(u,x), dtang(u,y)). Те саме (графічне порівняння похідних) зробимо для величини U=1-x^2/2+y^2/2, яка не є змінною функцій форми.

Приклад 1. Похідні на межі плоскої фігури.

Розв'язання. За допомогою майстра створіть 2D модель задачі.

File->New -> Model Wizard -> 2D -> Mathematics - Classical PDEs - Poisson's Equation (poeq) -> Stationary

Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Розрахунковою геометрією буде еліпс.

Geometry -> **Primitives** – Ellipse

Geometry 1 - Ellipse 1 (e1):

Size and Shape – **a-semiaxis, b-semiaxis** = 2,1

LMB > Build All Objects

Побудуйте СкЕ сітку з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Задайте коефіцієнти рівняння, залишивши зоною дії увесь еліпс.

Poisson's Equation 1:

Diffusion Coefficient – **c** <> = 1

Source Term – **f** <> = -1

Задайте граничну умову Діріхле.

Physics -> **Boundary** -> **Boundaries** -> Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection**=All boundaries,1,2,3,4

Dirichlet Boundary Condition – **r** = x^2-y^2/2

Виконайте обчислення.

Home -> Compute

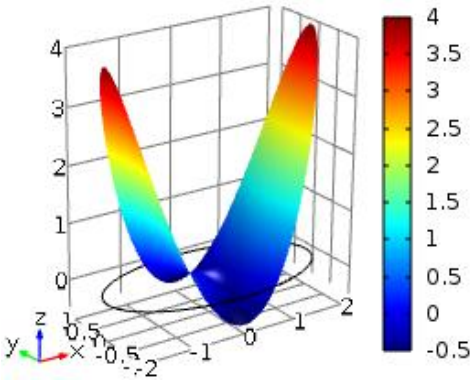
В гілці **Results** з'явиться графічна група **2D Plot Group 1** з вузлом **Surface 1**, що містить кольоровий графік розв'язку. Додайте до нього вертикальну деформацію.

2D Plot Group 1 – Surface 1: RMB> Height Expression

2D Plot Group 1 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = True,1

LMB>Plot



Побудуйте графіки дотичних похідних на межі еліпса.

Results -> **Plot Group** – 1D Plot Group

1D Plot Group 2 -> **Add Plot** – Line Graph

1D Plot Group 2 - Line Graph 1:

Selection – **Selection** - < > = Manual,1,2,3,4 (*всі дуги еліпса*)

y-Axis Data – **Expression** = uTx

x-Axis Data – **Parameter** = Arc length

Coloring and Style –**Width** = 3

Legends – **Show legends , Legends** = True,Manual

– **Legends (таблиця)** = uTx

1D Plot Group 2 - Line Graph 1: RMB > Duplicate

1D Plot Group 2 - Line Graph 2:

y-Axis Data – **Expression** = uTy

Legends – **Legends (таблиця)** = uTy

LMB > Plot (*наступний рисунок ліворуч*)

Обчисліть і намалюйте графіки координат вектора ∇u_T, обчислюваного за формулою (1). Для цього в моделі задайте відповідні змінні.

Definitions -> **Variables** – Local Variables

Definitions – Variables 1:

Geometric Entity Selection – **Geometric entity level** = Boundary

– **Selection** = All boundaries

Variables – **Name, Expression** = k, ux*nx+uy*ny

Variables – **Name, Expression** = Tux, ux-k*nx

Variables – **Name, Expression** = `Tuy, uy-k*ny`

Створені тут змінні `Tux`, `Tuy` визначені лише на межі еліпса.

Поновіть обчислення і додайте графіки змінних в гілку **1D Plot Group 2**.

Study 1: RMB > Update Solution

1D Plot Group 2 - Line Graph 1: RMB > Duplicate

1D Plot Group 2 - Line Graph 3:

y-Axis Data – **Expression** = `Tux`

Coloring and Style – **Color** = `Yellow`

– **Width** = `2`

Legends – **Show legends** = `False`

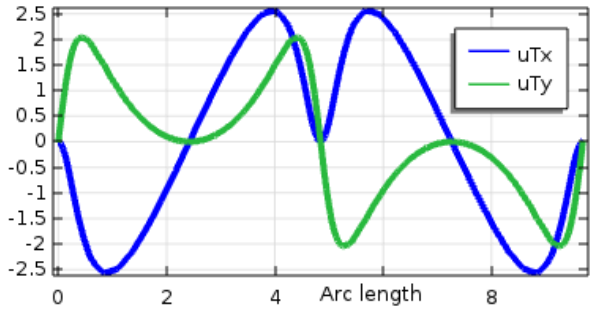
1D Plot Group 2 - Line Graph 3: RMB > Duplicate

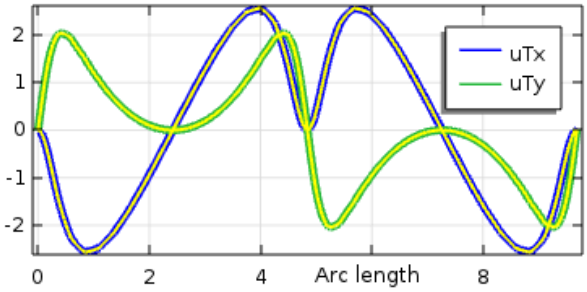
1D Plot Group 2 - Line Graph 4:

y-Axis Data – **Expression** = `Tuy`

LMB > Plot

(*наступний рисунок праворуч*)





Як бачите, криві співпадають.

З'ясуємо поведінку похідних `ux`, `uy` і «внутрішніх граничних похідних» `down(d(u,x))`, `down(d(u,y))`, додавши їх графіки в гілку **1D Plot Group 2**.

1D Plot Group 2 - Line Graph 1: RMB > Duplicate

1D Plot Group 2 - Line Graph 5:

y-Axis Data – **Expression** = `ux`

Legends – **Legends (таблиця)** = `ux`

1D Plot Group 2 - Line Graph 5: RMB > Duplicate

1D Plot Group 2 - Line Graph 6:

y-Axis Data – **Expression** = `uy`

Legends – **Legends (таблиця)** = `uy`

1D Plot Group 2 - Line Graph 3: RMB > Duplicate

1D Plot Group 2 - Line Graph 7:

y-Axis Data – **Expression** = `down(d(u,x))`

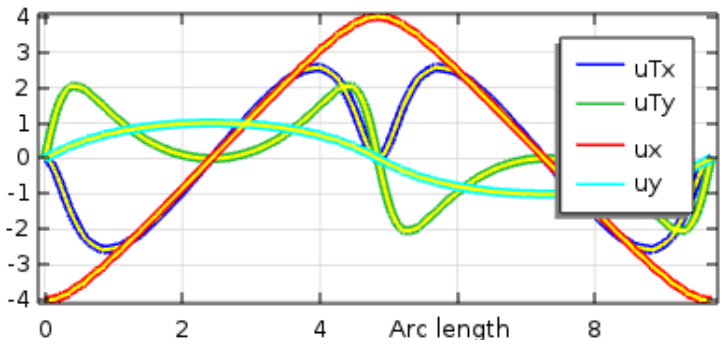
Legends – **Show legends** = `False`

1D Plot Group 2 - Line Graph 7: RMB > Duplicate

1D Plot Group 2 - Line Graph 8:

y-Axis Data – **Expression** = `down(d(u,y))`

LMB > Plot



Отже `ux=down(d(u,x))` і `uy=down(d(u,y))`. Але похідні `d(u,x)`, `d(u,y)`, `up(d(u,x))`, `up(d(u,y))` дорівнюють нулю. Перевірте це самостійно, додавши відповідні графіки в гілку **1D Plot Group 2**.

Побудуйте графіки векторних полів (`uTx,uTy`) і (`ux,uy`) на межі геометрії. Для цього спочатку створіть ДД, яке співпадає з контуром фігури.

`Results` -> More Data Sets – Edge 2D

Data Sets - Edge 2D 1 :

`Data` – **Data set** = `Study 1/Solution 1`

`Selection` – **Selection** = `All boundaries`

`Results` -> **`Plot Group`** – 2D Plot Group

2D Plot Group 3:

`Data` – **Data set** = `Edge 2D 1`

`2D Plot Group 3` -> Add Plot – Arrow Line

2D Plot Group 3 - Arrow Line 1 :

`Expression` – **x component, y component** = `uTx,uTy`

`Coloring and Style` – **Scale factor** = `True,0.4`

– **Number of arrows** =`30`

LMB > Plot (*наступний рисунок ліворуч*)

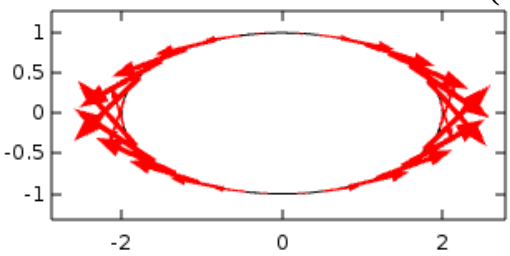
2D Plot Group 3 - Arrow Line 1 : RMB > Duplicate

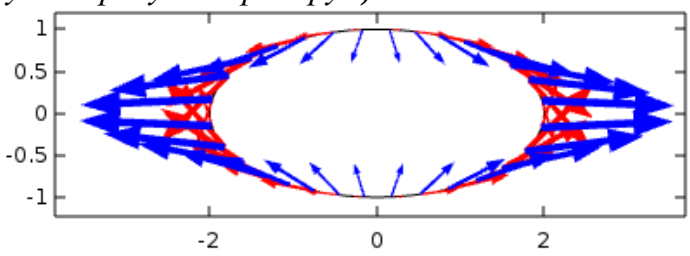
2D Plot Group 3 - Arrow Line 2 :

`Expression` – **x component, y component** = `ux,uy`

`Coloring and Style` – **Color** =`Blue`

LMB > Plot (*наступний рисунок праворуч*)

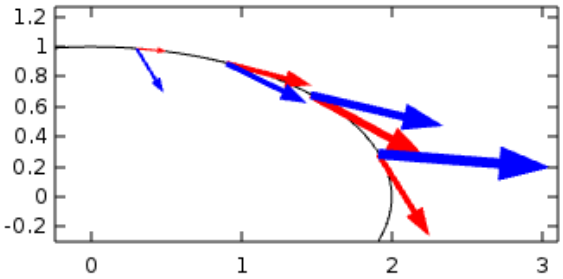




Природно, що векторне поле (`uTx,uTy`) є дотичним до еліпса. В той же час вектор (`ux,uy`) зображає градієнт розв'язку `u` (`u` є змінної функцій форми), обчислений на межі з середини еліпса.

З останнього рисунка незрозуміло, чи є вектори (`uTx,uTy`) проекцією градієнта (`ux,uy`) на дотичні до еліпса. Але в цьому можна наочно упевнитися, якщо намалювати невелику кількість стрілок. Спробуйте самостійно побудувати наступний рисунок. Для цього створіть ДД типу `Edge 2D` лише для

однієї дуги еліпса (з чотирьох) і у вузлах `Arrow Line` вкажіть невелику кількість стрілок, наприклад, чотири.



Перевірте самостійно, що на межі еліпса `(uTx,uTy)=(dtang(u,x), dtang(u,y))`. Для цього можна в графічну групу **2D Plot Group 3** додати зображення векторного поля `(dtang(u,x), dtang(u,y))` на контурі еліпса, яке співпаде з зображенням поля `(uTx,uTy)`. Можна також в групі **1D Plot Group 2** добудувати лінійні графіки величин `dtang(u,x)`, `dtang(u,y)` які, вочевидь, співпадуть з побудованими там кривими `uTx` і `uTy`.

Всі використані вище похідні відносилися до шуканої величини `u`, яка є змінної функцій форми. Розглянемо тепер величину, яка не є змінною функцій форми. Спочатку створіть її в новій таблиці змінних, обравши зоною дії всю модель.

`Definitions` **`->Variables`** – Local Variables

Component 1 - Definitions – Variables 2:

`Geometric Entity Selection` – **Geometric entity level** = `Entire model`

`Variables` – **Name, Expression** = `U, 1-x^2/2+y^2/2`

Змінну `U` ми визначаємо на всій моделі, тобто всередині еліпса і на його межі, а зовні еліпса змінна не визначена.

Поновіть обчислення і побудуйте графік змінної `U`.

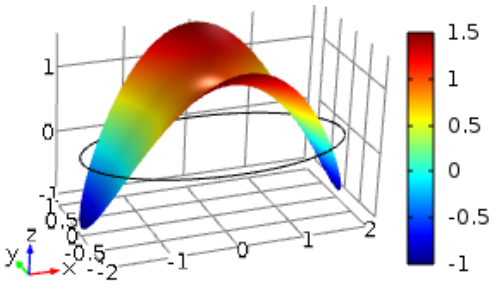
Study 1: RMB > Update Solution

2D Plot Group 1: RMB > Duplicate

2D Plot Group 4 - Surface 1:

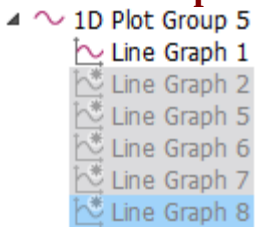
`Expression` – **Expression** = `U`

LMB > Plot

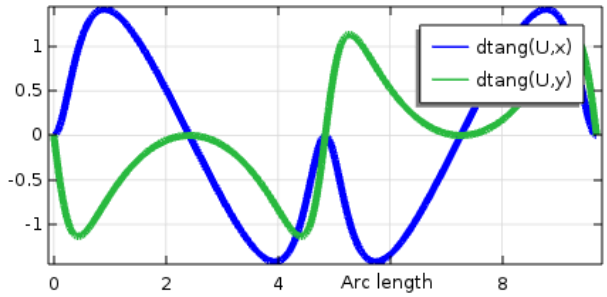


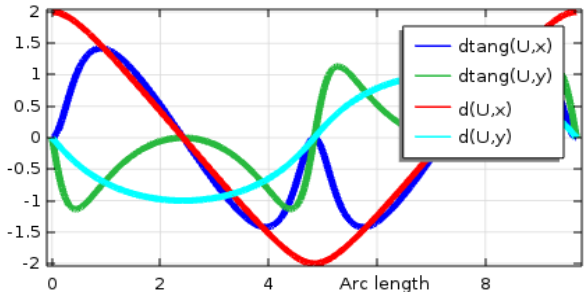
Оскільки `U` не є змінною функцій форми, то для неї величини `Ux`,`Uy` і `UTx`, `UTy` невизначені, але замість них можна використовувати похідні `d(U,x)`, `d(U,y)` і оператор «дотичного диференціювання» `dtang(U,x)`, `dtang(U,y)`. Побудуйте графіки цих величин, взявши більшу частину налаштувань з графічної групи **1D Plot Group 2**. Зайві графічні вузли видаліть, а інші, поки вони не налаштовані, деактивуйте.

1D Plot Group 2: RMB > Duplicate
1D Plot Group 5 - Line Graph 2: RMB > Disable
Видаляти і деактивувати вузли можна групами. Для цього потрібно, утримуючи клавішу Shift, клацнути мишею по першому вузлу з групи, що видалятиметься, потім клацнути по останньому вузлу групи. Колір групи в ДМ зблідне. Відпустивши клавішу Shift, клацніть по групі правою кнопкою миші, і з випадного меню оберіть бажану команду Delete (або Disable). При стислому запису цих дій назви вузлів групи будемо обмежувати кутовими дужками < >, в яких перелічуватимемо вузли, або між позначкою першого і останнього вузла писатиме тире. Отже інструкцію видалення вузлів **Line Graph 3**, **Line Graph 4** ми позначатимемо наступним чином:
1D Plot Group 5 - < Line Graph 3, Line Graph 4> : RMB > Delete
Інструкцію деактивації групи прилеглих вузлів записуватимемо наступним чином.
1D Plot Group 5 - <Line Graph 5 - Line Graph 8> : RMB > Disable



Тепер мінятимемо налаштування вузлів продубльованої графічної гілки. Спочатку накресліть графіки величин `dtang(U,x)` і `dtang(U,y)`.
1D Plot Group 5 - Line Graph 1:
y-Axis Data – Expression = `dtang(U,x)`
Legends – Legends (таблиця) = `dtang(U,x)`
1D Plot Group 5 - Line Graph 2:
y-Axis Data – Expression = `dtang(U,y)`
Legends – Legends (таблиця) = `dtang(U,y)`
LMB > Plot (*наступний рисунок ліворуч*)
Потім побудуйте графіки величин `d(U,x)`, `d(U,y)`.
1D Plot Group 5 - Line Graph 5:
y-Axis Data – Expression = `d(U,x)`
Legends – Legends (таблиця) = `d(U,x)`
RMB > Enable
1D Plot Group 5 - Line Graph 6:
y-Axis Data – Expression = `d(U,y)`
Legends – Legends (таблиця) = `d(U,y)`
LMB > Plot (*наступний рисунок праворуч*)





Перевірте, що величини `d(U,x)`, `d(U,y)` співпадають з «внутрішніми похідними» `down(d(U,x))`, `down(d(U,y))`. Для цього добудуйте їх графіки в поточній гілці ДМ.

1D Plot Group 5 - Line Graph 7:

`y-Axis Data` – Expression = `down(d(U,x))`

`Legends` – Show legends = `False`

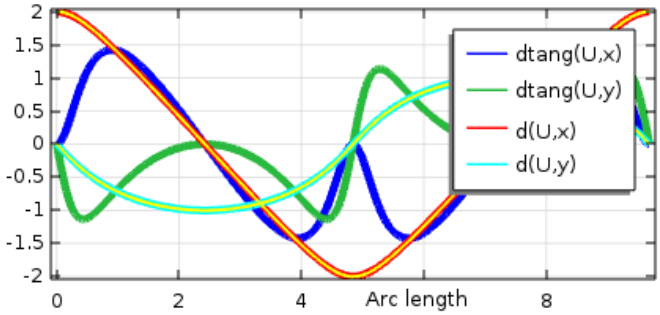
RMB > Enable

1D Plot Group 5 - Line Graph 8:

`y-Axis Data` – Expression = `down(d(U,y))`

`Legends` – Show legends = `False`

LMB > Plot

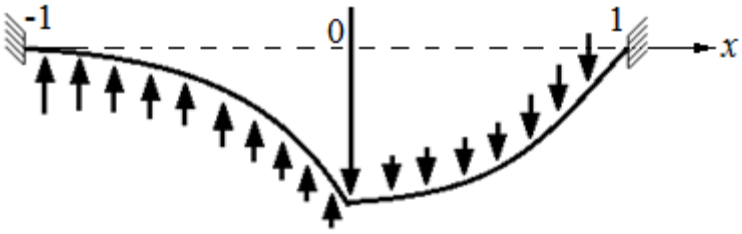


Відповідні криві співпали. Якщо в виразах останніх двох вузлів **Line Graph 7** і **Line Graph 8** замінити оператор `down(...)` на `up(...)`, то отримаємо графіки тотожних нулів. Це означає що зовні еліпса змінна `U` невизначена і її «зовнішні похідні» покладаються нулями.

Приклад 2. На відрізку [-1,1] відшукати неперервний розв'язок крайової задачі

$$u''_{xx} = x + \delta(x),\ -1 \le x \le 1,\ u(-1) = 0,\ u(1) = 0.$$

Вона моделює натягнуту закріплену з двох кінців струну, в точці `x=0` якої прикладена одинична зосереджена сила, зображувана в рівнянні дельта-функцією Дірака $\delta(x)$.



Аналітичний розв'язок цієї задачі відомий

$$u(x) = \frac{x^3 - 3 - x + 3|x|}{6}, \qquad (2)$$

і ми зможемо порівняти його з тим, який відшукає система.

Р о з в ' я з а н н я . За допомогою майстра створіть 1D модель задачі.

File->New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Poisson's Equation (poeq) -> Stationary

Вимкніть контроль розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Геометрія моделі складатиметься з відрізка [-1,1] і точки $x_0 = 0$, в якій буде прикладатися зосереджена сила. Створіть їх.

Geometry -> **Primitives->** Interval

Geometry 1 - Interval 1 (i1):

Interval – **Left endpoint, Right endpoint** =-1,1

Geometry -> **Primitives** -> Point

Завершіть конструювання геометрії натисканням на кнопку Build All Objects .

Створіть СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Задайте параметри рівняння.

Poisson's Equation(poeq) :

Domain Selection – **Selection** = All domains

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = x (*розподілена зовнішня навантага*)

Додайте в гілку фізики **Poisson's Equation (poeq)** вузол типу Flux/Source, який буде представляти зосереджену силу.

Physics -> **Boundaries** – Flux/Source

Poisson's Equation (poeq) - Flux/Source 1:

Boundary Selection – **Selection** = Manual,2 (*індикатор точки* x=0)

Boundary Flux/Source – **g** = 1

Задайте граничні умови (закріплення).

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,1,3 (*кінці відрізка* [-1,1])

Dirichlet Boundary Condition – **r** = 0

Виконайте обчислення.

Home -> Compute

В графічній зоні вузла **1D Plot Group 1 - Line Graph 1** буде побудована крива розв'язку. Налаштуйте її стиль.

1D Plot Group 1 - Line Graph 1:

Coloring and Style – **Width** = 3

Для порівняння побудуйте графік розв'язку (2).

1D Plot Group 1 -> **Add Plot** – Line Graph

1D Plot Group 1 - Line Graph 2:

Selection – **Selection** - < > = Manual,1,2 (*відрізки* [-1,0] і [0,1])

y-Axis Data – **Expression** = (-3+x^3-x+3*abs(x))/6

x-Axis Data – **Parameter, Expression** = Expression, x

Coloring and Style – **Color** = Yellow

– **Width** = 2

LMB > Plot (*наступний рисунок ліворуч*)

Як бачите, графіки співпадають.

Коли в системі розв'язується задача методом скінченних елементів, то одночасно з розв'язком u створюється змінна ux, яка представляє його похідну. Побудуйте графік похідної двома способами.

1D Plot Group 1 - Line Graph 1 - RMB > Duplicate

1D Plot Group 1 - Line Graph 3:

y-Axis Data – **Expression** = ux

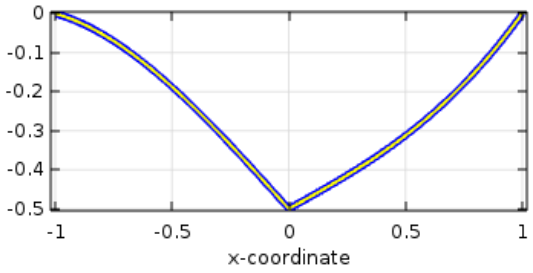
1D Plot Group 1 - Line Graph 2 - RMB > Duplicate

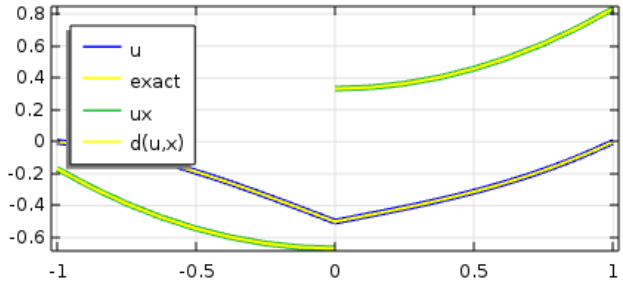
1D Plot Group 1 - Line Graph 4:

y-Axis Data – **Expression** = d(u,x)

Легенди графіків налаштуйте самостійно.

LMB > Plot (*наступний рисунок праворуч*)





Отже всередині відрізків [-1,0] і [0,1] маємо рівність ux=d(u,x).

Порівняйте значення похідних ux і d(u,x) в точці x=0. Для цього використайте вузли Point Evaluation. Створіть один такий вузол наступними інструкціями

Results -> **Derived Values** – Point Evaluation

Derived Values - Point Evaluation 1:

Selection – **Selection** = Manual,2 (*індикатор центральної точки*)

Expression – **Expression** = ux

LMB > Evaluate

Потім 7 разів виконайте команду дублювання створеного вузла.

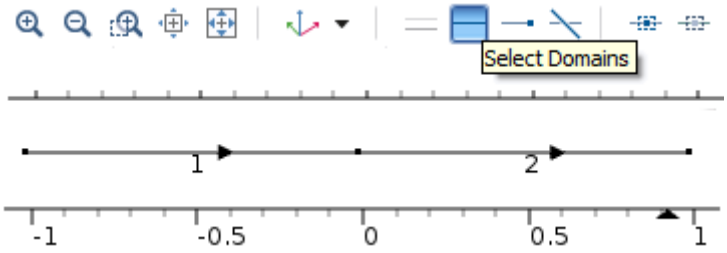
Point Evaluation 1 : RMB > Duplicate (*виконати 7 разів*)

Заповніть поля виразів **Expression** створених вузлів Point Evaluation так, як зображено в наступній таблиці. В трьох останніх вузлах обчислюватиметься друга похідна в точці x=0.

Вузол	**Expression**	**Відповідь**	**Пояснення**
Point Evaluation 1	ux	-0.16797	Середнє значення ux
Point Evaluation 2	side(1,ux)	-0.66797	Ліва межа ux
Point Evaluation 3	side(2,ux)	0.33203	Права межа ux
Point Evaluation 4	mean(d(u,x))	-0.16797	(Ліва границя+ Права границя)/2

Point Evaluation 5	d(u,x)	0.0	Значення d(u,x)
Point Evaluation 6	uxx	1.4211E-14	Середнє значення uxx
Point Evaluation 7	down(uxx)	-0.062500	Ліва границя uxx
Point Evaluation 8	up(uxx)	0.062500	Права границя uxx

Щоб зрозуміти, які номери слід використовувати в операторах side, потрібно активувати вузол **Component 1 (comp1) - Definitions – View 1**, і в його ПВ встановити прапорці **Show geometry labels** та **Show edge direction arrows**, і потім, якщо треба, клацнути над графічним вікном по кнопці Select Domains.



Індикатори відрізків 1 і 2, які з'являються (зображені на рисунку), можна використовувати в операторах side(сторона, вираз) при позначенні сторони обчислення виразу.

Зауваження. Дотичні похідні uTx, dtang(u,x) в точці x=0 дорівнюють нулю (перевірте самостійно).

Надрукуйте значення перших п'яти створених вузлів Point Evaluation в таблицю **Table 1**.

Gradient of u, x component,	side(1,ux), Point: 2	side(2,ux), Point: 2	mean(d(u,x)), Point: 2	d(u,x), Point: 2
-0.16797	-0.66797	0.33203	-0.16797	0.0000

Як бачите, в точці x=0 змінна ux дорівнює -0.16797=(-0.66797+0.33203)/2, тобто середньому граничних значень з двох боків. Самі граничні значення зліва і справа дорівнюють -0.66797 та 0.33203.

У вузлі **Point Evaluation 4** ми продемонстрували використання оператора mean(expr), який обчислює середнє значення виразу expr в точці, розглядаючи її як вершину графічних примітивів, що з нею межують. Його значення співпадає зі значенням ux в цій вершині, бо воно також обчислюється як середнє.

Останні 3 рядки таблиці обчислюють величину uxx в точці x=0, а також її ліве і праве граничне значення down(uxx), up(uxx). Надрукуйте ці значення в таблицю **Table 2**.

Gradient of ux, x component, Point: 2	down(uxx), Point: 2	up(uxx), Point: 2
1.4211E-14	-0.062500	0.062500

Таблиця підтверджує, що uxx=(down(uxx)+up(uxx))/2.

П'ятий рядок нашої таблиці містить значення оператора d(u,x) в розглядуваній вершині. Воно дорівнює 0 в будь якій кінцевій точці (включно з вершиною x=0). Всередині відрізків (виключаючи точки x=-1,0,1) похідні ux і d(u,x) співпадають. Це зрозуміло з їх графіків. Для перевірки можна обчислити значення цих похідних у будь якій внутрішній точці, наприклад, в

точці x=0.5. Створіть два вузли типу Global Evaluation, виконавши наступні інструкції.

Results -> **Derived Values** – Global Evaluation

Derived Values - Global Evaluation 1:

Expression – **Expression** = at1(0.5,ux)

Оскільки точка x=0.5 не є вершиною геометрії, то для обчислення виразів в ній ми застосували оператор at1, а не at0.

LMB > Evaluate – New Table

Global Evaluation 1 : RMB > Duplicate

Derived Values - Global Evaluation 2:

Expression – **Expression** = at1(0.5,d(u,x))

LMB > Evaluate – Table 3

at1(0.5,ux)	at1(0.5,d(u,x))
0.45703	0.45703

У внутрішній точці (яка не є вершиною геометрії) значення похідних ux і d(u,x) співпадають. ■

Як випливає з попередніх прикладів, коли змінна визначена в області і на її межі, то на межі вона (змінна) успадковує значення від своїх граничних значень в області. Наприклад, в останньому прикладі в точці x=1 (на межі) значення ux співпадає зі значенням down(ux) (перевірте це самостійно), в той час, як up(ux)=0. Якщо межа внутрішня, то перші і другі похідні по просторовим координатам обчислюються як середні значення по суміжним областям.

7.6. Змінні в моделях менших розмірностей (Lower Dimensions).

В COMSOL Multiphysics в просторових 3D (або 2D) моделях можна працювати з геометріями меншої розмірності. Наприклад, можна виконувати розрахунок тонкої оболонки, розглядаючи її як поверхню в тривимірному просторі. Або можна розв'язувати задачу в криволінійному стрижні, інтерпретуючи його як просторову лінію. Моделі меншої розмірності (Lower Dimensions) можна створювати в прикладних інтерфейсах, а також в математичному PDE інтерфейсі. Звичайно, в таких моделях є деякі особливості використання змінних. В першу чергу це стосується градієнтів шуканих величин, які можна обчислювати лише в дотичних до поверхні або кривої напрямках. Отже тут дотичні похідні uTx,uTy,uTz (u – шукана величина) є обов'язковим атрибутом обробки результатів досліджень. В цьому параграфі на двох прикладах (для поверхні і кривої в 3D) буде пояснено спосіб використання дотичних похідних в задачах меншої розмірності.

В наступній моделі буде продемонстровано використання змінних дотичних похідних при розв'язанні ДРЧП в криволінійній оболонці без моделювання її товщини. В ній обчислюватиметься густина струму $\boldsymbol{J}$, що тече по оболонці, і електричний потенціал V, який у випадку відсутності джерел задовольняє рівнянню $\nabla\cdot(-\sigma\nabla V)=0$, де σ - питома електропровідність середовища (См/м). Тут густина струму $\boldsymbol{J}=-\sigma\nabla V$ представляє кількість

заряду, що проходить через одиницю площі поперечного перерізу провідника за одиницю часу. При розв'язанні задачі на поверхні вектор $\boldsymbol{J}$ бажано відносити до одиниці довжини кривої, яку пересікає струм. Для цього величину $-\sigma\nabla V$ слід помножити на товщину оболонки. Отже розв'язувати потрібно рівняння

$$\nabla\cdot\left(-\sigma d\ \nabla V\right)=0. \qquad (1)$$

При сталому d форма рівняння не змінюється, але густина струму, віднесена до одиниці довжини «лінійного» поперечного перерізу, буде обчислюватися за формулою $\boldsymbol{J}=-\sigma\cdot d\cdot\nabla V$.

Для спрощення прикладу ми оберемо сферичну оболонку. Більш реалістичний приклад можна знайти в бібліотеці додатків (Application Library), який відкривається послідовністю команд

File -> Application Libraries -> COMSOL Multiphysics –
Equation Based – shell diffusion

Приклад 1. Електричний струм в сферичній оболонці.

Нехай оболонка вироблена із сталевого листа завтовшки 1 мм і електропровідністю 7690000 См/м. Біля південного і північного полюсів сфери є кругові отвори, різниця потенціалів між якими становить 100 В.

Розв'язання. Створіть модель, призначену для розв'язання ДРЧП менших розмірностей (Lower Dimensions). Це означає, що геометрія моделі матиме розмірність меншу за розмірність простору, наприклад, буде криволінійною поверхнею в 3D, а не тілом.

File -> New -> Model Wizard -> 3D -> Mathematics - PDE Interfaces - Lower Dimensions - Coefficient Form Boundary PDE (cb) -> Stationary

Задайте параметри.

Global Definitions : RMB > Parameters

Parameters – **Name, Expression** = sigma, 7690000 [S/m]
– **Name, Expression** = d, 1[mm] (*товщина оболонки*)

Геометрія моделі складатиметься зі сфери, яка має два кругових отвори навколо полюсів.

Geometry -> **Primitives** – Sphere

Geometry 1 - Sphere 1 (sph1):

Object Type – **Type** = Surface (*тип – поверхня, а не тіло*)
Size – **Radius** = 1

Geometry -> **Primitives** – Cylinder

Size and Shape – **Radius, Height** = 0.25,2
Position – **x:,y:,z:**= 0,0,-1
Axis – **Axis type**= z-axis

Geometry -> **Operations - Booleans and Partitions** -> Difference

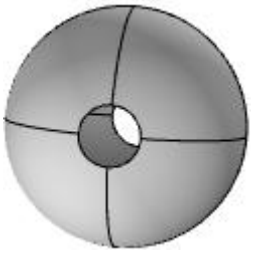
Geometry 1 - Difference 1 (dif1):

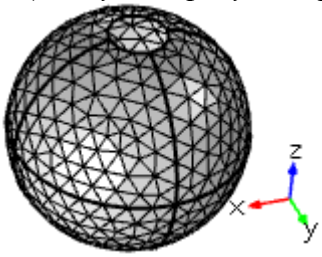
Difference – **Object to add** = sph1 (*сфера*)
Union – **Object to subtract** = cyl1 (*циліндр*)

LMB > Build All Objects (*наступний рисунок ліворуч*)

Побудуйте СкЕ *сітку* з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All (*наступний рисунок праворуч*)





Задайте рівняння. Спочатку визначте зону дії рівняння, обравши всі ділянки поверхні.

Coefficient Form Boundary PDE:

Boundary Selection – **Selection** = All boundaries (*вся поверхня*)

Далі визначіть одиниці вимірювання шуканої величини і зовнішніх джерел струму (хоча джерел немає, останнє визначає одиниці вимірювання лівої і правої частин ДРЧП). В ПВ цього ж вузла **Coefficient Form Boundary PDE** введіть:

Units – **Dependent variable quantity** = Electric potential (V)

– **Source term quantity – Unit:** = A*m^-2

Задайте коефіцієнти рівняння (1).

Coefficient Form PDE 1:

Diffusion Coefficient section – **c** <> = sigma*d

Source Term – **f** <> = 0

Damping or Mass Coefficient – $\mathbf{d_a}$ <> = 0

Задайте граничні умови. Вважатимемо, що один отвір заземлений, а на іншому заданий потенціал 100 В.

Physics -> **Edge - Edges** -> Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection**=Manual,5,6,12,15 (*південний отвір*)

Dirichlet Boundary Condition – **r** = 0

Physics -> **Edge - Edges** -> Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection**=Manual,7,8,13,17 (*північний отвір*)

Dirichlet Boundary Condition – **r** = 100

Виконайте обчислення.

Home -> Compute

В гілці **Results** з'явиться графічна група **3D Plot Group 1** з вузлом **Surface 1**, що містить кольоровий графік розв'язку, тобто потенціалу. Додайте на рисунок стрілковий графік вектора густина струму $\boldsymbol{J} = -\sigma \cdot d \cdot \mathrm{grad} V$ (в якості потенціалу V в моделі виступає змінна u).

3D Plot Group 1 -> **Add Plot** -> Arrow Surface

3D Plot Group 1 – Arrow Surface 1:

Expression – **x component** = -sigma*d*uTx

– **y component** = -sigma*d*uTy

– **z component** = -sigma*d*uTz

Оскільки шукана величина *u* відома лише на поверхні, то похідні `ux,uy,uz` невизначені. Для обчислення векторного поля густини струму використовуються змінні дотичних похідних шляхом додавання суфікса `T` до імені змінної. Отже тут ми використали змінні `uTx,uTy,uTz`, які разом утворюють дотичний вектор до поверхні. Він є вектором проекції градієнта змінної функції форми `u` на дотичну площину $(\nabla u)_T = \nabla u - (\boldsymbol{n}\cdot\nabla u)\boldsymbol{n}$, де $(\nabla u)_T$ позначає «дотичний градієнт», складений з «дотичних похідних вздовж просторових координат», $\boldsymbol{n}$ – одиничний вектор зовнішньої нормалі.

`Coloring and Style` – **Arrow length** = `Normalized`

LMB > Plot (*наступний рисунок ліворуч*)

Зауваження. В нашій моделі замість змінних `uTx`, `uTy`, `uTz` в полях **`Expression`** вузла **Arrow Surface 1** можна використати вирази `dtang(u,x)`, `dtang(u,y)`, `dtang(u,z)`.

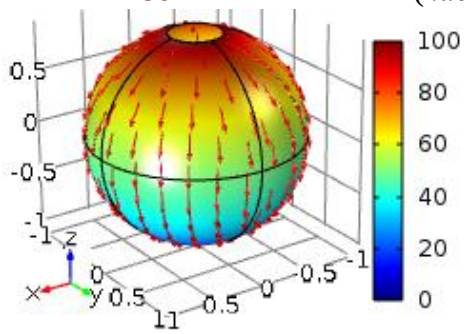
На поверхні оболонки можна також відтворити графік величини густини струму $|\boldsymbol{J}| = |-\sigma\cdot d\cdot\nabla u_T| = \sigma\cdot d\cdot\sqrt{uTx^2+uTy^2+uTz^2}$ (A/m).

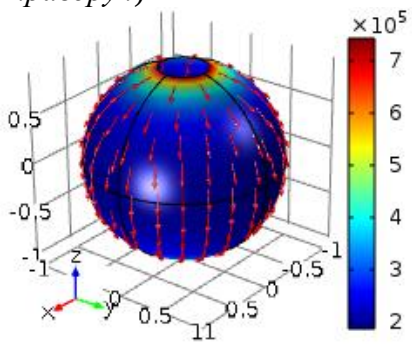
3D Plot Group 1: RMB > Duplicate

3D Plot Group 2 – Surface 1:

`Expression` – **Expression** = `sigma*d*sqrt(uTx^2+uTy^2+uTz^2)`

LMB > Plot (*наступний рисунок праворуч*)





Зауваження. Для побудови стрілкового графіка вздовж кривої на поверхні, потрібно створити джерело даних типу `Edge 3D`, і потім на ньому побудувати `Arrow Line` графік з виразами типу `uTx`, `uTy`, `uTz`. В нашому прикладі будувати такий графік вздовж контурів отвору біля полюсів немає сенсу, бо значення `u` вздовж них не змінюється (на них задано фіксовану граничну умову).

Проаналізуємо роботу дотичних похідних з величиною, яка не є змінною функції форми. Для цього спочатку створіть таку змінну.

`Definitions` **`->Variables`** – Local Variables

Definitions – Variables 1:

`Variables` – **Name, Expression** = `U, x^2-y^2`

Змінна `U` визначена для всієї моделі, тобто на сфері, і не пов'язана з фізичним змістом задачі.

Поновіть обчислення.

Study 1: RMB > Update Solution

Побудуйте кольоровий графік змінної `U` на поверхні сфери, а також стрілковий графік векторного поля її «дотичного градієнта».

3D Plot Group 1: RMB > Duplicate

3D Plot Group 3 - Surface 1:

`Expression` – **Expression** = `U`

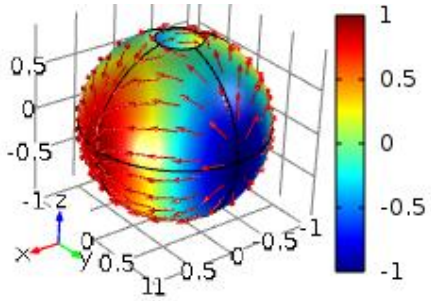
3D Plot Group 3 – Arrow Surface 1:

`Expression` – **x component** = `dtang(U,x)`

– **y component** = `dtang(U,y)`

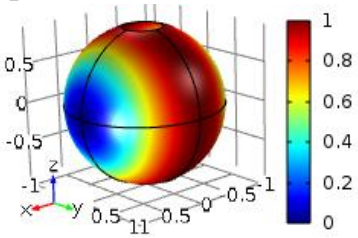
– **z component** = `dtang(U,z)`

LMB > Plot

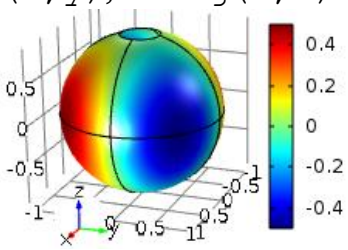


Спроба замінити похідні `dtang(U,x)`, `dtang(U,y)`, `dtang(U,z)` на вирази `UTx`, `UTy`, `UTz` або на `Ux`, `Uy`, `Uz` призводять до повідомлення про помилку.

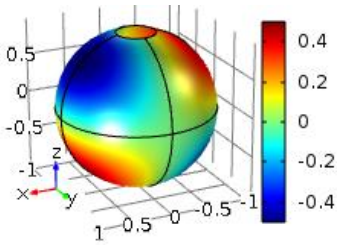
Для тренування побудуйте самостійно поверхневі графіки наступних виразів: `dtang(x,x)`, `dtang(x,y)`, `dtang(x,z)`.



`dtang(x,x)`



`dtang(x,y)`



`dtang(x,z)`

Отже вони і не одиничні і не нульові.

Зауваження. На побудованій оболонці можна використовувати поверхневі змінні `s1` і `s2`, але їх дотичні похідні, наприклад, `s1Tx` не існують, а `dtang(s1,x)` – дорівнює нулю. Як і для будь-якої поверхні, для оболонки визначені дотичні вектори $\boldsymbol{t}_1, \boldsymbol{t}_2$ і вектор нормалі $\boldsymbol{n}$, координати яких позначаються через `t1x,t1y,t1z,t2x,t2y,t2z,nx,ny,nz`. Також визначені геометричні змінні `dnx`, `dny`, `dnz`, `unx`, `uny`, `unz`, `curv1`, `curv2`, `tcurv1x`, `tcurv1y`, `tcurv1z`, `tcurv2x`, `tcurv2y`, `tcurv2z`. Але їх дотичні похідні не визначені або тотожно дорівнюють нулю.

Зауваження. При побудові геометрії у вигляді сфери можна було використати її інструмент **`Layers`**, побудувавши внутрішній сферичний шар. Але при встановленій опції **`Object Type`** – **Type**=`Surface` ми все одно отримаємо поверхню тіла, а не суцільне тривимірне тіло.

Приклад 2. Розв'язати задачу розподілу температури в однорідному ізотропному кільці одиничного радіуса, утвореному зігнутим стрижнем круглого поперечного перерізу діаметром d. Теплообмін на поверхні стрижня відсутній і задано початковий розподіл температури $T(\alpha,0)=f(\alpha)$, де α - полярний кут (див. приклад в п. 3.4).

Рівняння теплопровідності у випадку відсутності зовнішніх джерел тепла має вигляд $\rho C\dfrac{\partial T}{\partial t}+\nabla\cdot(-k\nabla T)=0$, де T – температура, k – коефіцієнт теплопровідності (Вт/м·град), ρ і C – густина і питома теплоємність матеріалу, а величина $\boldsymbol{q}=-k\nabla T$ представляє густину теплового потоку. При розв'язанні задачі в одновимірному стрижні вектор $\boldsymbol{q}$ бажано відносити до поперечного перерізу стрижня. Для цього величину $-k\nabla T$ слід домножити на площу поперечного перерізу S і розв'язувати рівняння

$$\rho C S\frac{\partial T}{\partial t}+\nabla\cdot(-k S\nabla T)=0. \qquad (2)$$

Якщо площа S стала, то форма рівняння не змінюється, але тепловий потік вздовж стрижня обчислюватиметься за формулою $\boldsymbol{q}=-k S\ \nabla T$.

Щоб порівняти розв'язок поточної задачі з розв'язком, отриманим в прикладі п. 3.4, оберіть наступні параметри (гіпотетичні) матеріалу кільця:

$$\rho=1000\ кг/м^3,\ C=4.8\ дж/кг\cdot град,\ d=16.287\ мм,\ k=4800\ Вт/м\cdot град.$$

Тоді $k\cdot S\approx 1.0000$, $\rho\cdot C\cdot S\approx 1.0000$ (опущено одиниці вимірювання) і ми матимемо задачу таку, як в п. 3.4.

Розв'язання. Створіть тривимірну `Lower Dimension` модель. Це означає, що геометрія моделі матиме розмірність меншу за розмірність простору. Якщо задача розв'язуватиметься вздовж лінії (одновимірного кільця), то слід обирати Coefficient Form Edge PDE (ce) інтерфейс (можна також застосувати інтерфейси General Form Edge PDE або Weak FormEdge PDE).

File -> New -> Model Wizard -> 3D -> Mathematics – PDE Interfaces - Lower Dimensions - Coefficient Form Edge PDE (ce) -> Time Dependent

Задайте параметри.

Global Definitions : RMB > Parameters

Parameters – **Name, Expression** = `k, 4800 [W/(m*K)]`
– **Name, Expression** = `ro, 1000[kg/m^3]`
– **Name, Expression** = `C, 4.8[J/(kg*K)]`
– **Name, Expression** = `d, 16.287[mm]` (*діаметр кільця*)
– **Name, Expression** = `S, pi*d^2/4`

Створіть функцію початкової температури.

Global Definitions: RMB > Functions – Analytic

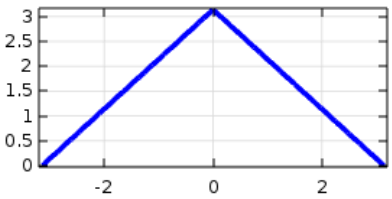
Function name = `f0`

Definition – **Expression** = `pi-abs(x)`

Для перевірки виконайте побудову графіка цієї функції.

Plot Parameters – **Low limit, Upper limit** = `-pi, pi`

LMB> Plot



Геометрія моделі складатиметься з лінії у формі кільця. При активному вузлі **Geometry 1** виконайте команду

`Geometry -> Work Plane -> Work Plane`

В ПВ створеного вузла **Work Plane 1** всі параметри залиште без змін. Вони відповідають тому, що робоча площина буде збігатися з площиною XY. Клацніть по кнопці Show Work Plane у верхній частині панелі властивостей.

Активуйте вузол **Work Plane 1 - Plane Geometry** і в створеній площині побудуйте окружність.

Work Plane 1 - Plane Geometry:

`Work Plane` **-> Primitives** – Circle

Work Plane 1 - Plane Geometry - Circle 1 (c1):

Object Type – **Type** = `Curve` (*тип – крива, а не тіло*)

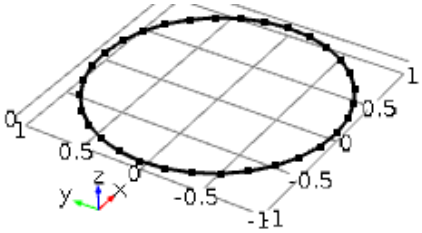
Size and Shape – **Radius** = `1`

LMB > Build All Objects

Побудуйте СкЕ *сітку* з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Щоб побачити вузли СкЕ сітки, можливо вам доведеться натиснути кнопку Select Edges над графічним вікном.



Задайте рівняння. Спочатку визначіть зону дії рівняння.

Coefficient Form Boundary PDE(ce):

Boundary Selection – **Selection** = `All edges`

Далі оберіть одиниці вимірювання шуканої величини і зовнішніх джерел тепла (хоча джерел немає, останнє визначає одиниці вимірювання лівої і правої частин ДРЧП).

Units – **Dependent variable quantity** = `Temperature (K)`

– **Source term quantity – Unit:** = `W/m`

Зверніть увагу на те, що шукана величина температури позначається в нашій моделі літерою `u` (**Dependent Variables - Field name** = `u`).

Задайте коефіцієнти рівняння (2).

Coefficient Form PDE 1:

Diffusion Coefficient section – **c <>** = `k*S`

Source Term – **f <>** = `0`

Damping or Mass Coefficient – $\mathbf{d_a}$ **<>** = `ro*C*S`

Задайте початкову температуру. Перевірте, що в ПВ вузла **Initial Values 1** в секції вибору **Selection** обрано індикатори всіх дуг окружності. Потім введіть **Initial Values 1:**

Initial Values – **Initial Value for u:** – **u** = `f0(atan2(y,x))[K]`

Тут величина `atan2(y,x)` обчислює полярний кут α точок/перерізів кільця.

Зауваження. Граничні умови в поточній задачі задавати непотрібно.

Налаштуйте розв'язувач.

Study 1 - Step 1: Time Dependent:

Study Settings – **Times (s)** =

`range(0,0.01,0.25) range(0.275,0.025,1.5)`

Виконайте обчислення.

`Home` -> Compute

В гілці **Results** з'явиться графічна група **3D Plot Group 1** з вузлом **Line 1**, що містить кольоровий графік температури. Налаштуйте стиль графіка і додайте до нього вертикальну деформацію.

3D Plot Group 1:

Data – **Time (s)** = `0`

3D Plot Group 1 - Line 1:

Coloring and Style – **Line type** = `Tube`

– **Tube radius expression** = `0.03`

– **Radius scale factor** = `True,1`

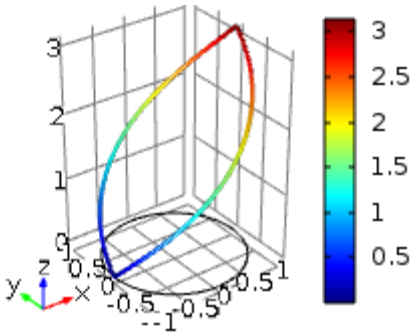
3D Plot Group 1 - Line 1: RMB > Deformation

3D Plot Group 1 - Line 1 - Deformation 1:

Expression – **x component, y component, z component** = `0,0,u`

Scale – **Scale factor** = `True,1`

LMB > Plot



На рисунку зображено графік температури в початковий момент часу, але змінивши значення в полі **1D Plot Group 1:** **Data** – **Time** можна відобразити графік в будь який момент.

Побудуйте кольоровий графік величини теплового потоку $q = k\,S\,|\nabla u|$.

`Results` -> **Plot Group** -> 3D Plot Group

3D Plot Group 2:

Data – **Time (s)** = `0.2`

`3D Plot Group 2` -> **Add Plot** -> Line

3D Plot Group 2– Line 1:

Expression – **Expression** = `k*S*sqrt(uTx^2+uTy^2+uTz^2)`

Coloring and Style – **Line type** = `Tube`

– **Tube radius expression** = `0.03`

– **Radius scale factor** = `True,1`

RMB > Deformation

3D Plot Group 2 - Line 1 - Deformation 1:

Expression – **x component, y component** = `0,0`

Expression –**z component** = `k*S*sqrt(uTx^2+uTy^2+uTz^2)`

Scale– **Scale factor** = `True,1`

Тут для обчислення величини дотичного градієнта $|\nabla u_T|$ застосовано вираз `sqrt(uTx^2+uTy^2+uTz^2)`, який використовує змінні дотичних похідних (похідні `d(u,x)`, `d(u,y)`, `d(u,z)` непридатні).

Додайте на рисунок стрілковий графік вектора густини теплового потоку $\boldsymbol{q} = -k\,S\,\nabla u_T$.

`3D Plot Group 2->` Add Plot – Arrow Line

3D Plot Group 2 - Arrow Line 1 :

Expression – **x component** = `-k*S*uTx`

– **y component** = `-k*S*uTy`

– **z component** = `-k*S*uTz`

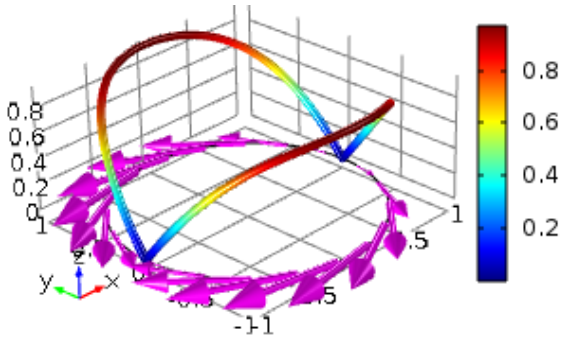
Coloring and Style – **Scale factor** = `True,1`

– **Placement** =`Uniform`

– **Number of arrows** =`20`

– **Color** =`Magenta`

LMB>Plot



Тут при побудові стрілкового графіка були знову використані дотичні похідні `uTx`,`uTy`,`uTz`.

Побудуйте лінійний графік температури в залежності від подовжньої координати кільця.

`Results` **-> Plot Group ->** 1D Plot Group

1D Plot Group 3:

Data –**Time selection** = `Manual`

Data –**Times (1-76)** = `range(1,10,76)`

`1D Plot Group 3->` **Add Plot ->** Line Graph

1D Plot Group 3 – Line Graph 1:

Selection – **Selection, <>** = `Manual,1,2,3,4`

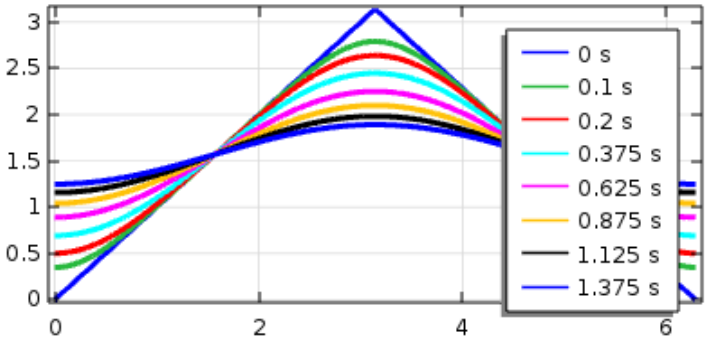
y-Axis Data – **Expression** = u

x-Axis Data – **Parameter** = Arc length

Coloring and Style –**Width** = 3

Legends – **Show legends** = True

LMB > Plot



Порівняйте отриманий графік з аналогічним графіком п. 3.4.

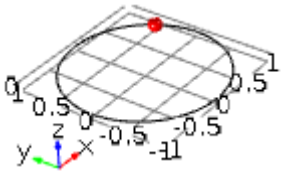
В цьому прикладі можна побудувати графіки температури або величини теплового потоку в фіксованій точці в залежності від часу. Для цього спочатку створіть точкове джерело даних.

Results -> Data Set - Cut Point 3D

Data Sets -Cut Point 3D 1:

Point Data – **x:, y:,z:** = cos(pi/6), sin(pi/6),0

LMB > Plot



Потім побудуйте графіки.

Results **-> Plot Group ->** 1D Plot Group

1D Plot Group 4-> **Add Plot ->** Point Graph

1D Plot Group 4 – Point Graph 1:

Data – **Data set** = Cut Point 3D 1

y-Axis Data – **Expression** = u

x-Axis Data – **Parameter** = Time

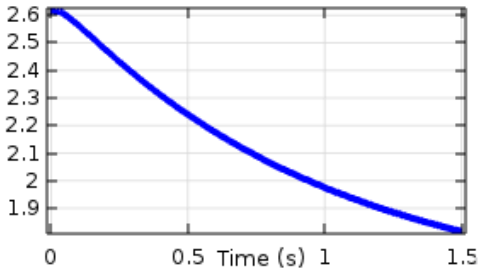
LMB > Plot (*наступний рисунок ліворуч*)

1D Plot Group 4: RMB > Duplicate

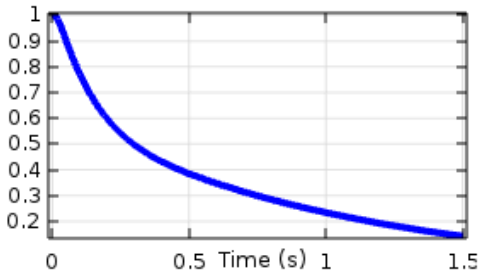
1D Plot Group 5 – Point Graph 1:

y-Axis Data – **Expression** = k*S*sqrt(uTx^2+uTy^2+uTz^2)

LMB > Plot (*наступний рисунок праворучч*)



Температура в точці.



Густина теплового потоку в точці.

Зауваження. Дотичні похідні можна диференціювати по часу, вказуючи позначки похідних uTxt, uTyt,uTzt у зазначеному порядку (наприклад, для похідної uTxt спочатку x, потім t).

В поточному прикладі шукана величина u була змінною функцій форми. Тепер проаналізуємо поведінку похідних величини U=x^2-y^2, яка не є змінною функцій форми. Спочатку створіть таку змінну в моделі.

Definitions **->Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = U, x^2-y^2

Тут змінна U визначена у всій моделі, тобто на окружності, і не пов'язана з фізичним змістом задачі.

Оновіть обчислення.

Study 1: RMB > Update Solution

Зрозуміло, що похідні Ux, Uy, Uz змінної U не визначені, а похідні d(U,x), d(U,y), d(U,z) будуть класичними похідними U по просторовим координатам. Крім того ясно, що дотичні похідні UTx, UTy, UTz не існують (такі похідні створюються лише для змінних, обчислюваних на СкЕ сітці), але похідні dtang(U,x), dtang(U,y), dtang(U,z) можна використовувати. Для тренування читача побудуємо на кільці графіки похідних d(U,x) і dtang(U,x), а також векторних полів (0,0,d(U,x)) і (0,0,dtang(U,x)).

Results **->Plot Group ->** 3D Plot Group

3D Plot Group 6:

3D Plot Group 6 ->**Add Plot ->** Line

3D Plot Group 6 – Line 1:

Expression – **Expression** = d(U,x) (d(x^2-y^2,x)=2*x)

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.03

– **Radius scale factor** = True,1

RMB > Deformation

3D Plot Group 6 - Line 1 - Deformation 1:

Expression – **x component, y component,** = 0,0

Expression –**z component** = d(U,x)

Scale– **Scale factor** = True,0.5

3D Plot Group 6-> Add Plot – Arrow Line

3D Plot Group 6 - Arrow Line 1 :

Expression – **x component, y component** = 0,0

– **z component** = d(U,x)

Coloring and Style – **Scale factor** = True,0.5

– **Placement** =Uniform

– **Number of arrows** =30

– **Color** =Magenta

LMB > Plot (*наступний рисунок ліворуч*)

3D Plot Group 6: RMB > Duplicate

3D Plot Group 7 – Line 1:

Expression – **Expression** = `dtang(U,x)`

3D Plot Group 7 - Line 1 - Deformation 1:

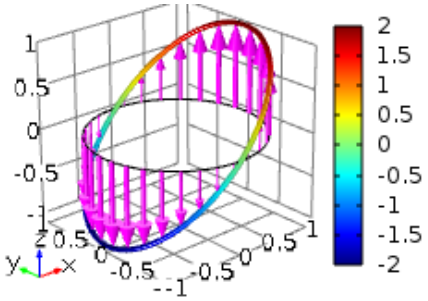
Expression – **z component** = `dtang(U,x)`

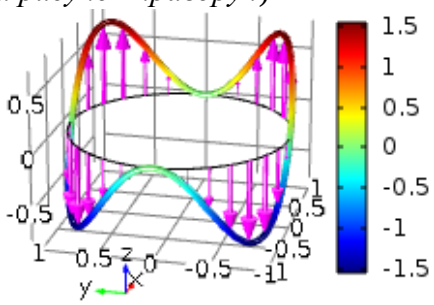
3D Plot Group 7 - Arrow Line 1 :

Expression – **z component** = `dtang(U,x)`

LMB>Plot

(*наступний рисунок праворуч*)





Ще можна подивитися на графіки похідних `dtang(x,x)`, `dtang(y,x)` вздовж кільця.

`Results` -> **Plot Group** -> 1D Plot Group

1D Plot Group 8:

Data – **Time selection** = `First`

`1D Plot Group 8`-> **Add Plot** -> Line Graph

1D Plot Group 8 – Line Graph 1:

Selection – **Selection**, <> = `Manual,1,2,3,4`

y-Axis Data – **Expression** = `dtang(x,x)`

x-Axis Data – **Parameter** = `Arc length`

Coloring and Style – **Width** = `3`

Legends – **Show legends** , **Legends** = `True,Manual`

– **Legends (таблиця)** = `dtang(x,x)`

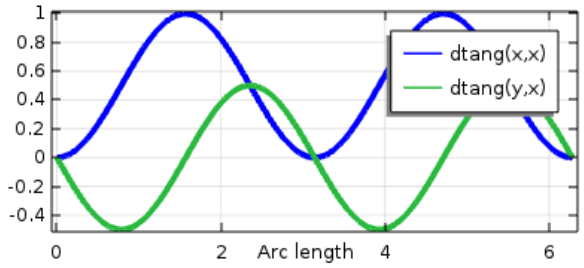
1D Plot Group 8 – Line Graph 1: RMB > Duplicate

1D Plot Group 8 – Line Graph 2:

y-Axis Data – **Expression** = `dtang(y,x)`

Legends – **Legends (таблиця)** = `dtang(y,x)`

LMB > Plot



В тривимірних моделях на одновимірних примітивах існує змінна `s1` і змінні `t1x,t1y,t1z` – координати дотичного вектора. Перевіримо це для нашої `Lower Dimensions` моделі.

3D Plot Group 6: RMB > Duplicate

3D Plot Group 9 – Line 1:

Expression – **Expression** = `s1`

3D Plot Group 9 - Line 1 - Deformation 1:

Expression – z component = s1

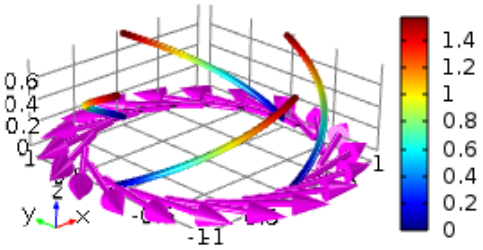
3D Plot Group 9 - Arrow Line 1 :

Expression – x component, y component, z component =t1x,t1y,t1z

Coloring and Style – Scale factor = True,1

– Number of arrows =20

LMB>Plot



7.7. Системи координат користувача і їх змінні.

В моделях COMSOL Multiphysics в залежності від просторової розмірності використовуються декартові (в 1D, 2D, 3D моделях) або циліндричні (в Axisymmetric моделях) координати. Але можна застосовувати і інші.

В тривимірному просторі, в якому запроваджена система декартових координат x, y, z, між ними і криволінійними координатами u, v, w встановлюється взаємно однозначна відповідність за формулами

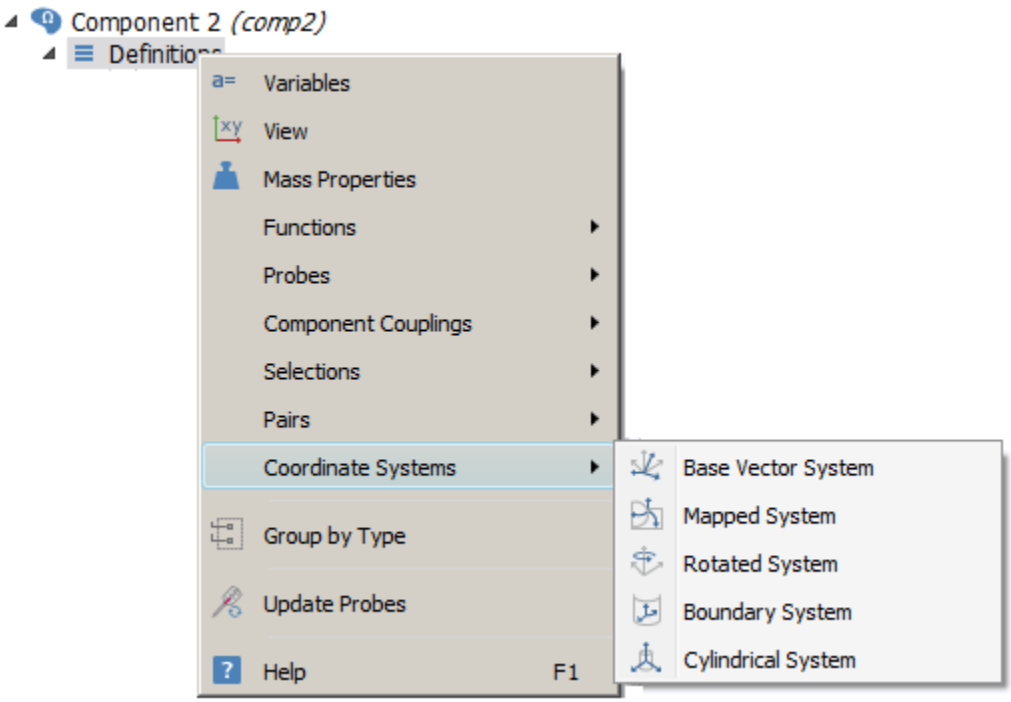
$$x = x(u,v,w),\ y = y(u,v,w),\ z = z(u,v,w) \qquad (1)$$

або

$$u = u(x,y,z),\ v = v(x,y,z),\ w = w(x,y,z). \qquad (2)$$

Рівняння $u = const$, $v = const$, $w = const$ є рівняннями координатних поверхонь криволінійної системи в прямокутній системі координат. Кожна пара координатних поверхонь, що проходить через фіксовану точку, утворює в перетині координатну лінію. Їх параметричні рівняння отримуються з рівнянь (1), якщо в них по черзі змінювати лише одну змінну u, v, w, залишаючи незмінними дві інші. Схожим чином вводяться криволінійні координати на площині. Криволінійні системи координат є одним із засобів вказати положення точок в просторі. Іншим їх призначенням є спрощення визначення векторних полів.

В COMSOL Multiphysics інструмент «координатні системи користувача» призначений для більш зручного задавання різноманітних умов, зокрема при оперуванні формулами. Маючи в розрахунковій області одночасно кілька системи координат, ви можете застосовувати позначення їх змінних в будь-яких виразах, спрощуючи постановку задач. Отже «система координат» в COMSOL моделі, це вузол, в якому визначаються додаткові змінні (або базисні вектори), застосовувані для позначення положення точок простору і задання векторних полів. Побачити типи систем координат можна з випадного меню гілки **Definitions**, в якій створюються вузли цих систем. На наступному рисунку наведено ці типи для двовимірного випадку.



Так, у вузлі `Mapped System` задаються довільні вирази (2), що зв'язують імена нових координат зі старими. В результаті в геометрії моделі одночасно визначаються декартові і криволінійні координати. Їх імена можна використовувати при заданні виразів в будь-яких місцях моделі: в рівняннях, граничних і початкових умовах, а також при побудові графіків.

Після створення в гілці **Definitions** вузла системи координат, її можна наглядно зобразити, побудувавши координатні лінії або координатні поверхні. Можна також зобразити поле її одиничних базисних векторів.

Приклад 1. Створення і зображення криволінійної системи координат.

Розв'язання. Створіть порожню модель.

File->New -> Blank Model

Додайте в модель двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть в моделі перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Геометрія моделі складатиметься з кола одиничного радіуса.

`Geometry` -> **`Primitives`** – Circle

Geometry 1 - Circle 1 (c1):

`Size and Shape` – **Radius** = `1`

LMB > Build All Objects

Побудуйте СкЕ сітку з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Створіть вузол криволінійної системи координат (u, v), пов'язаної з декартовими координатами формулами $u = x + y^2/2$, $v = y - 0.3 \cdot x^3$.

`Definitions` -> **`Coordinate Systems`** – Mapped System

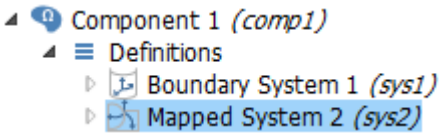
В гілці **Definitions** з'являється вузол **Mapped System 2 (sys2)** системи координат, позначення і формули перетворення якої задаються в ПВ.

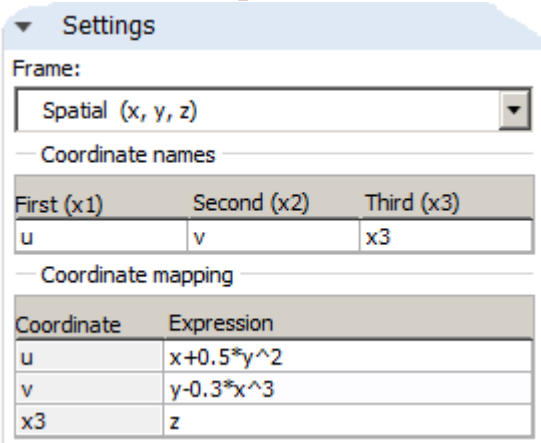
Definitions – Mapped System 2 (sys2):

Settings – **Coordinate names – First, Second** = `u,v` (*введення в таблиці*)

– **Coordinate mapping – Coordinate, Expression** = `u,x+0.5*y^2`

– **Coordinate mapping – Coordinate, Expression** = `v,y-0.3*x^3`





Створені ідентифікатори змінних (додаючи префікс `sys2`) можна використовувати в моделі одночасно з декартовими координатами. Наприклад, змінні (u,v) можна застосувати при побудові координатних ліній.

Щоб змінні системи координат були працездатними, модель потрібно відкомпілювати. Це буде зроблено після створення і обчислення гілки досліджень `Study`.

Untitled.mph (root): RMB > Add Study = `Add Study ->`

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Побудуйте координатні лінії $u = const$, $v = const$.

`Results` -> **`Plot Group`** -> 2D Plot Group

2D Plot Group 1:

`Color Legend` – **Show legends =** `False`

`2D Plot Group 1`-> **`Add Plot`** -> Contour

2D Plot Group 1 – Contour 1:

`Expression` – **Expression** = `sys2.u`

`Coloring and Style` – **Contour type** = `Line`

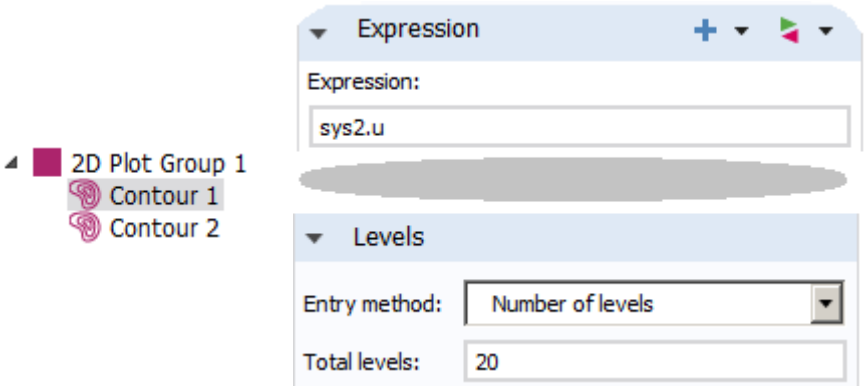
– **Coloring, Color** = `Uniform, Black`

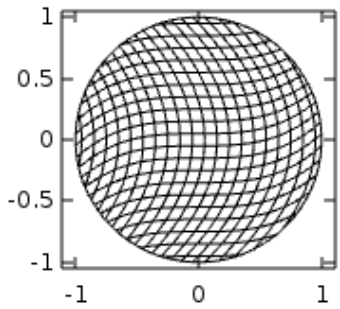
2D Plot Group 1 – Contour 1: RMB > Duplicate

2D Plot Group 1 – Contour 2:

`Expression` – **Expression** = `sys2.v`

LMB>Plot (*наступний рисунок праворуч*)





Побудуйте векторне поле одиничних векторів цієї системи координат.

`Results` **-> `Plot Group`** -> 2D Plot Group

2D Plot Group 2:

`Color Legend` – **Show legends =** `False`

`2D Plot Group 2`-> **`Add Plot`** -> More Plots - Coordinate System Surface

2D Plot Group 2 – Coordinate System Surface 1:

`Coordinate System` – **Source** = `Coordinate System`

– **Coordinate System** = `Mapped System 2(sys2)`

`Positioning` – **x grid point, Entry method** = `Numder of points`

– **x grid point, Points** = `10`

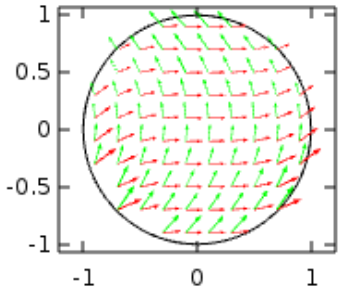
– **y grid point, Entry method** = `Numder of points`

– **y grid point, Points** = `10`

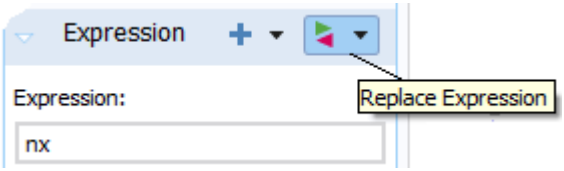
`Coloring and Style` – **Arrow length** = `Proportional`

– **Scale factor** = `True,0.2`

LMB>Plot



Зауваження. В будь якому компоненті після створення розрахункової області окрім основної системи координат (декартової або циліндричної) автоматично створюється допоміжна «гранична» система координат (Boundary System) - локальна система базових векторів дотичних і нормальних до межі тіла. На межі 2D фігур ці вектори позначаються $(\boldsymbol{t}_1, \boldsymbol{n})$, а на межі 3D тіл - $(\boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{n})$. Позначення проекцій цих векторів на декартові осі можна побачити в вікнах `Replace Expression`, які розкриваються при натисканні на однойменну кнопку, що розташована зверху праворуч кожного текстового поля, призначеного для введення виразів.



Для прикладу побудуйте над контуром нашого кола криву, висота якої визначатиметься значенням першої компоненти вектора нормалі.

Results -> **Plot Group** -> 2D Plot Group

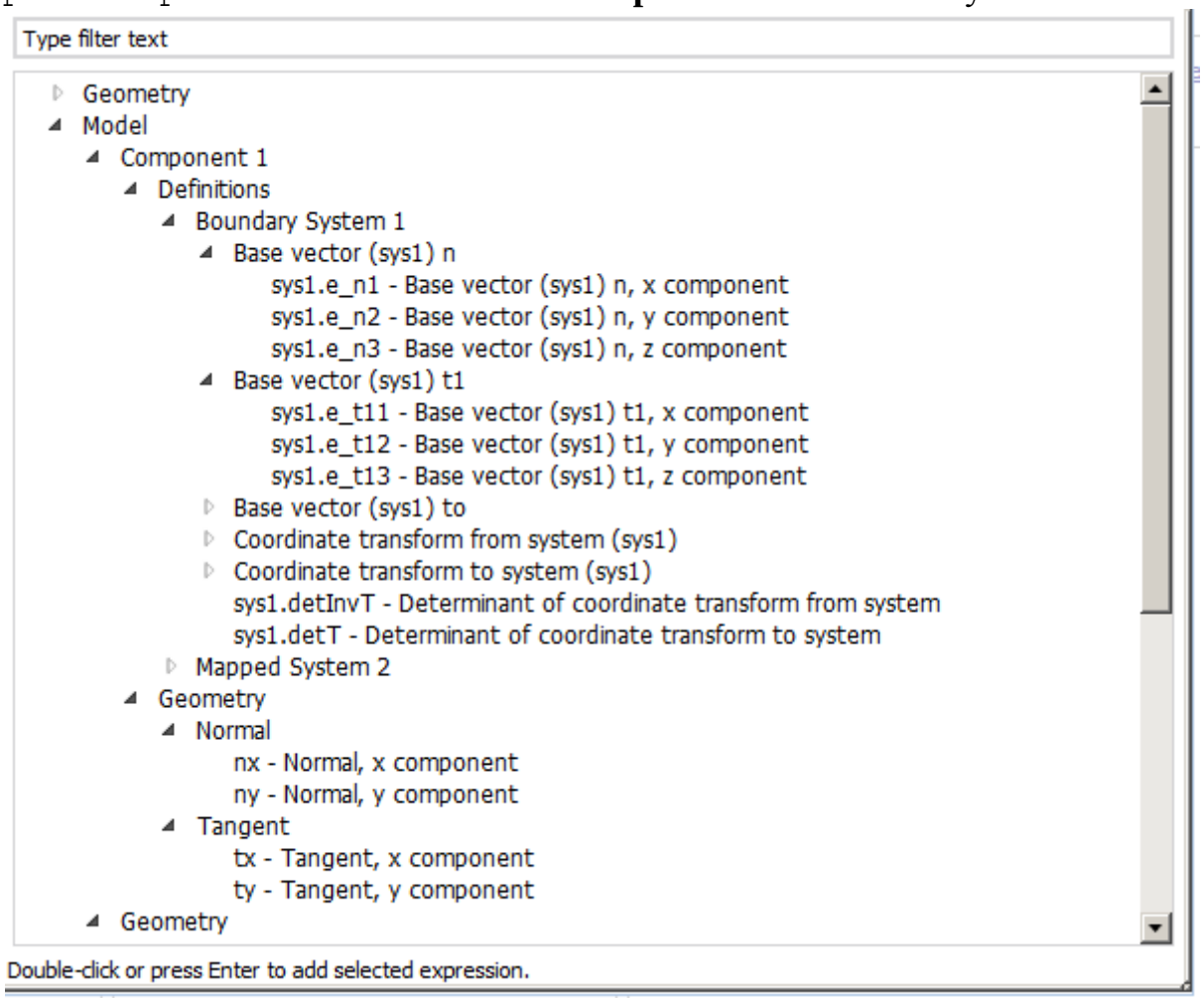
2D Plot Group 3:

Color Legend – **Show legends =** False

2D Plot Group 3-> **Add Plot** -> Line

2D Plot Group 1 – Line 1:

Щоб побачити, як позначаються компоненти вектора нормалі, розкрийте вікно Replace Expression текстового поля **Expression** поточного вузла **Line 1**.



Клацнувши двічі по рядку sys1.e_n1- Base vector(sys1) n,x component, ми вставимо потрібний ідентифікатор в поле виразів.

Expression – **Expression** = sys1.e_n1

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.03

– **Radius scale factor** = True,1

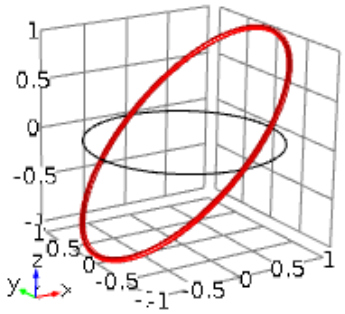
– **Coloring, Color** = Uniform, Red

RMB> Height Expression

2D Plot Group 3 – Line 1 – Height Expression 1:

Axis – **Scale factor** = True,1

LMB>Plot

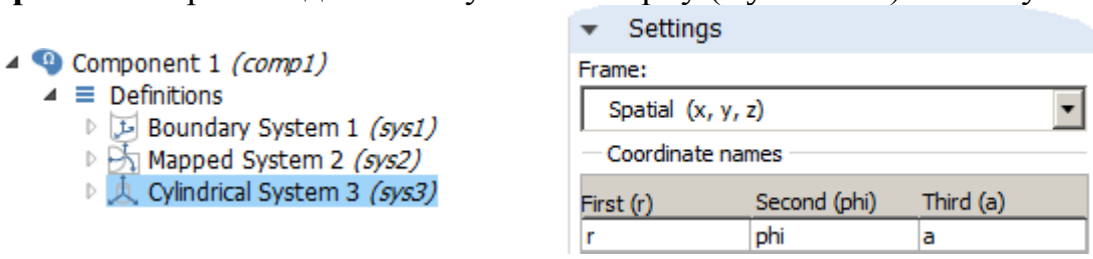


Позначення `sys1.e_n1` (та інші) компонентів вектора нормалі і `sys1.e_t11` (та інші) дотичного вектора можна використовувати в моделі, наприклад при заданні граничних умов. Таким чином, автоматичне створення вузла `Boundary System` призводить до того, що у двовимірному компоненті на межі його геометрії завжди визначені змінні `sys1.e_n1` і т.д.

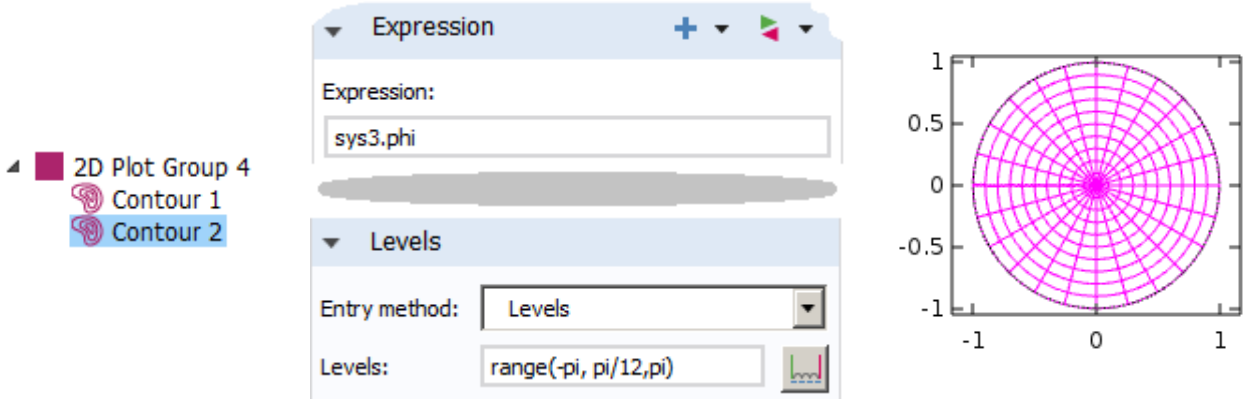
Якщо користувачеві потрібні інші координати в розрахунковій області або на її межі, то він повинен створити їх самостійно.

Зауваження. З наведеного вище вікна `Replace Expression` видно, що в гілці `Component – Geometry` визначені спрощені позначення (`tx`,`ty`), (`nx`,`ny`) компонентів дотичного і нормального векторів $(\boldsymbol{t},\boldsymbol{n})$. Звичайно, що існують аналогічні позначення для координат векторів $(\boldsymbol{t}_1,\boldsymbol{t}_2,\boldsymbol{n})$ в тривимірних моделях.

Вправа. Створіть в одиничному колі полярну (Cylindrical) систему координат.



Побудуйте її лінії рівня та векторне поле одиничних векторів, пам’ятаючи що в COMSOL полярний кут `phi` змінюється від $-\pi$ до $+\pi$.



■

Координатні системи корисні при застосуванні формул, які в науковій літературі зазвичай записуються з використанням спеціальних координат (із зручності). Наприклад, пошук гармонічної функції в області (такі функції задовольняють рівнянню Лапласа) по її значенням $\varphi(\tau)$ на межі називається задачею Діріхле. Аналітичний розв’язок задачі Діріхле в колі дається формулою Пуассона [1]

$$u(z)=\frac{1}{2\pi}\int_0^{2\pi}\varphi(\tau)\frac{1-r^2}{1-2r\cos(\tau-\varphi)+r^2}d\tau,\ \left(z=re^{i\varphi}\right). \qquad (3)$$

Для побудови графіка функції $u(z)=u\left(re^{i\varphi}\right)=u(r,\varphi)$, щоб не переписувати формулу (3) в декартових координатах, зручно використати циліндричні координати (r,φ).

Приклад 2. Реалізувати формулу (3) і побудувати графік функції `u` при граничному значенні $\varphi(\tau)=(\pi-|\tau|)/\pi$.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Додайте в модель двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Побудуйте в геометрії одиничне коло.

`Geometry` -> **`Primitives`** – Circle

Geometry 1 - Circle 1 (c1):

`Size and Shape` – **Radius** = `1`

`Layers` – **Layer name, Thickness** = `0.01`

Оскільки формула (3) непридатна для обчислення функції на межі контура, то для близьких до контура точок можна отримати суттєву похибку. При побудові графіків буде кращим виключити з області тонку зовнішню зону кола. Для створення таких зон у вузлах деяких графічних примітивів є розділ **`Layers`** з таблицею, в якій вказується товщина зон/шарів та їх назва. Зовнішній шар розташовується в таких таблицях першим.

Layers

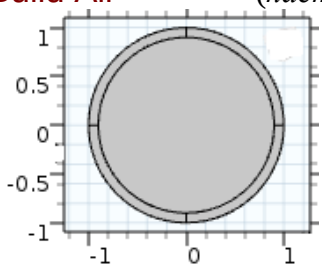
Layer name	Thickness
Layer 1	0.01
Layer 2	

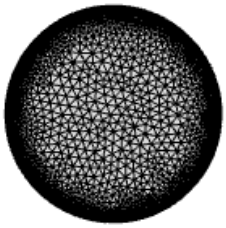
LMB > Build All Objects (*наступний рисунок ліворуч, товщина шару збільшена*)

Побудуйте СкЕ сітку.

Mesh 1: `Mesh Settings` –**Element size** = `Fine`

LMB > Build All (*наступний рисунок праворуч*)





Зверніть увагу, що в зовнішньому кільці і зоні біля нього побудована дуже дрібна сітка.

Створіть оператор інтегрування по контуру кола.

`Definitions` > Component Couplings - Integration

Definitions - Integration 1 (intop1):

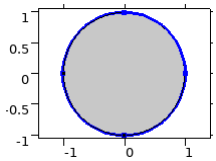
Operator name = `intop1`

`Source Selection` – **Geometric entity level** = `Boundary`

Остання інструкція вказує, що інтегрування буде виконуватися по лінії.

– **Selection** = `Manual, 5,6,9,12`

(*дуги зовнішнього одиничного кола*)



Будьте уважні, не виділіть внутрішні дуги створеного шару.

Створіть функцію граничної умови.

Global Definitions: RMB > Functions – Analytic

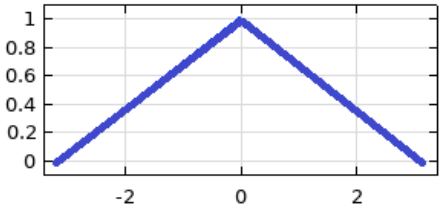
Function name = `f`

`Definition` – **Expression** = `(pi-abs(x))/pi`

Для перевірки виконайте побудову графіка цієї функції.

`Plot Parameters` – **Low limit, Upper limit** = `-pi, pi`

LMB> Plot



Створіть вузол циліндричної системи координат.

`Definitions` -> **`Coordinate Systems`** – Cylindrical System

В гілці **Definitions** створюється вузол системи координат **Cylindrical System 2 (sys2)**), позначення і розташування якої наводяться в його ПВ. Цей вузол створює змінні `r` і `phi`, які використовуються для позначення полярних координат точок площини.

Coordinate names

First (r)	Second (phi)	Third (a)
r	phi	a

Origin

x	y
0	0

Створіть таблицю змінних моделі, в якій одночасно з декартовими координатами можна використовувати змінні `r` і `phi` створеної системи (додаючи префікс `sys2`). Наприклад, `sys2.r` позначатиме полярний радіус. Для спрощення введення формули (3), окрім змінної `u`, ми створимо кілька допоміжних змінних.

Definitions: RMB > Variables

Definitions – Variables 1:

`Variables` – **Name, Expression** = `Zu, 1-dest(sys2.r)^2`

`Variables` – **Name, Expression** = `Zd, 1-2*dest(sys2.r)*cos(sys2.phi-dest(sys2.phi))+dest(sys2.r)^2`

`Variables` – **Name, Expression** = `Z, Zu/Zd`

`Variables` – **Name,Expression** = `u,intop1(f(sys2.phi)*Z)/(2*pi)`

Variables

Name	Expression
Zu	1-dest(sys2.r)^2
Zd	1-2*dest(sys2.r)*cos(sys2.phi-dest(sys2.phi))+dest(sys2.r)^2
Z	Zu/Zd
u	intop1(f(sys2.phi)*Z)/(2*pi)

Змінні `r` і `phi`, коли вони є незалежними зовнішніми змінними, включено в оператор `dest(...)`, а коли по ним виконується інтегрування, застосовуються без такого оператора. Зокрема тут інтегрування виконується лише по змінній `sys2.phi`, яка в (3) позначена ідентифікатором τ.

Оператори і змінні потрібно відкомпілювати. Це виконується при обчисленні вузла дослідження.

Untitled.mph (root): RMB > Add Study = `Add Study ->`
Preset Studies – Stationary -> X

Study 1: LMB > Compute

Щоб будувати графік `u` над колом з виключеним зовнішнім шаром, створіть звужене ДД.

Results: Data Sets - Study 1/Solution 1 : RMB > Duplicate

Data Sets - Study 1/Solution 1 (2) : RMB > Selection

Study 1/Solution 1 (2) - Selection:

Geometric Entity Selection – **Geometric entity level** = `Domain`

Geometric Entity Selection – **Selection** = `5` (*центральне коло*)

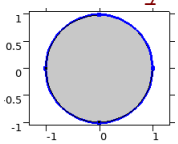
Після створення копії джерела даних ми додали підвузол **Selection**, який для неї (копії) дозволяє обрати рівень геометрії. В нашому випадку ми обрали центральне коло (одиничне коло з виключеним тонким зовнішнім шаром).

Крім того, ми збираємося побудувати графік граничної функції $\varphi(\tau)$. Для цього нам знадобиться звуження ДД на контур одиничного кола. Повторіть команду дублювання ДД, оберіть в створеній копії граничний рівень геометрії, а потім оберіть безпосередньо дуги кола.

Data Sets - Study 1/Solution 1 (2) : RMB > Duplicate

Data Sets - Study 1/Solution 1 (3) - Selection:

Geometric Entity Selection – **Geometric entity level** = `Boundary`
– **Selection** = `5,6,9,12`



Тепер побудуйте графік розв'язку `u`.

`Results` -> **Plot Group** ->2D Plot Group

2D Plot Group 1:

Data – **Data set** = `Study 1/Solution 1(2)`

`2D Plot Group 1`-> **Add Plot** -> Surface

2D Plot Group 1 – Surface 1:

Expression – **Expression** = `u`

RMB> Height Expression

2D Plot Group 1 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = `True,1`

Для наочності побудуйте над контуром кола графік граничної функції $\varphi(\tau)$.

`2D Plot Group 1->` **Add Plot ->** Line

2D Plot Group 1 – Line 1:

Data – **Data set** = `Study 1/Solution 1 (3)`

Expression – **Expression** = `f(sys2.phi)`

Coloring and Style – **Line type** = `Tube`

– **Tube radius expression** = `0.03`

– **Radius scale factor** = `True,1`

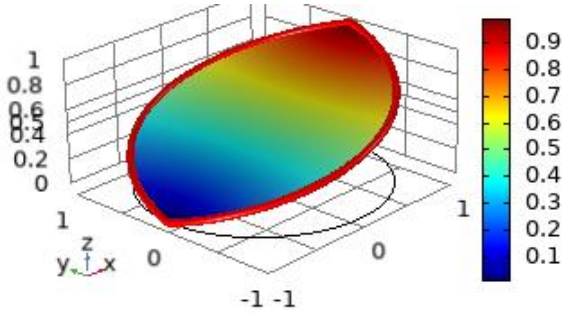
– **Coloring, Color** = `Uniform, Red`

RMB> Height Expression

2D Plot Group 1 – Line 1 – Height Expression 1:

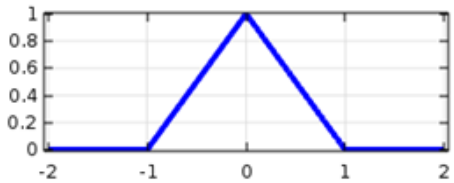
Axis – **Scale factor** = `True,1`

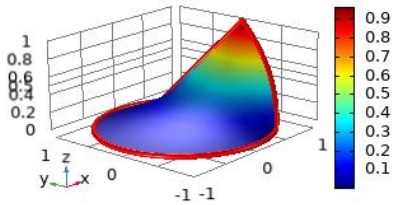
LMB>Plot



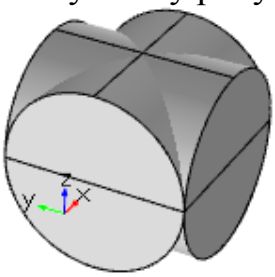
Як бачите, при створенні змінної `u` і при побудові графіка граничної функції ми використовували ідентифікатори `sys2.r`, `sys2.phi` полярної системи координат, що значно зручніше застосування декартових координати `x`,`y`.

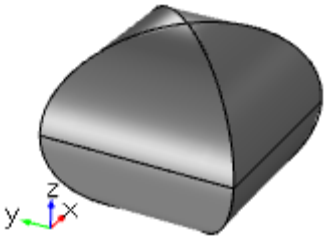
Вправа. За допомогою формули (3) побудувати в одиничному колі розв'язок задачі Діріхле з граничним значенням $\varphi(\tau)=(|\tau+1|-2|\tau|+|\tau-1|)/2$. Розв'язати задачу засобами COMSOL (методом СкЕ) і побудувати графік різниці двох розв'язків.



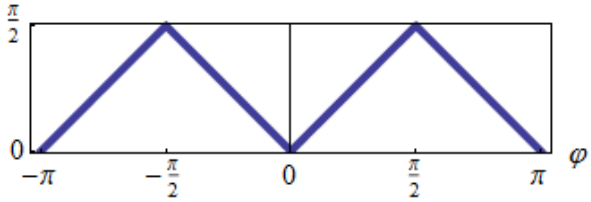


Приклад 3. Розв'язати стаціонарного рівняння теплопровідності $\Delta u=-1$ у тілі, отриманому як перетин двох перпендикулярних циліндрів. Циліндри зображені на наступному рисунку ліворуч, а тіло – праворуч.





Вважатимемо, що на поверхні тіла температура залежить від азимутальних кутів φ циліндричних систем координат, пов'язаних з відповідними циліндрами. Наприклад, нехай температура відносно кутів φ змінюється лінійно від 0 на «екваторі» до $\pi/2$ на «полюсах» тіла. Враховуючи, що азимутальний кут φ в COMSOL Cylindrical System змінюється від $-\pi$ до $+\pi$, функція залежності температури від кута φ може мати наступний графік.



Її рівняння матиме вигляд $f(\varphi)=\pi-|\varphi+\pi/2|+|\varphi|-|\varphi-\pi/2|$. Задавати таку граничну температуру, використовуючи декартові координати буде клопітно. Але якщо створити дві циліндричні системи координат, поздовжні осі яких збігаються з осями утворювальних циліндрів, то це спростить постановку задачі.

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Додайте в модель тривимірний компонент.

Untitled.mph (root) : RMB > Add Component ->3D

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Побудуйте в геометрії два циліндра з ортогональними поздовжніми осями.

Geometry -> **Primitives** -> Cylinder

Cylinder 1 (cyl1):

Size and Shape – **Radius, Height** = 1, 2

Position – **x:, y:, z:** = -1,0,0

Axis – **Axis type, x:, y:, z:** = Cartesian,1,0,0 (*вектор поздовжньої осі*)

Geometry -> **Primitives** -> Cylinder

Cylinder 2 (cyl2):

Size and Shape – **Radius, Height** = 1, 2

Position – **x:, y:, z:** = 0,-1,0

Axis – **Axis type, x:, y:, z:** = Cartesian,0,1,0 (*вектор поздовжньої осі*)

LMB > Build Selected (*початковий рисунок прикладу ліворуч*)

Створіть тіло, як перетин двох циліндрів.

Geometry -> Operations -> Booleans and Partitions – Intersection

Geometry 1 - Intersection 1 (int1):

Intersection – **Input objects** = cyl1, cyl2 (*обидва циліндра*)

LMB > Build All Objects (*початковий рисунок прикладу праворуч*)

Задайте дві циліндричні системи координат.

Definitions -> **Coordinate Systems** – Cylindrical System

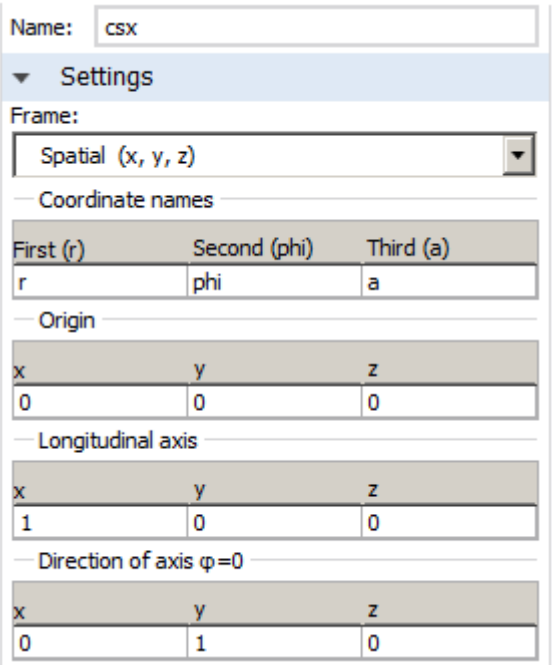
В гілці **Definitions** створюється вузол системи координат **Cylindrical System 2**. Налаштуйте його.

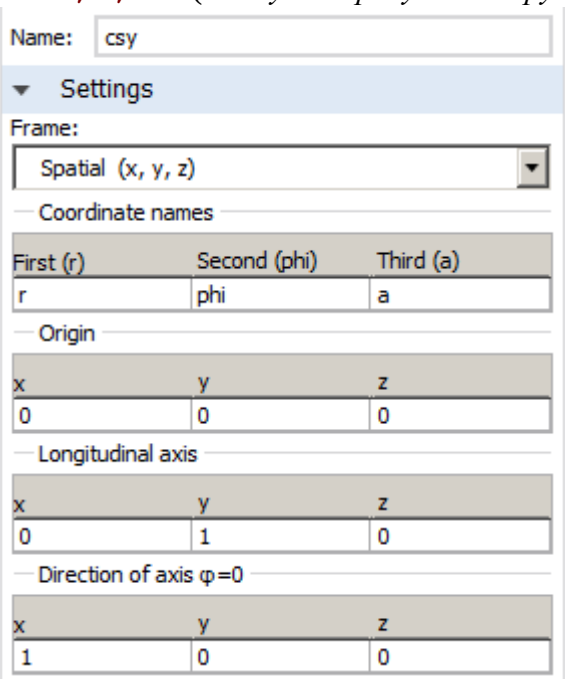
Cylindrical System 2:

Name = csx

Settings – **Longitudinal axis, x,y,z** = 1,0,0 (*вектор поздовжньої осі*)

– **Direction of axis φ=0** = 0,1,0 (*наступний рисунок ліворуч*)





Definitions -> **Coordinate Systems** – Cylindrical System

Cylindrical System 3:

Name = csy

Settings – **Longitudinal axis, x,y,z** = 0,1,0 (*вектор поздовжньої осі*)

– **Direction of axis φ=0** =1,0,0 (*попередній рисунок праворуч*)

Створіть допоміжну функцію для зручного введення граничної умови.

Global Definitions: RMB > Functions – Analytic

Function name = f

Definition – **Expression** = pi-abs(x+pi/2)+abs(x)-abs(x-pi/2)

Для перевірки побудуйте графік цієї функції.

Plot Parameters – **Low limit, Upper limit** = -pi, pi

LMB> Plot (*другий рисунок на початку прикладу*)

Щоб змінні координатних систем і функція стали доступними, їх треба відкомпілювати. Це виконується при обчисленні вузла дослідження моделі.

Untitled.mph (root): RMB > Add Study = Add Study ->

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Тепер можна зобразити координатні поверхні системи координат **Cylindrical System 2 (csx)**.

Results -> **Plot Group** ->3D Plot Group

3D Plot Group 1:

Color Legend – **Show legends =** False

3D Plot Group 1->Add plot-Isosurface

3D Plot Group 1 – Isosurface 1:

y-Axis Data – Expression = `csx.r`

Levels – Entry method, Levels = `Levels, range(0.2,0.2,1)`

3D Plot Group 1 – Isosurface 1: RMB > Duplicate

3D Plot Group 1 – Isosurface 2:

y-Axis Data – Expression = `csx.phi`

Levels – Entry method, Levels = `Levels, range(-pi,pi/4,pi)`

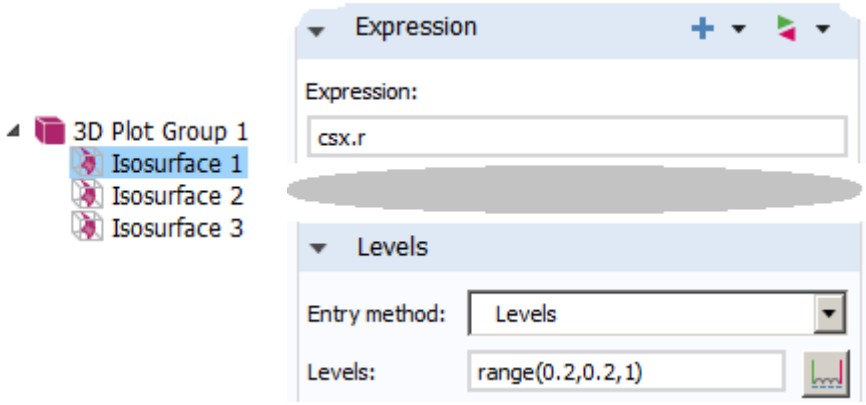
3D Plot Group 1 – Isosurface 1: RMB > Duplicate

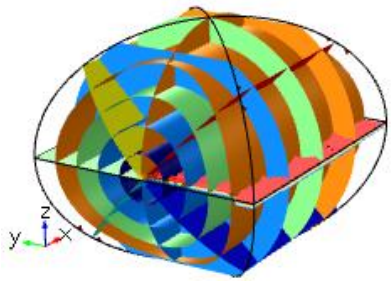
3D Plot Group 1 – Isosurface 3:

y-Axis Data – Expression = `csx.a`

Levels – Entry method, Levels = `Levels, range(-1,0.5,1)`

LMB> Plot (*наступний рисунок праворуч*)





Отже координатними поверхнями є концентричні циліндри з поздовжньою віссю X, площини, що проходять крізь цю вісь, і площини, які їй перпендикулярні.

Щоб розв'язувати рівняння теплопровідності, в модель потрібно додати відповідну «фізику».

Component 1: RMB > Add Physics = `Add Physics ->`

Mathematics – Classical PDEs – Heat Equation (hteq) ->X

Зона дії рівняння вказується в ПВ вузла **Heat Equation (hteq)** в секції вибору **Domain Selection**. Залиште там одиницю – значення за замовчуванням, яке відповідає всьому тілу. В розділі **Dependent Variable** залиште також `u` - ім'я шуканої величини.

Налаштуйте рівняння.

Heat Equation (hteq) - Heat Equation 1 :

Diffusion Coefficient – c = 1

Source Term – f = 1

Absorption Coefficient – d_a = 0

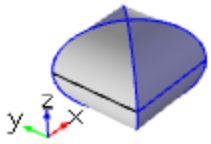
Задайте граничні умови на боках першого утворювального циліндра.

`Physics` -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – Selection , <>=`Manual,3,4,5,6`

Dirichlet Boundary Condition – r =`f(csx.phi)`



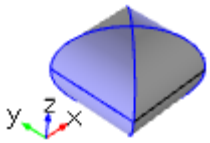
Задайте граничні умови на боках другого утворювального циліндра.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** , <>=Manual,1,2,7,8

Dirichlet Boundary Condition – **r** = f(csy.phi)



В обох випадках ми використали функцію $f(\varphi)$, але азимутальний кут φ в першому випадку відноситься до системи координат **Cylindrical System 2 (csx)**, а в другому – до **Cylindrical System 3 (csy)**.

Знову виконайте обчислення моделі.

Study 1: LMB > Compute

В цьому разі крім компіляції змінних буде знайдено розв'язок u рівняння теплопровідності. Побудуйте його поверхневий графік.

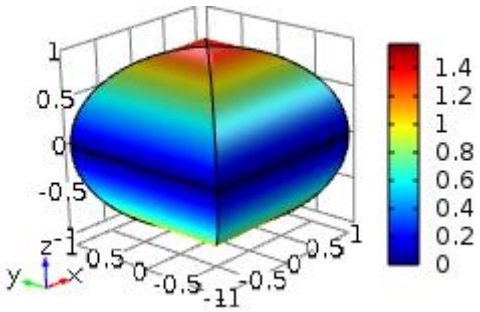
Results -> Plot Group - 3D Plot Group

RMB > Surface

3D Plot Group 2 - Surface 1 :

Expression – **Expression** = u

LMB > Plot



Побудуйте графік розв'язку по двом «меридіальним» і одному «екваторіальному» перерізу. Для цього створіть джерела даних, які відповідають бажаним перерізам, а потім на них побудуйте поверхневі графіки величини u.

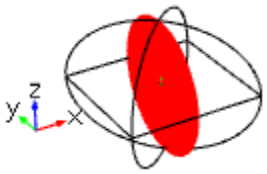
Results -> Data Set -> Cut Plane

Data Sets – Cut Plane 1:

Plane Data – **Plane** = yz-planes

Переріз по площині YZ

LMB > Plot



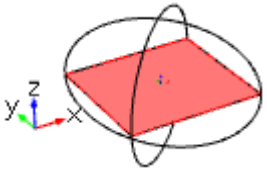
Results -> Data Set -> Cut Plane

Data Sets – Cut Plane 2:

Plane Data – **Plane** = xy-planes

Переріз по площині XY

LMB > Plot

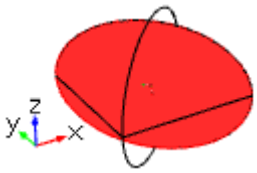


Results -> **Data Set** - Cut Plane

Results – Data Sets – Cut Plane 3:

Plane Data – Plane type = General

Plane Data – Plane entry method = Point and normal



Побудова площини перерізу по точці і нормалі.

Plane Data – Point – x, y, z = 0,0,0

Plane Data – Normal – x, y, z = 1,1,0

LMB > Plot

Тепер накресліть графіки розв'язку u на цих перерізах. Щоб їх кольорові палітри співпадали, діапазони на них треба задати однаковими.

Results -> Plot Group - 3D Plot Group

3D Plot Group 3 : RMB > Surface

3D Plot Group 3 - Surface 1 :

Data – Data set = Cut Plane 1

Expression – Expression = u

Range – Manual color range = True

– Minimum, Maximum = 0 , pi/2 (*два поля*)

3D Plot Group 3 - Surface 1 : RMB > Duplicate

3D Plot Group 3 - Surface 2 :

Data – Data set = Cut Plane 2

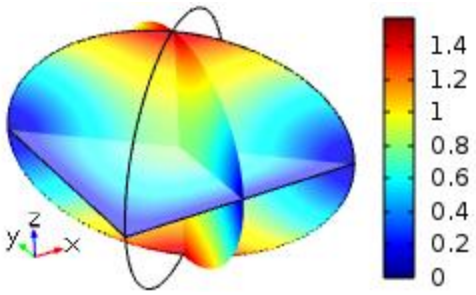
Coloring and Style – Color legend = False

3D Plot Group 3 - Surface 2 : RMB > Duplicate

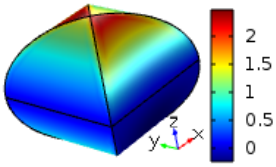
3D Plot Group 3 - Surface 3 :

Data – Data set = Cut Plane 3

LMB > Plot



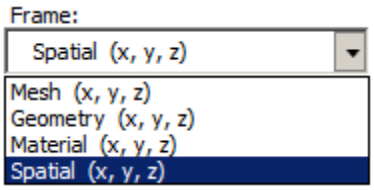
Зауваження 1. У виразах можуть одночасно застосовуватися змінні всіх систем координат, створених в компоненті. Наприклад, в розглянутій задачі вираз в граничних умовах на поверхні одного із циліндрів може мати вид f(csx.phi)*csy.r+z, в який входять змінні декартової і обох циліндричних систем координат.



Зауваження 2. Розглянуту задачу було б краще розв'язувати, використовуючи її симетрію. Але для несиметричних граничних умов, наприклад таких, які наведені в попередньому зауваженні, це не підходить.

Зауваження 3. В COMSOL Multiphysics координати можуть бути «прив'язані» не до простору, а до середовища, геометрії або СкЕ сітки. Наприклад координати, що пов'язані з середовищем, називають матеріальними Такі

координати виділяють індивідуальні точки довкілля і в процесі його руху не змінюються.. Спосіб прив'язки координат в термінології COMSOL називається фреймом, який можна обирати з випадного списку **Frame** ПВ створюваної системи координат, або з вузла, в якому використовуються такі системи. З вибором фрейма можуть змінюватися позначення координат. Наприклад, матеріальні декартові координати позначаються великими літерами `X,Y,Z`, а координати, що пов'язані з входовою геометрією - `xg,yg,zg`. Зазвичай, якщо середовище, геометрія і СкЕ сітка не рухаються, координатні системи співпадають.



Література.

1. Лаврентьев М.А., Шабат Б.В. Методы теории функций комплексного переменного. – М.: Наука, Главная редакция физико-математической литературы, 1973, 736 стр.

8. Комплексні величини.

В цій главі ми покажемо, які можливості надає COMSOL Multiphysics для роботи з комплексними величинами.

Уявна одиниця $\left(\sqrt{-1}\right)$ позначається літерою `i` або `j`. Комплексні сталі записуються, наприклад, наступним чином `3+2*i`. Якщо `x` і `y` позначають декартові координати на площині, то `z=x+i*y` можна розглядати як комплексну змінну на цій площині. Для роботи з комплексними змінними можна використовувати позначки арифметичних операцій та майже всі вбудовані функції, бо вони «вміють» працювати з комплексними аргументами. До того ж є функції, які за своєю суттю призначені для роботи з комплексними числами (КЧ): `real(z)`, `imag(z)` – дійсна і уявна частини КЧ, `abs(z)`, `arg(z)` – модуль і аргумент КЧ, `conj(z)` – спряження КЧ. Є ще оператор `realdot(z1,z2)`, який трактує комплексні аргументи як двовимірні вектори і обчислює їх скалярний добуток (це не функція, бо диференціювання по `z1` і `z2` працює не так, як очікується).

8.1. Використання комплексних змінних.

Приклад. Робота з елементарними функціями комплексного змінного.

Розв'язання. Створіть порожню модель.

File->New -> Blank Model

Додайте в модель двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Геометрія моделі складатиметься з масиву 3 x 3 квадратів розміром 2 x 2. Побудуємо їх. Спочатку створіть один квадрат.

Geometry->**Primitives->** Square

Geometry 1 - Square 1 (sq1):

Size – **Size length:** = 2

Position – **x:, y:** = -3, -3 (*два поля*)

LMB > Build Selected

Тепер створіть масив квадратів.

Geometry -> **Transforms ->** Array

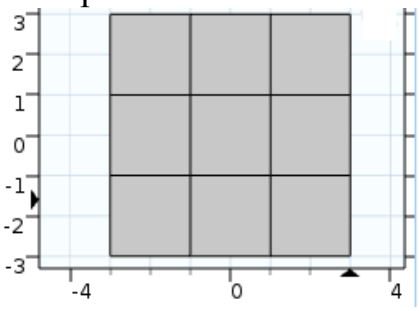
Geometry 1 - Array 1 (arr1):

Input – **Input objects** = sq1

Size – **x size, y size:** = 3, 3 (*два поля*)

Displacement – **x, y:** = 2, 2 (*два поля*)

Закінчіть конструювання геометрії натисканням на кнопку Build All Objects .



Активуйте вузол створеного компонента **Component 1** і виконайте команду:

Component 1:

Definitions -> **Variables** – Local Variables

В ПВ створеного вузла **Variables 1** заповніть таблицю змінних так, як зображено на наступному рисунку.

Variables

Name	Expression
Z	x+i*y

Тут ми створили комплексну змінну Z=x+i*y, яка діє на всій геометрії.

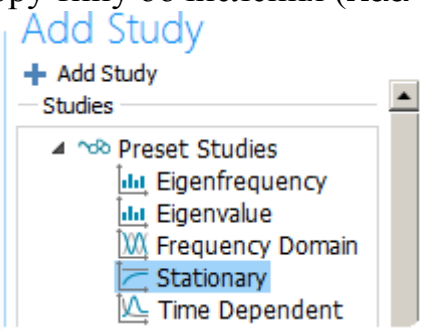
Створіть СкЕ сітку з параметрами за замовчуванням.

Component 1 – Mesh 1 : LMB > Build All

Щоб змінні моделі працювали, потрібно виконати обчислення, а для цього треба мати вузол дослідження.

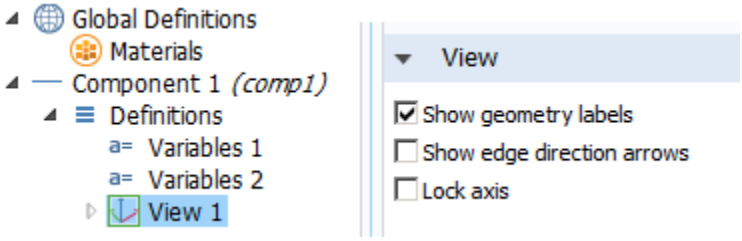
Untitled.mph (root): RMB > Add Study

Система відкриє вікно вибору типу обчислення (`Add Study`).



Виконайте подвійне клацання миші по рядку `Stationary` і закрийте вікно хрестиком в правому верхньому куті. В дереві моделі з'явиться вузол **Study 1**.

Щоб при наступних побудовах відображалися індикатори геометричних примітивів встановіть в ПВ вузла **Component 1 – Definitions – View 1** опцію Show geometry labels = `True`.



Виконайте обчислення.

`Home` -> Compute

Для вибору різних областей, над якими будуватимуться графіки функцій, корисно створити кілька додаткових джерел даних.

Results:

Data Sets - Study 1/Solution 1 : RMB > Duplicate

Data Sets - Study 1/Solution 1 (2) : RMB > Selection

Data Sets - Study 1/Solution 1 (2) - Selection:

`Geometric Entity Selection` – **Geometric entity level** = `Domain`

`Geometric Entity Selection` – **Selection** = `5` (*центральний квадрат*)

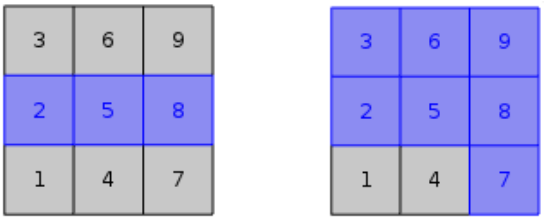
Після створення копії джерела даних ми додали підвузол **Selection**, який для неї (копії) дозволяє обрати рівень геометрії, що буде використовуватися. В нашому випадку ми обрали центральний квадрат. Повторіть команду дублювання ДД.

Data Sets - Study 1/Solution 1 (2) : RMB > Duplicate

Data Sets - Study 1/Solution 1 (3) - Selection:

`Geometric Entity Selection` – **Selection** = `2,5,8`

(*середні квадрати на наступному рисунку ліворуч*)



Ще раз повторіть команду дублювання ДД.

Data Sets - Study 1/Solution 1 (3) : RMB > Duplicate

Data Sets - Study 1/Solution 1 (3) - Selection:

Geometric Entity Selection – **Selection** = 2,3,5,6,7,8,9

(*попередній рисунок праворуч*)

Побудуйте графік функції arg(Z) в центральному квадраті.

Results -> **Plot Group** ->2D Plot Group

2D Plot Group 1:

Data – **Data set** = Study 1/Solution 1(2)

Color Legend – **Show maximum and minimum values** = True

2D Plot Group 1-> **Add Plot** -> Surface

2D Plot Group 1 – Surface 1:

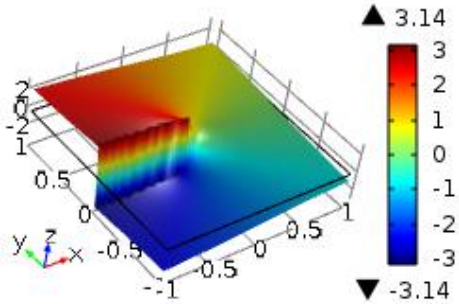
Expression – **Expression** = arg(Z)

RMB> Height Expression

2D Plot Group 1 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = True,0.1

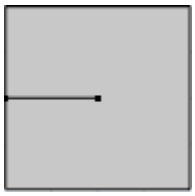
LMB>Plot

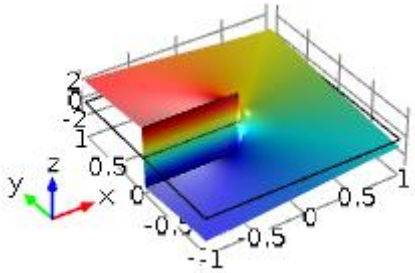


З графіка зрозуміло, що вираз arg(Z) зазнає розриву на від'ємній півосі. Максимальні і мінімальні позначки ±3.14 зображають значення аргументу Z по різні боки розриву.

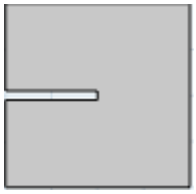
Зауваження. З останнього графіка випливає, що комплексні функції в COMSOL інколи можуть вести себе не так, як це прийнято в теорії функцій комплексного змінного. Дійсно, частина функцій $f(z)$ при переході їх параметра z через від'ємну півось в напрямку проти годинникової стрілки очікують, що аргумент цього параметра z продовжуватиме зростати. Ця особливість в більшості задач несуттєва, але інколи її треба враховувати. Наприклад, функція COMSOL обчислення квадратного кореня від комплексного числа sqrt(z) повертає результат з аргументом $-\pi/2 \le \arg z \le \pi/2$, що при побудові конформних відображень, вирази яких містять цю функцію, приводитиме до некоректних результатів. Скорегувати таку особливість можна, створивши функцію користувача, яка обчислюватиме квадратний корінь від комплексного числа зі значенням в діапазоні $0 \le \arg z \le \pi$.

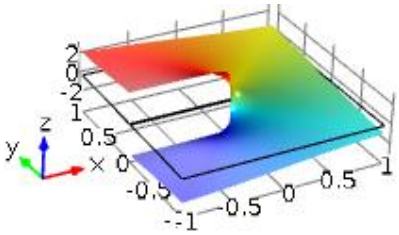
Вправа. Якщо в розрахунковій геометрії по лінії розриву накреслити відрізок, то він змусить систему розташувати на ньому вузли СкЕ сітки. Тоді зображення розриву функції arg(Z) виглядатиме рівніше.





Замість відрізка можна намалювати вузький (але не занадто) прямокутник, який видалити з розрахункової геометрії. Тоді розрив функції виглядатиме як розрив.





Виконайте зазначені побудови. Інше рішення для справжнього зображення розриву полягає в застосуванні в графічних вузлах фільтрів. Приклад цього буде наведено наприкінці параграфа.

■

Побудуйте графік функції `abs(Z)` в центральному квадраті.

`Results` **-> Plot Group ->**2D Plot Group

2D Plot Group 2:

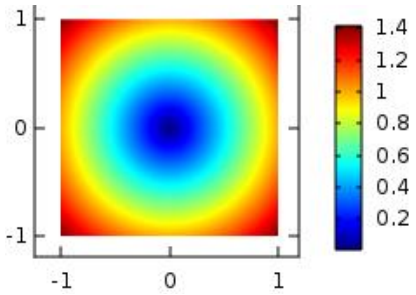
Data – **Data set** = `Study 1/Solution 1(2)`

`2D Plot Group 1`-> **Add Plot ->** Surface

2D Plot Group 2 – Surface 1:

Expression – **Expression** = `abs(Z)`

LMB>Plot



Побудуйте графік функції `abs(cos(Z))` в квадратах `2,5,8`. Для цього виконайте інструкції:

2D Plot Group 1: RMB > Duplicate

2D Plot Group 3:

Data – **Data set** = `Study 1/Solution 1(3)`

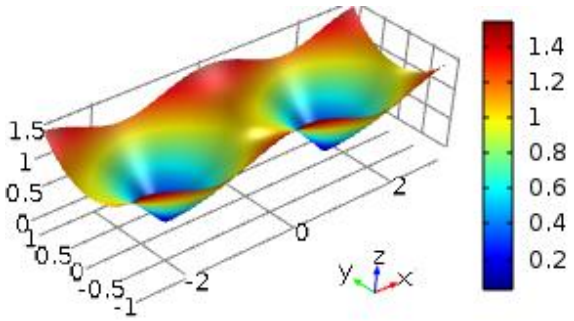
2D Plot Group 3 – Surface 1:

Expression – **Expression** = `abs(cos(Z))`

2D Plot Group 3 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = `True,1`

LMB>Plot



Рельєф модуля можна доповнити лініями рівня аргументу.

2D Plot Group 3: RMB > Contour

2D Plot Group 3 – Contour 1:

Expression – **Expression** = `arg(cos(Z))`

Levels – **Entry method**= `Levels`

Levels – **Levels** = `range(-pi,0.5,pi)`

Coloring and Style – **Level Labels** = `True`

– **Coloring, Color** = `Uniform, Black`

– **Color legend** = `False`

LMB>Plot (*наступний рисунок ліворуч*)

Ці криві можна розташувати на поверхні модуля, додавши ще її лінії рівня.

2D Plot Group 3 – Contour 1: RMB > Height Expression

2D Plot Group 3 – Contour 1 – Height Expression 1:

Expression – **Height data** = `Expression`

– **Expression** = `abs(cos(Z))`

Axis – **Scale factor** = `True,1`

2D Plot Group 3 – Contour 1:

Coloring and Style – **Level Labels** = `False`

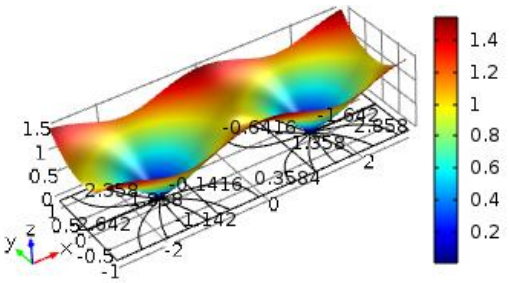
2D Plot Group 3 – Contour 1: RMB > Duplicate

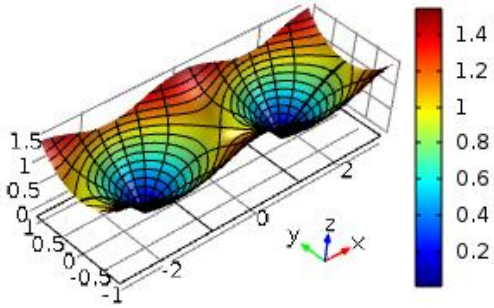
2D Plot Group 3 – Contour 2:

Expression – **Expression** = `abs(cos(Z))`

Levels – **Levels** = `range(0,0.1,1.4)`

LMB>Plot (*наступний рисунок праворуч*)





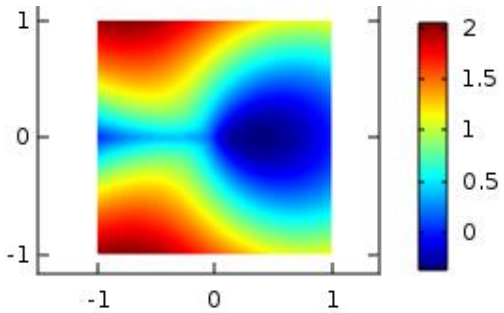
Протестуємо роботу оператора `realdot`, побудувавши графік функції `realdot(Z,log(Z))`.

2D Plot Group 2: RMB > Duplicate

2D Plot Group 4 – Surface 1:

Expression – **Expression** = `realdot(Z,log(Z))`

LMB>Plot



Побудуйте графік функції `abs(tan(Z))`, для наочності виключивши з геометрії квадрати 1 і 4.

2D Plot Group 3: RMB > Duplicate

2D Plot Group 5:

`Data` – **Data set** = `Study 1/Solution 1(4)`

2D Plot Group 5 – Surface 1:

`Expression` – **Expression** = `abs(tan(Z))`

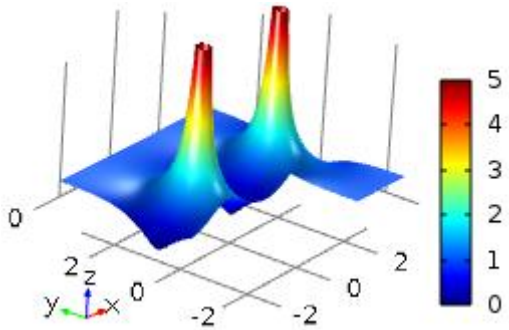
`Range` – **Manual color range** = `True`

– **Minimum, Maximum** = `0, 5`

`Range` – **Manual data range** = `True`

– **Minimum, Maximum** = `0, 5`

LMB>Plot



Функція `sign(Z)` повертає напрямок комплексного числа, тобто одиничний вектор вздовж радіуса–вектора відповідної комплексної точки. Наприклад, надрукуйте значення `sign(2+1.5*i)`.

`Results` **`->`** **`Derived Values`** – Global Evaluation

Derived Values - Global Evaluation 1:

`Expression` – **Expression** = `sign(2+1.5*i)`

LMB > Evaluate

Буде отримано значення `0.80000+0.60000i`.

Тепер побудуйте векторне поле $(\mathrm{Re}[\mathrm{sign}(Z)], \mathrm{Im}[\mathrm{sign}(Z)])$ - це буде поле одиничних векторів, спрямованих вздовж радіуса-вектора точок.

`Results` **`-> Plot Group ->`**2D Plot Group

2D Plot Group 6:

`Data` – **Data set** = `Study 1/Solution 1(1)`

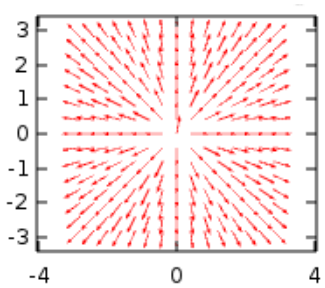
`Plot Settings` – **Plot data set edges** = `False`

`2D Plot Group 6->` **`Add Plot ->`** Arrow Surface

2D Plot Group 6 – Arrow Surface 1:

`Expression` – **x component** = `real(sign(Z))`

Expression – **y component** = `imag(sign(Z))`
LMB>Plot

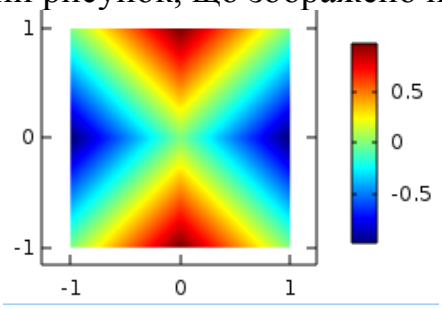


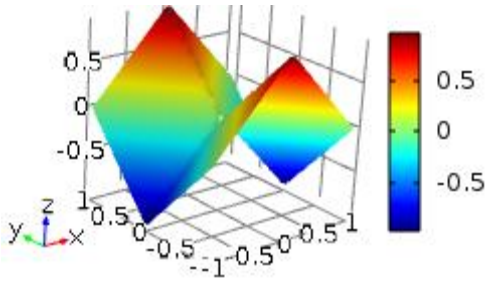
В операторі `integrate` ми не можемо вказати комплексні межі інтегрування, але межі, що залежать від просторових координат, вказувати можна. Наприклад, побудуємо графік виразу $\int_x^y \operatorname{sign}(t)\,dt$.

2D Plot Group 4: RMB > Duplicate
2D Plot Group 7 – Surface 1:

Expression – **Expression** = `integrate(sign(t),t,x, y)`
LMB>Plot (*наступний рисунок ліворуч*)

Якщо додати підвузол `Height Expression` (зробіть це самостійно), то буде побудований рисунок, що зображено праворуч.





Можна зображати зони комплексної площості. Для цього корисно створити джерело даних типу `Grid 2D`. Такі джерела даних використовуються для обчислення функцій та змінних тоді, коли немає потреби прив'язуватися до СкЕ сітки та форми області (форма ДД `Grid` співпадає з прямокутником або прямокутним паралелепіпедом). В нашому випадку ми можемо створити прямокутник з дуже дрібною прямокутною сіткою, що значно поліпшить подальші зображення. Ці зображення будуть виглядати краще, якщо навколо них буде відображатися прямокутна межа, для якої ми створимо ДД типу `Edge 2D`.

`Results`-> More Data Sets – Grid 2D

Data Sets – Grid 2D 1:

Data – **Source**= `Data set`
Data – **Data set** = `Study 1/Solution 1(1)`
Parameter Bounds – First parameter – **Name:** = `x`
– **Minimum, Maximum** = `-3 ,3` (*два поля*)
Parameter Bounds – Second parameter – **Name:** = `y`
– **Minimum, Maximum** = `-3 ,3` (*два поля*)
Resolution – **x resolution, y resolution:** = `300,300` (*два поля*)

Отже координати в площині нового ДД будуть позначатися x,y так, як і в геометрії моделі.

Створіть ДД, яке буде представляти контур області.

Results-> More Data Sets – Edge 2D

Data Sets - Edge 2D 1 :

Data – **Data set** = Study 1/Solution 1(1)

Selection – **Selection** =Manual, 1,2,3,5,7,9,14,16,21,22,23,24

(*зовнішній контур масиву квадратів*)

Тепер намалюйте зону комплексної площини imag(Z^7)<0.

Results **-> Plot Group ->**2D Plot Group

2D Plot Group 8:

Data – **Data set** = Grid 2D 1

2D Plot Group 1-> **Add Plot ->** Surface

2D Plot Group 8 – Surface 1:

Expression – **Expression** = imag(Z^7)<0

Coloring and Style – **Color table** = GrayScale

2D Plot Group 8 : RMB> Line

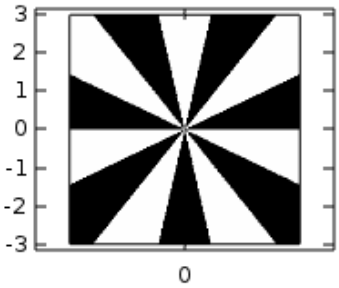
2D Plot Group 8 – Line 1:

Data – **Data set** = Edge 2D 1

Expression – **Expression** = 1

Coloring and Style – **Coloring, Color** = Uniform, Black

LMB>Plot



Подивимось, як виглядає область комплексної площини $-1 \le \operatorname{Im} z^2 \le 1$.

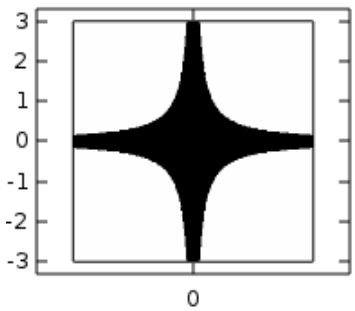
2D Plot Group 8: RMB > Duplicate

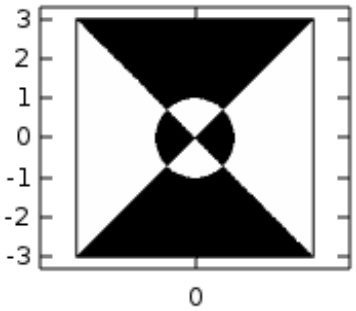
2D Plot Group 9 – Surface 1:

Expression – **Expression** = imag(Z^2)<1 && imag(Z^2)>-1

Coloring and Style – **Reverse color table** = True

LMB > Plot (*наступний рисунок ліворуч*)





«Зобразимо» ще одну нерівність.

2D Plot Group 9: RMB > Duplicate

2D Plot Group 10 – Surface 1:

Expression – **Expression** = `real(Z^2)>real(1/Z^2)`

Coloring and Style – **Reverse color table** = `False`

LMB>Plot (*попередній рисунок праворуч*)

Над ДД **Grid 2D 1** можна будувати поверхні.

2D Plot Group 3: RMB > Duplicate

2D Plot Group 11:

Data – **Data set** = `Grid 2D 1`

Color Legend – **Show maximum and minimum values** = `False`

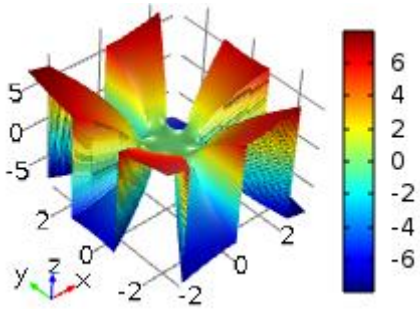
2D Plot Group 11 – Surface 1:

Expression – **Expression** = `real(asinh(Z^5))`

2D Plot Group 11 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = `True,0.3`

LMB>Plot



Існує можливість більш реальної побудови цієї розривної поверхні, якщо використати фільтри. Вони можуть залишити одну чи іншу частину поверхні, яка задовольняє якимось логічним умовам. В нашому випадку це можуть бути умови додатності чи від'ємності зазначеного виразу.

Фільтри не працюють з ДД типу `Grid`, тому графіки треба будувати над ДД `Study/Solution`. Але фільтри за замовчуванням «обрізають» графіки по границям скінченних елементів. Тому щоб краї обрізів були гладкішими подрібніть СкЕ сітку. Для цього в ПВ вузла **Mesh 1** введіть

Mesh 1:

Mesh Settings –**Element size** = `Extremely fine`

і натисніть кнопку Build All. Потім заново обчисліть модель.

Study 1: LMB > Compute

Для графічної групи **2D Plot Group 11** змініть ДД.

2D Plot Group 11:

Data – **Data set** = Study 1/Solution 1 (1)

Дублюйте вузол **2D Plot Group 11 – Surface 1** і додайте до створюваного вузла фільтр.

2D Plot Group 11 – Surface 1 : RMB > Duplicate

2D Plot Group 11 – Surface 2 :

Range – **Manual color range** = True

– **Minimum, Maximum** = -8 , 8 (*два поля*)

Coloring and Style – **Color legend** = False

2D Plot Group 11 – Surface 2 : RMB > Filter

2D Plot Group 11 – Surface 2 – Filter 1 :

Element Selection – **Logical expression for inclusion** = real(asinh(Z^5))>0

Ще раз виконайте дублювання вузла Surface.

2D Plot Group 11 – Surface 2 : RMB > Duplicate

2D Plot Group 11 – Surface 3 – Filter 1 :

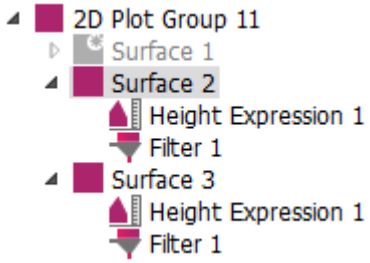
Element Selection – **Logical expression for inclusion** = real(asinh(Z^5))<0

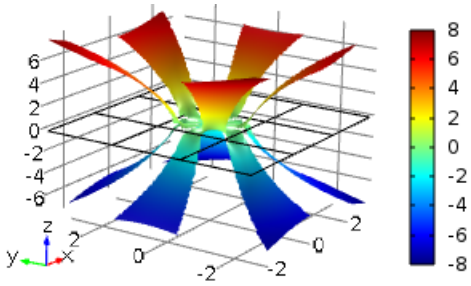
Coloring and Style – **Color legend** = True

Деактивуйте вузол **Surface 1** і побудуйте графік.

2D Plot Group 11 – Surface 1 : RMB > Disable

LMB>Plot





Незручність показаного підходу полягає в тому, що потрібно вручну задавати спільний діапазон зміни величини, інакше кольорові палітри двох графіків Surface не будуть узгоджені. Щоб їх погодити, ми у вузлі **Surface 2** встановили значення полів **Minimum, Maximum** =∓8, а вузол **Surface 3** їх успадкував.

Потрібно пам’ятати, що графічні вузли завжди відображають дійсну частину комплексного виразу, навіть, якщо функція real не використовується.

2D Plot Group 4: RMB > Duplicate

2D Plot Group 12:

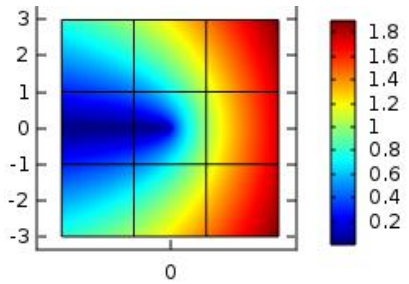
Data – **Data set** = Study 1/Solution 1 (1)

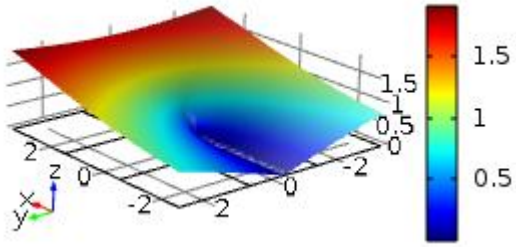
2D Plot Group 12 – Surface 1:

Expression – **Expression** = sqrt(Z)

LMB>Plot (*наступний рисунок ліворуч*)

Якщо додати підвузол Height Expression (зробіть це самостійно), то буде побудований рисунок, що зображено праворуч.





Замініть останній вираз на `real(sqrt(Z))` і ви отримаєте той же графік.

Незважаючи на те, що графічні вузли завжди обробляють дійсну частину виразу, комплексні дані можна друкувати в таблицях. При цьому друкуються обидві частини числа: дійсна і уявна.

Обчислимо значення комплексного виразу `Z^2` в точці `(2,1)`.

`Results` -> **`Derived Values`** – Global Evaluation

Derived Values - Global Evaluation 2:

`Expression` – **Expression** = `at2(2,1,Z^2)`

LMB > Evaluate

В таблиці **Table 1** буде надруковано значення `3.0000+4.0000i`.

Обчисліть значення комплексного виразу $\sqrt{Z}$ у вузлі `(3,1)`.

Derived Values: RMB > Point Evaluation

Point Evaluation 1:

`Data` – **Data set** = `Study 1/Solution 1 (1)`

`Selection` – **Selection** = `Manual,15` (*індикатор вузла* (3,1))

`Expression` – **Expression** = `sqrt(Z)`

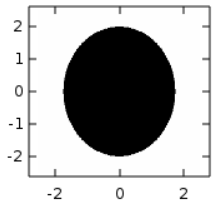
LMB > Evaluate –Table 1

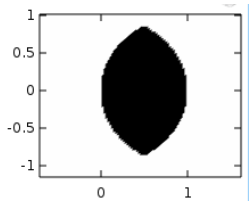
В таблиці **Table 1** буде додано комірку зі значенням `1.7553+0.28485i`.

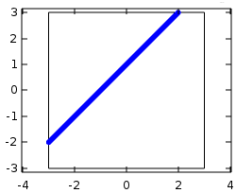
at2(2,1,Z^2)	sqrt(Z), Point: 15
3.0000+4.0000i	1.7553+0.28485i

Вправа. Зобразити на комплексній площині множини точок:

1. $|z-i|+|z+i|<4$;
2. $|z|<1$, $|z-1|<1$;
3. $\operatorname{Im}\dfrac{z+1}{z-i}=0$.

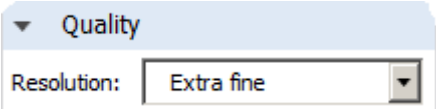
1. 

2. 

3. 

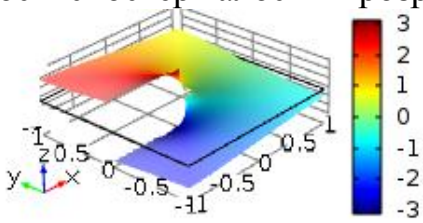
$|z-i|+|z+i|<4$ $\qquad$ $|z|<1$, $|z-1|<1$ $\qquad$ $\operatorname{Im}\dfrac{z+1}{z-i}=0$

Зауваження. Якщо у фігур спостерігатиметься занадто зубчастий контур, то для його поліпшення у `Surface` та `Contour` графічних вузлах застосуйте опцію **`Quality`** – **Resolution** = `Extra fine`



Зауваження. Контурні лінії комплексних виразів, які будуються на прямокутній (mapped) СкЕ сітці, в COMSOL 5.1 можуть креслитися неправильно!

Вправа. Використовуючи фільтри, перебудуйте графік функції `arg(Z)` так, щоб порушення неперервності спостерігалося як розрив поверхні.



8.2. Обчислення інтегралів.

Часто при розв'язанні задач, які містять роботу з комплексними функціями, доводиться обчислювати інтеграли типу $\oint_C u(\zeta)d\zeta$, де C – крива на комплексній площині. Безпосередньо в COMSOL інструментів обчислення таких інтегралів немає, але їх можна реалізувати за допомогою `Coupling` операторів. Для цього, по-перше, в розрахунковій області потрібно мати криву C. По-друге, треба створити оператор інтегрування по цій кривій. Він буде реалізовувати обчислення криволінійного інтегралу першого роду $\oint_C u\,d\,l$. Далі треба згадати про вбудований одиничний дотичний вектор до кривої $\boldsymbol{t}=(t_x,t_y)$. Вираз $(t_x+i\cdot t_y)\,d\,l$ можна проінтерпретувати як $d\,\zeta$ - диференціал комплексної змінної вздовж кривої C (у відповідному напрямку). А інтеграл $\oint_C u\cdot(t_x+i\cdot t_y)\,d\,l$ можна буде інтерпретувати як $\oint_C u\,d\,\zeta$. Отже після створення `Coupling` оператора інтегрування вздовж кривої C, наприклад, з ім'ям `intop1`, для обчислення інтеграла $\oint_C u\,d\,\zeta$ в **Expression** полях потрібно буде вводити вирази типу `intop1(u(z)*(tx+i*ty))`, а для ідентифікатора `z` у вузлі змінних **Variables** задати вираз `x+i*y`.

Приклад 1. Обчислити наступні інтеграли:

1. $\int\limits_L \operatorname{Im} z\,d\,z$ – по контуру півкола $|z|=3$, $0\le\arg z\le\pi$ (відповідь $-9\pi/2$);
2. $\int\limits_C \frac{1}{z}d\,z$ – по контурам C_1, C_2 верхнього і нижнього півкола $|z|=1$ (відповідь $\pm\pi i$);
3. $\int\limits_L \frac{\sin(\pi z/4)}{z^2-1}d\,z$ – по окружності $|z-1|=1$ (відповідь $\pi i\sqrt{2}/2$);
4. $\int\limits_L (2z+1)\bar{z}\,d\,z$ – по окружності $|z|=1$ (відповідь $2\pi i$);

5. $\int\limits_L \frac{e^{2z}}{z+\pi i} dz$ – по окружності $|z+i|=1$ (відповідь 0).

Р о з в ’ я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Додайте в неї двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Встановіть прапорець, який включає режим відображення напрямку зростання параметрів вздовж кривих.

Component 1 – Definitions - View 1:

`View` – Show edge direction arrows = `True`

Розрахункову геометрію створіть як об’єднання фігур, контури яких згадуються в умовах задачі.

`Geometry` -> **`Primitives`** – Circle

Geometry 1 - Circle 1 (c1):

`Size and Shape` – Radius = `3`

`Geometry` -> **`Primitives`** – Circle

Geometry 1 - Circle 2 (c2):

`Size and Shape` – Radius = `1`

Наступними інструкціями створіть два зсунутих одиничних кола.

`Geometry` -> **`Operations - Transforms`** – Move

Geometry 1 - Move 1 (mov1):

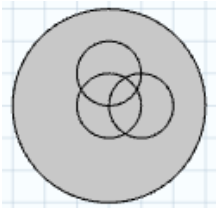
`Input` – Input objects = `c2`

– Keep input objects = `True` (*залишити базове коло*)

`Displacement` – x = `1,0` (*абсциси векторів зсуву*)

– y = `0,1` (*ординати векторів зсуву*)

LMB > Build All Objects



Побудуйте СкЕ сітку з параметрами за замовчуванням.

Mesh 1 : LMB > Build All

Створіть таблицю змінних моделі з ідентифікаторами незалежної змінної z, та трьох підінтегральних виразів з умов 3 – 5.

Definitions: RMB > Variables

Definitions – Variables 1:

`Variables` – Name, Expression = `z, x+i*y`

`Variables` – Name, Expression = `expr3, sin(pi*z/4)/(z^2-1)`

`Variables` – Name, Expression = `expr4,(2*z+1)*conj(z)`

`Variables` – Name, Expression = `expr5, exp(2*z)/(z+pi*i)`

Variables

Name	Expression
z	x+i*y
expr3	sin(pi*z/4)/(z^2-1)
expr4	(2*z+1)*conj(z)
expr5	exp(2*z)/(z+pi*i)

Створіть оператори інтегрування по кривим, зазначеним в умовах.

`Definitions` > `Component Couplings` - Integration

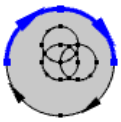
Definitions - Integration 1 (intop1):

Operator name = `intUpC3`

`Source Selection` – **Geometric entity level** = `Boundary`

Остання інструкція вказує, що інтегрування буде виконуватися по кривій, яка визначатиметься в наступній інструкції.

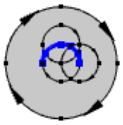
`Source Selection` – **Selection** = `Manual,2,16`



Definitions - Integration 1: RMB > Duplicate

Definitions - Integration 2: Operator name = `intUpC1`

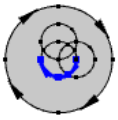
`Source Selection` – **Selection** = `3,8,14,19,21`



Definitions - Integration 1: RMB > Duplicate

Definitions - Integration 3: Operator name = `intDownC1`

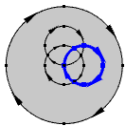
`Source Selection` – **Selection** = `4,10,18`



Definitions - Integration 1: RMB > Duplicate

Definitions - Integration 4: Operator name = `intRShiftC1`

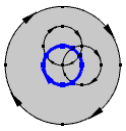
`Source Selection` – **Selection** = `11,12,17,20,23,24`



Definitions - Integration 1: RMB > Duplicate

Definitions - Integration 5: Operator name = `intC1`

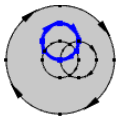
`Source Selection` – **Selection** = `3,4,8,10,14,18,19,21`



Definitions - Integration 1: RMB > Duplicate

Definitions - Integration 6: Operator name = `intUpShiftC1`

`Source Selection` – **Selection** = `5,6,7,13,15,22`



Щоб оператори і змінні працювали, потрібно виконати обчислення, а для цього в моделі треба мати вузол дослідження.

Untitled.mph (root): RMB > Add Study = `Add Study` –> Preset Studies – Stationary –> X

Study 1: LMB > Compute

Всі інтеграли обчислюватимемо в гілці **Derived Values** у вузлах типу `Global Evaluation`. Створіть перший такий вузол.

`Results` -> **`Derived Values`** – Global Evaluation

Derived Values - Global Evaluation 1:

Label = `Int_imag(z)`

`Expression` – **Expression** = `intUpC3(imag(z)*(tx+i*ty))`

LMB > Evaluate

Результат `14.137` з'явиться в таблиці. Перейменуйте таблицю.

Results – Tables – Table 1: Label = `T_imag(z)`

Для перевірки створіть ще один вузол `Global Evaluation`, з якого в ту ж таблицю надрукуйте точну відповідь $-9\pi/2$.

Derived Values - Int_imag(z): RMB > Duplicate

Derived Values - Int_imag(z) 1:

`Expression` – **Expression** = `-9*pi/2`

LMB > Evaluate - T_imag(z)

intUpC3(imag(z)*(tx+i*ty))	-9*pi/2
14.137	-14.137

З пояснювального рисунка інструкції створення оператора `intUpC3` видно, що в ньому інтегрування виконується за годинниковою стрілкою, а значення $-9\pi/2$ отримано при інтегруванні у протилежному напрямку. Тому відповіді відрізняються знаком.

Виконайте ще 5 разів команду

Derived Values - Int_imag(z): RMB > Duplicate

Заповніть поля виразів **Expression** створених вузлів `Global Evaluation` так, як показано в таблиці. Поміняйте також назви цих вузлів і створюваних після їх обчислення нових таблиць (перші два рядка таблиці відповідають щойно створеним вузлам Global Evaluation.

Вузол (нова назва)	Expression	Відповідь	Таблиця
Int_imag(z)	`intUpC3(imag(z)*(tx+i*ty))`	`14.137`	**T_imag(z)**
Int_imag(z) 1	`-9*pi/2`	`-14.137`	**T_imag(z)**
Int_1/z_UpC1	`intUpC1(1/z*(tx+i*ty))`	`-3.1416i`	**T_1/z**
Int_1/z_DownC1	`intDownC1(1/z*(tx+i*ty))`	`-3.1416i`	**T_1/z**
Int_RShift_C1	`intRShiftC1(expr3*(tx+i*ty))`	`-2.2214i`	**T_ RShift_C1**
Int_C1	`intC1(expr4*(tx+i*ty))`	`-6.2831i`	**T_ C1**
Int_UpShift_C1	`intUpShiftC1(expr5*(tx+i*ty))`	`5.8693E-5-5.5775E-5i`	**T_ UpShift_C1**

Самостійно перевірте збіг відповідей з точними значеннями інтегралів, наведеними на початку прикладу. Звертайте увагу на знаки відповідей. Вони

можуть відрізнятися внаслідок вибору системою протилежного порівняно з умовою напрямку обходу контура. Значення в таблиці **T_ UpShift_C1** можна вважати нульовим. Якщо у вузлі **Mesh 1** подрібнити сітку, то дійсна і уявна частини цієї відповіді стануть ближчими до нуля. ■

Нехай область D відображається за допомогою комплексної функції $f(z)$ конформно і взаємно однозначно на область G. Тоді площа області G може бути обчислена за формулою [3]

$$S = \iint_D |f'(z)|^2 \, dx\, dy. \qquad (1)$$

Якщо дуга C цілком належить області D, то довжина дуги L її образа може бути обчислена за формулою

$$L = \oint_C |f'(z)| \, dl \qquad (2)$$

Приклад 2. Застосування формул (1) і (2).

1. Знайти довжину дуги спіралі, на яку за допомогою функції e^z відображається відрізок $y = x$, $0 \le x \le 2\pi$.

2. Знайти площу області, на яку за допомогою функції e^z відображається прямокутник $1 \le x \le 2$, $0 \le y \le 4$.

Побудувати образи зазначених фігур.

Р о з в ' я з а н н я. Створіть порожню модель.

File-> New -> Blank Model

Додайте в неї двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Розрахункову область створіть у вигляді квадрата $[0, 2\pi] \times [0, 2\pi]$ (він буде зображати частину нескінченної площини $z = x + i \cdot y$).

`Geometry->`**`Primitives->`** Square

Geometry 1 - Square 1 (sq1):

`Size` – Size length: = `2*pi`

LMB > Build Selected

Для подальшого зображення фігур, довжина і площа яких обчислюватиметься, побудуйте в геометрії криву $y = x$ і прямокутник $1 \le x \le 2$, $0 \le y \le 4$.

`Geometry->`**`Primitives->`** Parametric Curve

Geometry 1 - Parametric Curve 1 (pc1) :

`Parameter` – Name, Minimum, Maximum = `s,0, 2*pi` (*три поля*)

`Expressions` – x, y = `s, s` (*два поля*)

`Geometry` **`-> Primitives`** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

`Size` – Width, Height = `1,4` (*два поля*)

`Position` – Base, x, y = `Corner,1,0` (*три поля*)

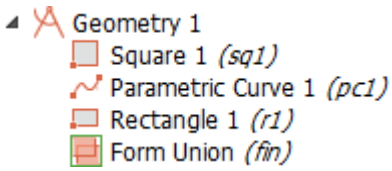
LMB > Build All Objects (*наступний рисунок в центрі*)

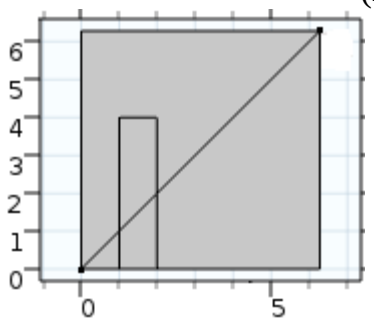
Побудуйте СкЕ сітку.

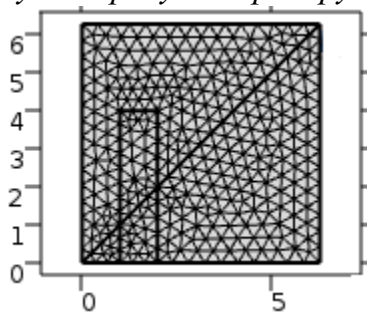
Mesh 1: `Mesh Settings` –Element size = `Finer`

LMB > Build All

(*наступний рисунок праворуч*)







Створіть оператори інтегрування по фігурам, зазначеним в умовах.

`Definitions` > `Component Couplings` - Integration

Definitions - Integration 1 (intop1):

Operator name = `intop1`

`Source Selection` – **Geometric entity level** = `Boundary`

Остання опція вказує, що інтегрування буде виконуватися по лінії.

– **Selection** = `Manual,12,13,14`

`Definitions` > `Component Couplings` - Integration

Definitions - Integration 1 (intop1):

Operator name = `intop2`

`Source Selection` – **Geometric entity level** = `Domain`

Остання опція вказує, що інтегрування буде виконуватися по області.

– **Selection** = `Manual,3,4`

Створіть і заповніть вузол змінних.

`Definitions` -> **`Variables`** – Local Variables

Definitions - Variables 1: (*наступні введення в таблиці*)

`Variables` – **Name, Expression =** `z,x+i*y`

– **Name, Expression =** `w, exp(z)`

– **Name, Expression =** `L, intop1(abs(d(w,z)))`

– **Name, Expression =** `S, intop2(abs(d(w,z))^2)`

Variables

Name	Expression
z	x+i*y
w	exp(z)
L	intop1(abs(d(w,z)))
S	intop2(abs(d(w,z))^2)

Створіть в моделі вузол досліджень і виконайте його обчислення.

Untitled.mph (root): RMB > Add Study = `Add Study` ->

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Щоб побудувати образи фігур нам знадобляться джерела даних, звужені на множини точок лінії і прямокутника.

Results – Data Sets - Study 1/Solution 1: RMB > Duplicate

Data Sets - Study 1/Solution 1 (2) : RMB > Selection

Study 1/Solution 1 (2) - Selection :

`Geometric Entity Selection` – **Geometric entity level** = `Boundary`

– **Selection** = `Manual,12,13,14`

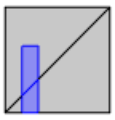
Results – Data Sets - Study 1/Solution 1 (1): RMB > Duplicate

Data Sets - Study 1/Solution 1 (3) : RMB > Selection

Study 1/Solution 1 (3) - Selection :

`Geometric Entity Selection` – **Geometric entity level** = `Domain`

– **Selection** = `Manual,3,4`



Тепер побудуйте образ відрізка $y = x$, $0 \le x \le 2\pi$.

`Results` -> `Plot Group` ->2D Plot Group

2D Plot Group 1:

`Data` – **Data set** = `Study 1/Solution 1(2)`

`Plot Settings` – **Plot data set edges =** `False`

`Color Legend` – **Show legends =** `False`

`2D Plot Group 1`-> `Add Plot` -> Line

2D Plot Group 1 – Line 1:

`Expression` – **Expression** = `1`

`Coloring and Style` – **Line type** = `Tube`

– **Tube radius expression** = `1`

– **Radius scale factor** = `True,1`

– **Coloring, Color** = `Uniform, Blue`

RMB> Deformation

2D Plot Group 2 – Line 1 – Deformation 1:

`Expression` – **x component** = `real(w)-x`

`Expression` – **y component** = `imag(w)-y`

`Scale`– **Scale factor** = `True,1`

LMB > Plot (*наступний рисунок ліворуч після масштабування*)

Якщо вузол **Deformation 1** не створювати, то буде зображатися початковий відрізок $y = x$, $0 \le x \le 2\pi$. Вектор у вузлі Deformation визначає зсув точок. Віднімання вектора (x,y) призводить до зсуву точок відрізка в початок координат, а додавання (`real(w)`,`imag(w)`) визначає їх нове положення.

Побудуйте образ прямокутника $1 \le x \le 2$, $0 \le y \le 4$.

`Results` -> `Plot Group` -> 2D Plot Group

2D Plot Group 2:

`Data` – **Data set** = `Study 1/Solution 1(3)`

`Plot Settings` – **Plot data set edges =** `False`

Color Legend – **Show legends =** False

2D Plot Group 1-> **Add Plot ->** Surface

2D Plot Group 2 – Surface 1:

Expression – **Expression** = 1

Coloring and Style – **Coloring, Color =** Uniform,Black

RMB> Deformation

2D Plot Group 2 – Surface 1 – Deformation 1:

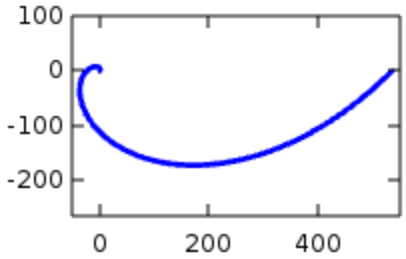
Expression – **x component** = real(w)-x

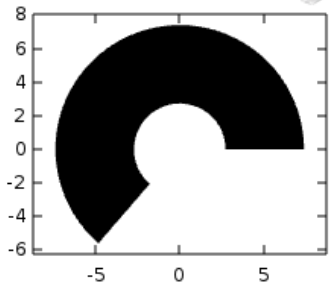
Expression – **y component** = imag(w)-y

Scale– **Scale factor** = True,1

Сенс вузла **Deformation 1** такий самий, як і для вузла **Line 1**. Віднімання радіуса-вектора (x,y) призводить до зсуву точок прямокутника в початок координат, а додавання (real(w),imag(w)) визначає їх нове положення.

LMB>Plot (*наступний рисунок праворуч після масштабування*)





Залишилося надрукувати значення змінних L і S, які представляють довжину кривої, зображену ліворуч, і площу області, зображену праворуч.

Results -> **Derived Values** – Global Evaluation

Derived Values - Global Evaluation 1:

Expression – **Expression** = L

LMB > Evaluate

Буде отримано значення 755.89.

Results -> **Derived Values** – Global Evaluation

Derived Values - Global Evaluation 2:

Expression – **Expression** = S

LMB > Evaluate – Table 1

L	S
755.89	94.418

Зауваження. Для розглянутих інтегралів відомі точні значення $L=\sqrt{2}\left(e^{2\pi}-1\right)$ і $S=2e^{2}\left(e^{2}-1\right)$. Порівняйте їх з надрукованими в таблиці. ■

Проаналізуємо, як можна обчислювати інтеграли Коші [2].

$$f(z)=\frac{1}{2\pi i}\oint_{C}\frac{\varphi(\zeta)}{\zeta-z}d\zeta\,, \qquad (3)$$

де C – контур області D, який проходиться так, щоб область залишалась весь час ліворуч. Формула (3) відрізняється від інтегралів першого прикладу наявністю параметра z в підінтегральному виразі. Вона створює аналітичну в області D функцію $f(z)$ по значенням неперервної на контурі C функції $\varphi(\zeta)$.

Зауваження. Якщо $\varphi(\zeta)$ є довільною на контурі C функцією (не аналітичною), то $f(z)$ все одно буде аналітичною в області D, але в такому разі $f(z)$ не прямуватиме до $\varphi(\zeta)$, коли z прямує до точок ζ контура.

При обчисленні в COMSOL інтеграла $\oint_C \frac{\varphi(\zeta)}{\zeta - z} d\zeta$, щоб розрізнити змінну інтегрування ζ і зовнішній параметр/змінну z, потрібно застосовувати оператор `dest`, наприклад, наступним чином

```
intop1(fi(z)/(z-dest(z))*(t1x+i*t1y))
```

де `intop1` є оператором інтегрування по заданій кривій.

Приклад 3. За формулою (3) побудувати аналітичну в одиничному колі функцію $f(z)$ при $\varphi(\zeta) = x^2 + i\,y$. Оскільки

$$x = \frac{1}{2}\left(z + \bar{z}\right),\ y = \frac{1}{2i}\left(z - \bar{z}\right),\ \text{ то } \varphi(\zeta) = \frac{1}{4}\left(\zeta + \bar{\zeta}\right)^2 + \frac{1}{2}\left(\zeta - \bar{\zeta}\right).$$

Р о з в ' я з а н н я . Створіть порожню модель.

File-> New -> Blank Model

Додайте в неї двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть перевірку розмірностей фізичних величин і встановіть прапорець, який включає режим відображення напрямку кривих.

Untitled.mph (root): `Unit System` – <> = `None`

Component 1 – Definitions - View 1:

`View` – Show edge direction arrows = `True`

Розрахунковою областю є одиничне коло.

`Geometry` -> **`Primitives`** – Circle

Geometry 1 - Circle 1 (c1):

`Size and Shape` – Radius = `1`

`Layers` – Layer name, Thickness = `0.01`

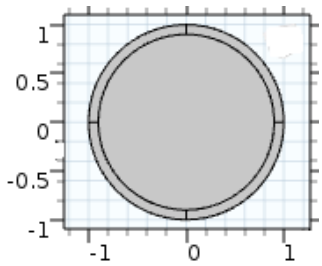
Оскільки формула (3) непридатна для обчислення функції на межі контура, то для близьких до контура точок матимемо суттєву похибку обчислень. Для побудови графіків дійсної і уявної частини створюваної комплексної функції буде кращим виключити з графіка тонку зовнішню зону кола. Для створення таких зон у вузлах деяких графічних примітивів є розділ **`Layers`** з таблицею, в якій вказується товщина зон/шарів та їх назва. Самий зовнішній шар розташований в таблиці першим.

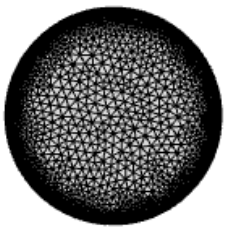
LMB > Build All Objects (*наступний рисунок ліворуч*)

Побудуйте СкЕ сітку.

Mesh 1: `Mesh Settings` –Element size = `Fine`

LMB > Build All (*наступний рисунок праворуч*)





Зверніть увагу, що в зовнішньому кільці і зоні біля нього побудована дуже дрібна сітка.

Створіть оператор інтегрування по контуру кола.

`Definitions` > Component Couplings - Integration

Definitions - Integration 1 (intop1):

Operator name = `intC1`

Source Selection – **Geometric entity level** = `Boundary`

– **Selection** = `Manual, 5,6,9,12`

(*дуги зовнішнього одиничного кола*)

Будьте уважні, не виділіть внутрішні дуги створеного шару.

Побудуйте таблицю змінних з ідентифікаторами незалежної змінної z, виразом для $\varphi(\zeta)$ і змінною, обчислюваною за формулою (3).

Definitions: RMB > Variables

Definitions – Variables 1:

Variables – **Name, Expression** = `z, x+i*y`

Variables – **Name, Expression** =`fb, (z+conj(z))^2/4+(z-conj(z))/2`

Variables – **Name, Expression** = `f,-intC1(fb/(z-dest(z))*(tx+i*ty))/(2*pi*i)`

Variables

Name	Expression
z	x+i*y
fb	(z+conj(z))^2/4+(z-conj(z))/2
f	-intC1(fb/(z-dest(z))*(tx+i*ty))/(2*pi*i)

Мінус поставлено у зв'язку з іншим напрямком обходу кола порівняно з формулою (3), де контур обходиться супротив годинникової стрілки.

Створіть в моделі вузол досліджень і виконайте обчислення.

Untitled.mph (root): RMB > Add Study = `Add Study ->`

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Побудуйте графік дійсної частини змінної `f` в одиничному колі, виключивши з нього тонкий зовнішній шар. Для цього спочатку створіть джерело даних, з геометрії якого виключено цей шар.

Results – Data Sets - Study 1/Solution 1: RMB > Duplicate

Results – Data Sets - Study 1/Solution 1 (2) : RMB > Selection

Results – Data Sets - Study 1/Solution 1 (2) - Selection :

Geometric Entity Selection – **Geometric entity level** = `Domain`

Geometric Entity Selection – **Selection**=`Manual,5` (*внутрішнє коло*)

Тепер створіть і налаштуйте 2D графічну групу.

Results -> **Plot Group** ->2D Plot Group

2D Plot Group 1:

Data – **Data set** = Study 1/Solution 1(2)

Color Legend – **Show maximum and minimum values** = True

2D Plot Group 1-> **Add Plot** -> Surface

2D Plot Group 1 – Surface 1:

Expression – **Expression** = real(f)

RMB> Height Expression

2D Plot Group 1 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = True,1

Для наочності побудуйте над контуром кола графік дійсної частини функції $\varphi(\zeta)$. Для цього спочатку створіть джерело даних Edge 2D, а потім на ньому в поточній графічній групі лінійний графік.

Results – Data Sets : RMB > Edge 2D

Data Sets - Edge 2D 1:

Data – **Data set** = Study 1/Solution 1(1)

Selection – **Selection** = Manual; 5,6,9,12

2D Plot Group 1-> **Add Plot** -> Line

2D Plot Group 1 – Line 1:

Data – **Data set** = Edge 2D 1

Expression – **Expression** = 1

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.03

– **Radius scale factor** = True,1

– **Coloring, Color** = Uniform, Blue

RMB> Height Expression

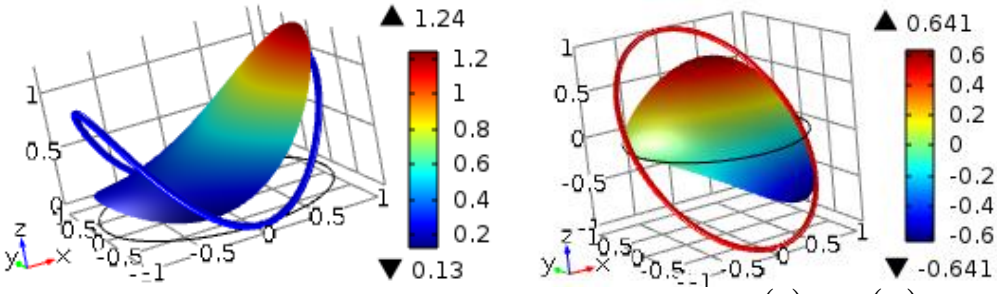
2D Plot Group 1 – Line 1 – Height Expression 1:

Expression – **Height data** = Expression

– **Expression** = x^2

Axis – **Scale factor** = True,1

LMB>Plot (*наступний рисунок ліворуч*)



Аналогічно будуються графіки уявних частин функцій $f(z)$ і $\varphi(\zeta)$ (поверхня и крива над контуром кола).

2D Plot Group 1: RMB > Duplicate

2D Plot Group 2 – Surface 1:

Expression – **Expression** = imag(f)

2D Plot Group 2 – Line 1:

`Coloring and Style` – **Coloring, Color** = `Uniform, Red`

2D Plot Group 2 – Line 1 – Height Expression 1:

`Expression` – **Expression** = `y`

LMB>Plot (*попередній рисунок праворуч*)

Як бачите функції $\mathrm{Re}[f(z)]$ і $\mathrm{Im}[f(z)]$ на контурі кола мають інші значення ніж $\mathrm{Re}[\varphi(\zeta)]$ і $\mathrm{Im}[\varphi(\zeta)]$. Це підтверджує зауваження щодо формули (3), наведене раніше.

Можна створити графічну групу з графіками функцій $\mathrm{Re}[f(z)]$ і $\mathrm{Im}[f(z)]$ одночасно. Але тоді для обох графіків треба встановити спільний діапазон палітри кольорів.

2D Plot Group 1: RMB > Duplicate

2D Plot Group 3 – Line 1: RMB > Disable

2D Plot Group 2 – Surface 1: RMB > Copy

2D Plot Group 3: RMB > Past Surface

2D Plot Group 3:

`Color Legend` – **Show maximum and minimum values** = `False`

2D Plot Group 3 – Surface 1:

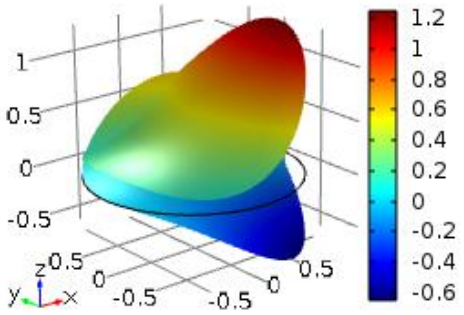
`Range` – **Manual color range, Minimum, Maximum** = `True,-0.65,1.25`

2D Plot Group 3 – Surface 2:

`Range` – **Manual color range, Minimum, Maximum** = `True,-0.65,1.25`

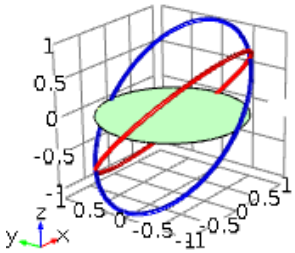
`Coloring and Style` – **Color legend** = `False`

LMB>Plot

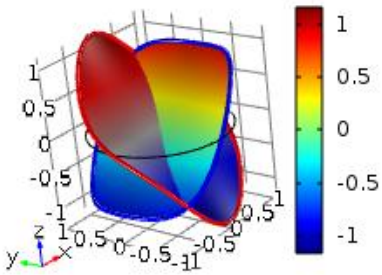


Хоча ці дві поверхні і зображають комплексну функцію, вони не дають достатнього розуміння її поведінки. В наступному параграфі ми ще повернемось до інших способів графічного зображення комплексних функцій. Збережіть файл, але не закривайте його.

Вправа. Побудуйте поверхню дійсної і уявної частин аналітичної в одиничному колі функції $f(z)$, яка обчислюється за формулою (3) при $\varphi(\zeta)=1/\zeta$ [4]. *Відповідь*: $f(z)=0+0\cdot i$.



Вправа. Побудуйте поверхню дійсної і уявної частин аналітичної в одиничному колі функції, яка обчислюється за формулою (3) при $\varphi(\zeta)=\sin\zeta$. Оскільки $\varphi(\zeta)$ є граничним значенням аналітичної функції, то функція $f(z)$ прямуватиме до $\varphi(\zeta)$, коли z наближатиметься до контура кола.



■

В попередньому прикладі ми будували аналітичні функції $f(z)$ за формулою (3), використовуючи функції $\varphi(\zeta)$ на контурі області. Якщо така гранична функція була аналітичною, то $f(z)$ прямувала до $\varphi(\zeta)$, коли z наближалася до точки ζ контура області. Таким чином, щоб $f(z)$ відновлювалася по своїм граничним значенням $\varphi(\zeta)$ останні повинні вже утворювати аналітичний вираз, і як ми бачили це не завжди так.

Відомо, що аналітична в області D функція $f(z)=u(z)+i\cdot v(z)$ відновлюється по її дійсній частині на контурі C області. Це складніша задача, але для деяких областей її розв'язок ми знаємо. Наприклад, аналітичну в одиничному колі функцію $f(z)=u(z)+i\cdot v(z)$ можна відновити по значенню на контурі її дійсної частини $u(\zeta)$ за допомогою інтеграла Шварца [2]

$$f(z)=\frac{1}{2\pi i}\int\limits_{|\zeta|=1}\frac{u(\zeta)}{\zeta}\frac{\zeta+z}{\zeta-z}d\zeta+i\cdot v(0), \qquad (4)$$

де значення $v(0)$ можна обирати довільно. Звідси випливає, що гармонічна в одиничному колі функція $u(x,y)=u(z)$, яка на контурі набуває значення $u(\zeta)$, може бути знайдена як дійсна частина $u(z)=\mathrm{Re}[f(z)]$ аналітичної функції (4).

Приклад 4. Побудувати аналітичну в одиничному колі функцію $f(z)$ по значенню на контурі x^2 її дійсної частини. В інтегралі (4) обрати $v(0)=0$.

Розв'язання. Продовжить роботу з останнім файлом. В ньому вже побудовано одиничне коло і зовнішній тонкий шар, який ми виключатимемо з графіка.

Додайте в модель змінну, які обчислюватиметься за формулою (4), і змінну, яка являє її дійсну частину, тобто буде гармонічною функцією.

Definitions – Variables 1:

Variables – **Name, Expression** =F, -intC1(x^2*(z+dest(z))/(z*(z-dest(z)))*(tx+i*ty))/(2*pi*i)

Мінус поставлено у зв'язку з іншим напрямком обходу кола порівняно з формулою (4), де контур обходиться супротив годинникової стрілки.

Variables – **Name, Expression** = U,real(F)

Variables

Name	Expression
z	x+i*y
fb	(z+conj(z))^2/4+(z-conj(z))/2
f	-intC1(fb/(z-dest(z))*(tx+i*ty))/(2*pi*i)
F	-intC1(x^2*(z+dest(z))/(z*(z-dest(z)))*(tx+i*ty))/(2*pi*i)
U	real(F)

Оновіть обчислення.

Study 1: RMB > Update Solution

Побудуйте графік гармонічної функції $U(z)$ – дійсної частини аналітичної функції $F(z)$, а також криву $u = x^2$ над контуром кола.

2D Plot Group 1: RMB > Duplicate

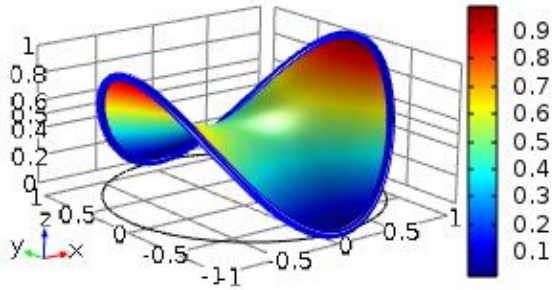
2D Plot Group 4:

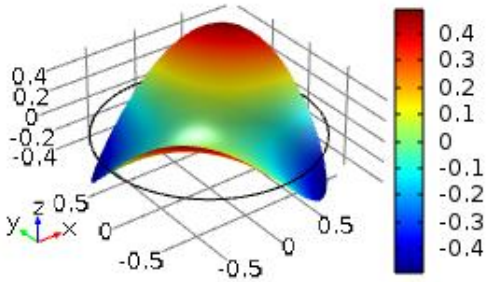
`Color Legend` – **Show maximum and minimum values** = `False`

2D Plot Group 4 – Surface 1:

`Expression` – **Expression** = `U`

LMB>Plot (*наступний рисунок ліворуч*)





Графік спряженої гармонічної функції $v(z)$ побудуйте самостійно (попередній рисунок праворуч). ■

В наступному прикладі буде розглянуто спосіб відтворення аналітичної в півплощині $y \geq 0$ функції $f(z)$ по значенню її дійсної частини $u(x)$ на межі $y = 0$. Відповідна формула має вид [2]

$$f(z) = \frac{1}{\pi i} \int_{-\infty}^{\infty} \frac{u(x)}{x - z} dx + i \cdot C, \tag{5}$$

де C – дійсна стала. Звідси випливає формула $u(x, y) = u(z) = \mathrm{Re}[f(z)]$ відтворення гармонічної в напівплощині $y \geq 0$ функції, яка на дійсній осі має значення $u(x)$.

Приклад 5. Побудувати гармонічну у верхній півплощині $y \geq 0$ функцію, яка на дійсній осі має значення $u(x) = \frac{1}{2}(|x+1| - 2|x| + |x-1|)$.

Розв'язання. Створіть порожню модель і додайте в неї двовимірний компонент.

File-> New -> Blank Model

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть перевірку розмірностей фізичних величин і встановіть прапорець, який включає режим відображення напрямку зростання параметра вздовж кривих.

Untitled.mph (root): `Unit System` – <> = None

Component 1 – Definitions - View 1:

`View` – **Show edge direction arrows** = `True`

Створіть функцію $u(x)$ граничного значення.

Global Definitions: RMB > Functions – Analytic

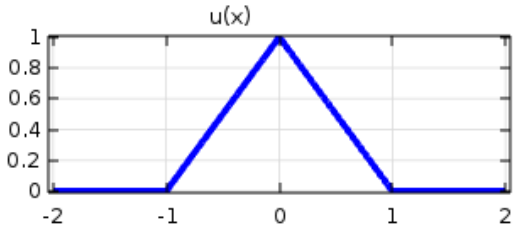
Function name = `u`

`Definition` – **Expression** = `(abs(x+1)-2*abs(x)+abs(x-1))/2`

Для перевірки виконайте побудову графіка цієї функції.

`Plot Parameters` – **Low limit, Upper limit** = `-2, 2`

LMB> Plot



Розрахунковою областю оберемо верхнє півколо достатньо великого радіуса (воно імітуватиме частину верхньої півплощини).

`Geometry` -> `Primitives` – Circle

Geometry 1 - Circle 1 (c1):

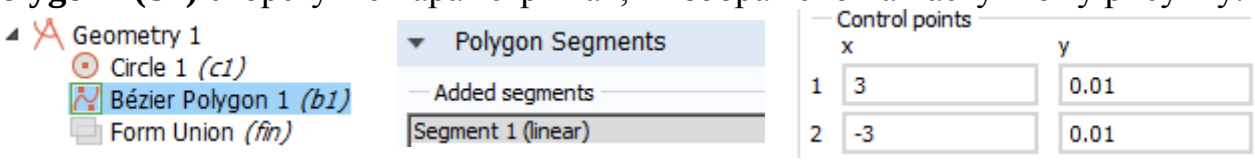
`Size and Shape` – **Radius, Sector angle** = `3, 180`

Формула (5) непридатна для обчислення функції на межі області, і в близьких до неї точках можемо мати суттєву похибку обчислень. При зображенні створюваної функції буде зручно виключити з графіка тонку смугу біля дійсної осі. Щоб це зробити, в геометрії накресліть відрізок поруч з дійсною віссю, наприклад, на відстані 0.01. Він буде межею такої смуги.

При активному вузлі **Geometry 1** виконайте команду

`Geometry` -> `Draw` – Line

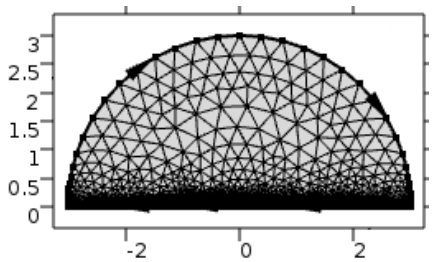
Наведіть курсор миші в графічне вікно, намалюйте відрізок, і в ПВ вузла **Bézier Polygon 1(b1)** скорегуйте параметри так, як зображено на наступному рисунку.



LMB > Build All Objects

На побудованій геометрії генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB> Build All



Зауважимо, що наявність тонкої полоси біля дійсної осі змусила систему побудувати дуже дрібну сітку в смузі і біля неї.

Створіть оператор інтегрування по дійсній осі.

`Definitions` > Component Couplings - Integration

Definitions - Integration 1 (intop1):

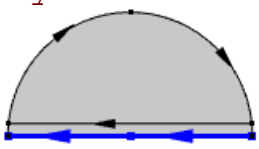
Operator name = `intR`

`Source Selection` – **Geometric entity level** = `Boundary`

– **Selection** = `Manual, 1,4`

(*діаметр напівкола*)

Будьте уважні, не виділіть сусідній відрізок.



Зверніть увагу на напрямок зростання параметра вздовж відрізка інтегрування.

Побудуйте таблицю змінних з ідентифікаторами незалежної змінної z, змінною *F*, яка обчислюватиметься за формулою (5), і зміною *U* для її дійсної частини.

Definitions: RMB > Variables

Definitions – Variables 1:

`Variables` – **Name, Expression** = `z, x+i*y`

`Variables` – **Name, Expression** = `F,intR(u(x)/(x-dest(z)))/(pi*i)`

Наприкінці цього прикладу ми пояснимо, чому тут не використано знак мінус перед інтегралом, незважаючи на протилежний напрямок руху порівняно з формулою (5).

`Variables` – **Name, Expression** = `U,real(F)`

Variables

Name	Expression
z	x+i*y
F	intR(u(x)/(x-dest(z)))/(pi*i)
U	real(F)

Створіть в моделі вузол досліджень і виконайте обчислення.

Untitled.mph (root): RMB > Add Study = `Add Study ->`

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Побудуйте графік дійсної частини змінної `F` у півколі, виключивши з нього тонку смугу біля дійсної осі. Для цього спочатку створіть джерело даних, з геометрії якого виключено цю полосу.

Results – Data Sets - Study 1/Solution 1: RMB > Duplicate

Results – Data Sets - Study 1/Solution 1 (2) : RMB > Selection

Results – Data Sets - Study 1/Solution 1 (2) - Selection :

`Geometric Entity Selection` – **Geometric entity level** = `Domain`

Geometric Entity Selection – **Selection**=Manual,2 (*зона над смугою*)

Тепер створіть і налаштуйте 2D графічну групу.

Results -> **Plot Group** ->2D Plot Group

2D Plot Group 1:

Data – **Data set** = Study 1/Solution 1(2)

2D Plot Group 1-> **Add Plot** -> Surface

2D Plot Group 1 – Surface 1:

Expression – **Expression** = U

RMB> Height Expression

2D Plot Group 1 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = True,1

Для наочності побудуйте графік функції $u(x)$ над дійсною віссю (граничне значення гармонічної функції U). Для цього спочатку створіть джерело даних Edge 2D, а потім на ньому в поточній графічній групі Line графік.

Results – Data Sets : RMB > Edge 2D

Data Sets - Edge 2D 1:

Data – **Data set** = Study 1/Solution 1(1)

Selection – **Selection** = Manual; 1,4 (*діаметр півкола*)

2D Plot Group 1-> **Add Plot** -> Line

2D Plot Group 1 – Line 1:

Data – **Data set** = Edge 2D 1

Expression – **Expression** = 1

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.05

– **Radius scale factor** = True,1

– **Coloring, Color** = Uniform, Black

RMB> Height Expression

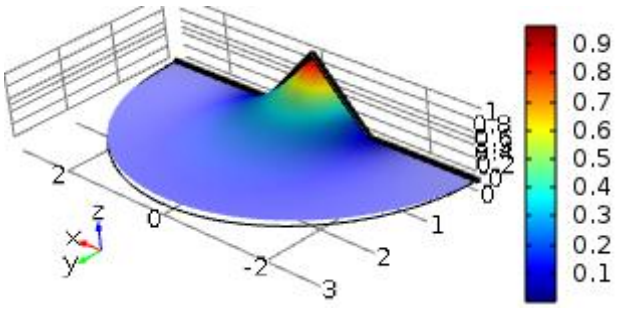
2D Plot Group 1 – Line 1 – Height Expression 1:

Expression – **Height data** = Expression

– **Expression** = u(x)

Axis – **Scale factor** = True,1

LMB>Plot



Пояснимо вибір знака в формулі intR(u(x)/(x-dest(z)))/(pi*i) при обчисленні змінної F. Справа в тому, що Coupling оператор інтегрування по

кривій не використовує напрям її обходу. Наприклад, вираз `intR(1)` поверне довжину відрізка інтегрування, тобто додатне число, незалежно від напрямку вздовж кривої. Якщо підінтегральний вираз скрізь додатний, то і інтеграл буде додатним незалежно від напрямку кривої.

Щоб врахувати напрямок вздовж діаметра в підінтегральний вираз потрібно додати множник `tx`, тобто X компоненту дотичного до кривої вектора (в нашому прикладі вона дорівнює `-1`). Для перевірки обчислимо значення виразів `intR(1)` і `intR(1*tx)`.

`Results` **`-> Derived Values`** – Global Evaluation

Derived Values - Global Evaluation 1:

`Expression` – **Expression** = `intR(1)`

LMB > Evaluate

В таблиці **Table 1** буде надруковано значення `6.0000`. Поміняйте вираз в цьому вузлі і надрукуйте результат в ту же таблицю.

Derived Values - Global Evaluation 1:

`Expression` – **Expression** = `intR(1*tx)`

LMB > Evaluate – Table 1

intR(1)	intR(1*tx)
6.0000	-6.0000

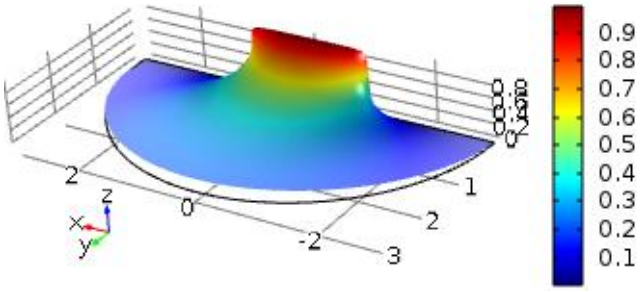
Те ж саме стосується виразу для змінної `F` у вузлі **Variables 1**. Якщо там застосувати формулу `intR(u(x)*(tx+i*ty)/(x-dest(z)))/(pi*i)`, то вона буде враховувати напрямок, і вам потрібно буде поставити мінус перед виразом (змінна `ty` в нашому прикладі дорівнює нулю).

Вправа. Побудувати у верхній півплощині $y \geq 0$ гармонічну функцію, яка на дійсній осі має значення

$$u(x) = \begin{cases} 1 \ , a \leq x \leq b \\ 0 \ , \ \textit{інакше} \end{cases}.$$

Порівняти графіки отриманого розв'язку і аналітичного

$$w(x, y) = \frac{1}{\pi}\left(\operatorname{arctg}\frac{b-x}{y} - \operatorname{arctg}\frac{a-x}{y}\right).$$



$$a = -1, b = 1.$$

■

Якщо аналітична в одиничному колі функція $f(z) = u + i \cdot v$ в якості своєї дійсної частини на межі кола має значення $u(\zeta)$, то вона відновлюється за

формулою (4). Її дійсна частина $u(z)$ є гармонічною функцією з тим же граничним значенням, тобто

$$u(z)=\mathrm{Re}\left[\frac{1}{2\pi i}\int\limits_{|\zeta|=1}\frac{u(\zeta)}{\zeta}\frac{\zeta+z}{\zeta-z}d\zeta\right].$$

Якщо аргументи гармонічної функції $u(z)=u(x,y)$ піддати перетворенню інверсії

$$x=\frac{\xi}{\xi^2+\eta^2},\ y=\frac{\eta}{\xi^2+\eta^2},$$

то результівна функція теж буде гармонічною [6, с. 414]. Це перетворення можна записати у вигляді $z=w/|w|^2=1/\overline{w}$, де $z=x+i\cdot y$ і $w=\xi+i\cdot\eta$. Його можна розглядати як відображення внутрішніх точок кола $|z|<1$ площини z в точки $|w|>1$ зовнішності кола площини w. Оскільки на межі кола $|\omega|=1$, то $\zeta=\omega/|\omega|^2=\omega$, $d\zeta=d\omega$ і $u(\zeta)=u(\omega)$. Тому

$$u\left(w/|w|^2\right)=U(w)=\mathrm{Re}\left[\frac{1}{2\pi i}\int\limits_{|\omega|=1}\frac{u(\omega)}{\omega}\frac{\omega+1/\overline{w}}{\omega-1/\overline{w}}d\omega\right].$$

Тобто в комплексній площині w гармонічна в зовнішності $|w|>1$ одиничного кола функція $U(w)$, яка на його межі має значення $u(\omega)$, обчислюється за наведеною формулою. Переходячи знову до позначення комплексної змінної z, отримуємо

$$U(z)=\mathrm{Re}\left[\frac{1}{2\pi i}\int\limits_{|\zeta|=1}\frac{u(\zeta)}{\zeta}\frac{\zeta+1/\overline{z}}{\zeta-1/\overline{z}}d\zeta\right]. \qquad (6)$$

де інтегрування по колу $|\zeta|=1$ виконується в напрямку проти годинникової стрілки.

Зауваження. Для зовнішніх крайових задач на площині зазвичай потрібно виставляти додаткову умову обмеженості розв'язку на нескінченності (для забезпечення однозначності).

Приклад 6. Побудувати гармонічну функцію $u(x,y)$ в зовнішності $|z|>1$ одиничного кола, якщо на межі кола вона має значення $u|_{\Gamma}=x^2$.

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = None

Додайте в модель двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

У вузлі **View 1** встановіть прапорець, який вмикає відображення напрямків зростання параметрів кривих.

Component 1 – Definitions - View 1:

View – **Show edge direction arrows** = True

Створіть розрахункову геометрію у формі квадрата з круговим отвором, що імітуватиме зовнішність одиничного кола

Geometry->**Primitives->** Square

Geometry 1 - Square 1 (sq1):

Size – **Size length:** = 6

Position – **Base, x:, y:** =Center, 0, 0

Geometry -> **Primitives** – Circle

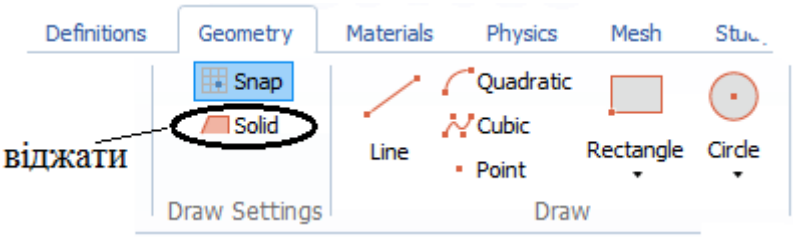
Geometry 1 - Circle 1(c1):

Size and Shape – **Radius:** = 1

Position – **Base, x, y** = Center,0 ,0

Формула (6), як і попередні, «не працює» на контурі інтегрування. Тому при побудові графіків розв'язку буде корисним з розрахункової геометрії виключити тонкий шар навколо кола. Це може бути тонке кільце, наприклад, $1 \le |z| \le 1.01$. Але спочатку для наочності ми виключимо товстіше кільце $1 \le |z| \le 1.1$, а після завершення моделювання і всіх обчислень, зменшимо його зовнішній радіус для отримання остаточного графіка.

Накресліть окружність (не коло). Спочатку вимкніть режим побудови солідних фігур, «віджавши» кнопку Solid на інструментальній стрічці Geometry.



За замовчуванням ця кнопка натиснута, і всі замкнені криві утворюють суцільні фігури, а не криві. Щоб накреслити лише окружність, цю кнопку потрібно «віджати». Тепер натисніть кнопку креслення кола.

Geometry -> **Draw** – Circle (*на останньому рисунку кнопка* Circle)

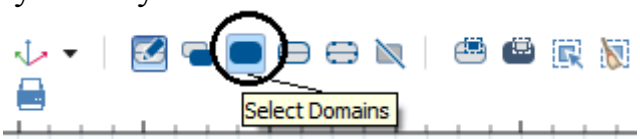
Наведіть курсор миші в графічне вікно і намалюйте окружність, клацнувши мишею в початку координат. Не відпускаючи ліву кнопку, рухайте мишу доки крива не набуде приблизно бажаного розміру. В ПВ створеного вузла **Circle 2 (c2)** скорегуйте параметри окружності (радіус і координати центра). Вони повинні мати наступні значення.

Geometry 1 - Circle 2(c2):

Size and Shape – **Radius:** = 1.1

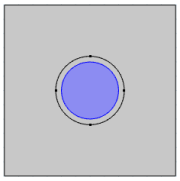
Position – **Base, x, y** = Center,0 ,0

В графічному вікні увімкніть режим Select Domains. Для цього над цим вікном натисніть однойменну кнопку.



Видаліть одиничне коло з розрахункової геометрії.

Geometry **->Booleans and Partitions** – Difference

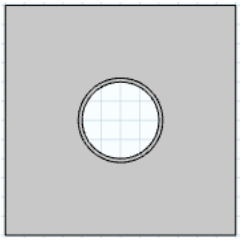


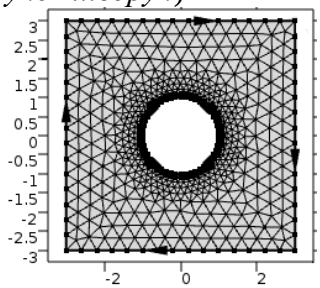
Geometry 1 - Difference 1 (dif1):

Difference – **Object to add** = sq1 (*квадрат*)

Union – **Object to subtract** = c1 (*одиничне коло*)

LMB > Build All Objects (*наступний рисунок ліворуч*)





Генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 : LMB > Build All (*попередній рисунок праворуч*)

В кільці і зоні біля нього буде побудована дрібніша сітка (це одна з причин, по якій це граничне кільце потрібно).

Створіть оператор інтегрування по контуру одиничного кола.

Definitions > Component Couplings - Integration

Definitions - Integration 1 (intop1):

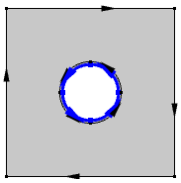
Operator name = intC1

Source Selection – **Geometric entity level** = Boundary

– **Selection** = Manual, 7,8,10,11

(*дуги внутрішнього одиничного кола*)

Будьте уважні, не виділіть дуги більшого кола.



Зверніть увагу на напрямок обходу одиничного кола - проти годинникової стрілки, такий, як потрібно для застосування формули (6).

Задайте в компоненті комплексну змінну z=x+i*y , допоміжні змінні f і F, а також змінну U, яка представлятиме шукану гармонічну функцію.

Definitions **->Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = z, x+i*y

Variables – **Name, Expression** = f, intC1(x^2*(z+dest(z))/(z*(z-dest(z)))*(tx+i*ty))/(2*pi*i)

Variables – **Name, Expression** = F, subst(f,z,1/conj(z))

Variables – **Name, Expression** = U, real(F)

Створіть вузол дослідження і виконайте обчислення.

Untitled.mph (root): RMB > Add Study = Add Study ->

Preset Studies – Stationary -> X

Study 1: LMB > Compute

В ДМ з'явиться джерело даних **Study 1/Solution 1**, але ніякі графіки не будуватимуться.

Графік змінної U доцільно будувати над трохи звуженою геометрією, яка співпадає з усією, за винятком тонкого шару біля внутрішнього кола. Створіть для цього відповідне ДД.

Results: Data Sets - Study 1/Solution 1 : RMB > Duplicate

Data Sets - Study 1/Solution 1 (2) : RMB > Selection

Study 1/Solution 1 (2) - Selection:

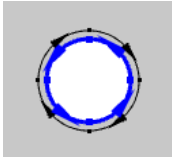
Geometric Entity Selection – **Geometric entity level** = Domain

– **Selection** = 1 (*зовнішність тонкого шару*)

Після створення копії джерела даних ми додали підвузол **Selection**, який для неї (копії) дозволяє обрати рівень геометрії. В нашому випадку ми не включили в ДД кільцеву зону біля одиничного кола.

Додатково ми збираємося побудувати графік граничної функції $u(\zeta)$ над контуром одиничного кола. Для цього нам знадобиться ДД типу Edge 2D.

Results – Data Sets : RMB > Edge 2D



Data Sets - Edge 2D 1:

Data – **Data set** = Study 1/Solution 1(1)

Selection – **Selection** = Manual; 7,8,10,11

Тепер все готово для побудови графіка U.

Results -> **Plot Group** ->2D Plot Group

2D Plot Group 1:

Data – **Data set** = Study 1/Solution 1(2)

2D Plot Group 1-> **Add Plot** -> Surface

2D Plot Group 1 – Surface 1:

Expression – **Expression** = U

RMB > Height Expression

2D Plot Group 1 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = True,2

Для наочності додайте графік граничної функції $u(\zeta) = x^2$, яка визначена лише на контурі одиничного кола.

2D Plot Group 1-> **Add Plot** -> Line

2D Plot Group 1 – Line 1:

Data – **Data set** = Edge 2D 1

Expression – **Expression** = 0

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.05

– **Radius scale factor** = True,1

– **Coloring, Color** = Uniform, Blue

RMB > Height Expression

2D Plot Group 1 – Line 1 – Height Expression 1:

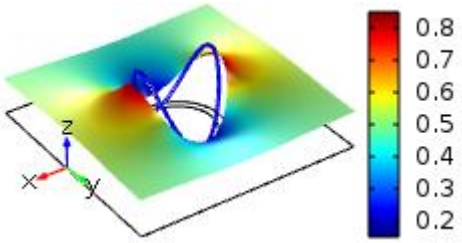
Expression – **Height data** = Expression

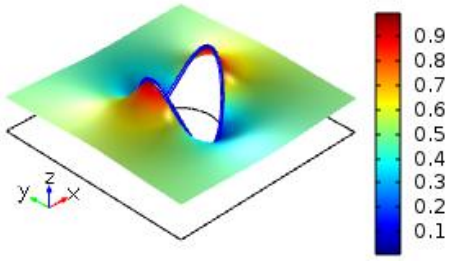
– **Expression** = x^2

Axis – **Scale factor** = True,2

LMB>Plot (*наступний рисунок ліворуч*)

Щілина між кривою і поверхнею пояснюється виключенням з геометрії кільцевого шару.





Виключений з геометрії кільцевий шар зробіть тоншим. Для цього поверніться до вузла **Circle 2** і зменьшіть радіус окружності.

Geometry 1 - Circle 2 (c2):

`Size and Shape` – **Radius:** = `1.01`

Переобчисліть модель.

Study 1: LMB > Compute

Перебудований графік вузла **2D Plot Group 1** зображено на попередньому рисунку праворуч.

Вправа. Для задачі побудови зовні кола радіуса R гармонічної функції по її значенню на контурі існує формула, яка не використовує комплексні змінні [6, с. 419]

$$u(r,\varphi)=\frac{1}{2\pi}\int_0^{2\pi}u(\psi)\frac{r^2-R^2}{r^2-2Rr\cos(\varphi-\psi)+R^2}\,d\psi\,.$$

Застосовуючи її, побудуйте розв'язок розглянутої в прикладі задачі (тобто R=1 і $u\big|_{\Gamma}=x^2=\cos^2\psi$).

Вказівка. Створіть в поточній моделі вузол циліндричної/полярної системи координат, а також змінну W, обчислювану по наведеній формулі. Її (формулу) можна реалізувати одним з наступних виразів:

```
intC1(cos(sys2.phi)^2*(dest(sys2.r)^2-1)/(dest(sys2.r)^2-
2*dest(sys2.r)*cos(dest(sys2.phi)-sys2.phi)+1))/(2*pi)
```

або

```
integrate(cos(phi)^2*(sys2.r^2-1)/
(sys2.r^2-2*sys2.r*cos(sys2.phi-phi)+1)/(2*pi),phi,0,2*pi)
```

8.3. Графічні зображення комплексних функцій.

Будь-яка комплексна функція може розглядатися як перетворення комплексної площині в себе або як відображення однієї комплексної площини в іншу. Наочно зобразити комплексну функцію графіком неможливо, оскільки він вимагає чотирьох вимірів (два на область визначення і ще два для області значень). Якщо замість значення функції розглядати її модуль або аргумент, то отримані рельєфи розміщуються в трьох вимірах і дають деяке уявлення про поведінку функції. Такі приклади ми розглядали в попередніх параграфах.

З іншого боку комплексна функція

$$w = f(z) = f(x + i \cdot y) = u(x, y) + i \cdot v(x, y)$$

може розглядатися як пара дійсних функцій $u = u(x, y)$, $v = v(x, y)$, які реалізують відображення площини (x, y) на площину (u, v). Досить наочним способом представлення таких відображень є побудова образів або прообразів сімейств координатних ліній. Можна побудувати образ прямокутної декартової сітки площини (x, y) на площині (u, v). Часто корисно зрозуміти як виглядають образи кривих полярної сітки площини (x, y) на площині (u, v). Інколи навпаки цікаво аналізувати прообрази декартової або полярної сітки площини (u,v) на площині (x, y). Звичайно, що немає потреби обмежуватися декартовими і полярними сітками, а можна будувати образи або прообрази будь яких координатних сіток або множин кривих.

Поточний параграф присвячений зображенням функцій комплексного змінного, а точніше інструментам COMSOL, які можна застосовувати для ілюстрації відображень, здійснюваним цими функціями.

Приклад 1. Образи кривих при відображеннях, здійснюваних комплексними функціями.

1. Побудувати образи прямих декартової координатної сітки при відображенні $w = z^2$. Побудувати образи прямих $x = 0.2; 0.4; 0.6$ та $y = 0.2; 0.4; 0.6$.
2. При відображенні $w = z^2$ побудувати образ еліпса $x = \cos s$, $y = 0.5 \sin s$.
3. При відображенні $w = z^2$ побудувати образ контура квадрата $|x| \le 1/2, |y| \le 1/2$.
4. Побудувати образи окружностей площини z при відображенні $w = z + 1/z$
5. Побудувати образ окружності $x^2 + y^2 = 2x$ при відображенні $w = 1/z$.

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Додайте в модель двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть в моделі перевірку розмірностей фізичних величин.

Untitled.mph (root): Unit System – <> = None

Розрахункову область візьміть у формі квадрата (він буде зображати частину нескінченної площини $z = x + i \cdot y$).

Geometry->**Primitives->** Square

Geometry 1 - Square 1 (sq1):

Size – **Size length:** = 6

Position – **x:, y:** = -3, -3 (*два поля*)

LMB > Build Selected

Побудуйте в області прямокутну СкЕ сітку.

Mesh **->Generators** – Mapped

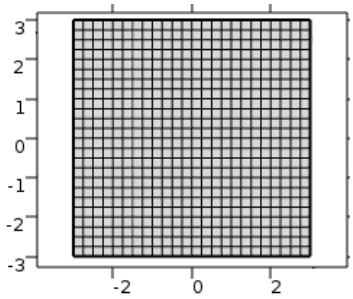
Задайте розміри клітинок сітки.

Component 2 - Mesh 1 – Size:

Element Size – **Custom**= True

Element Size Parameters – **Maximum element size**= 0.1

LMB > Build All



Створіть вузол змінних.

Definitions -> **Variables** – Local Variables

Definitions - Variables 1:

Variables – **Name, Expression =** z,x+i*y (*введення в таблиці*)

– **Name, Expression =** w,z^2 (*введення в таблиці*)

Щоб в наступних графічних вузлах бачити не зображення всієї розрахункової області (тобто квадрата), а лише його центральну частину (нас цікавить зображення сітки, а не квадрата), створіть і налаштуйте вузол View (вид) .

Component 1 – Definitions: RMB > View

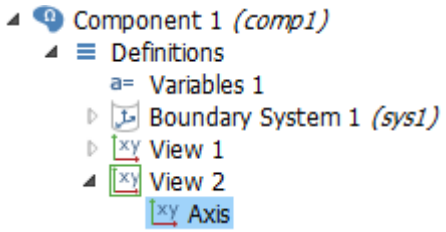
Definitions - View 2:

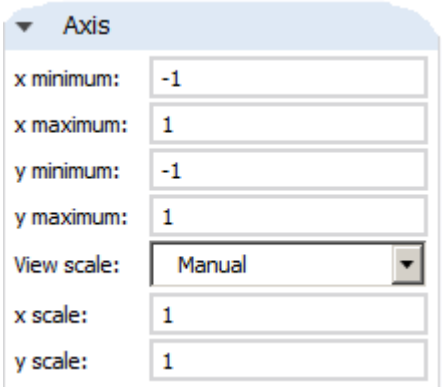
View – **Lock axis =** True

Прапорець **Lock axis** (блокування осей) буде зберігати поточні границі виду після використання інструментів масштабування.

У дочірньому вузлі **Definitions - View 2 – Axis** налаштуйте діапазони виду і їх пропорції так, як зображено на рисунку.

Definitions - View 2 - Axis:





LMB>Update

Щоб змінні моделі працювали, потрібно виконати обчислення, а для цього треба мати вузол досліджень.

Untitled.mph (root): RMB > Add Study = Add Study ->

Preset Studies – Stationary ->X

Study 1: LMB > Compute

1. Побудуйте зображення декартової координатної сітки та образи прямих $x = 0.2; 0.4; 0.6$, $y = 0.2; 0.4; 0.6$.

Results -> **Plot Group** ->2D Plot Group

2D Plot Group 1:

Plot Settings – **View =** View 2

– **Plot data set edges =** False

Color Legend – **Show legends =** False

2D Plot Group 1-> **Add Plot** -> Mesh

2D Plot Group 1 – Mesh 1:

Color – **Color table =** GrayScale

– **Wireframe color =** Black

Зауваження. У графічному вузлі типу Mesh немає поля **Expression** для задання виразу. Цей вузол призначений для зображення СкЕ сіток.

2D Plot Group 1 – Mesh 1: RMB > Deformation

2D Plot Group 1 – Mesh 1 - Deformation 1:

Expression – **x component** = real(w)-x

Expression – **y component** = imag(w)-y

Якщо вузол **Deformation 1** не створювати, то буде зображатися входова декартова сітка. Використання вектора (real(w),imag(w)) у вузлі Deformation визначає зсув точок, а не їх нове положення. Щоб отримати образи точок потрібно було вектор зсуву скорегувати на величину радіуса-вектора (x, y).

Scale– **Scale factor** = True,1

LMB>Plot (*наступний рисунок ліворуч*)

Зауваження. В системі існує ДД типу Mesh, призначене для візуалізації та аналізу СкЕ сітки без застосування обчислень. Але для наших цілей воно непридатне, бо не «містить в собі» змінних моделі.

Існує можливість зафарбувати клітинки координатної сітки в шаховому порядку.

2D Plot Group 1-> **Add Plot** -> Surface

2D Plot Group 1 – Surface 1:

Expression – **Expression** = sin(pi*x/0.1)*sin(pi*y/0.1)>0

Тут дільник 0.1, використаний в формулі, є кроком mapped сітки, заданим у вузлі **Mesh 1 – Size**. А сам вираз є характеристичною функцією множини однокольорових клітинок нескінченної шахівниці.

Coloring and Style – **Coloring, Color** =Color table, GrayScale

Quality – **Resolution** = Extra fine

Задання високої роздільної здатності в останній опції означає, що скінченні елементи розбиватимуться на більш дрібні ділянки під час рендеринга і якість зображення покращиться.

RMB> Deformation

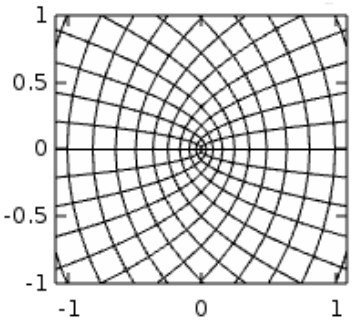
2D Plot Group 2 – Surface 1 – Deformation 1:

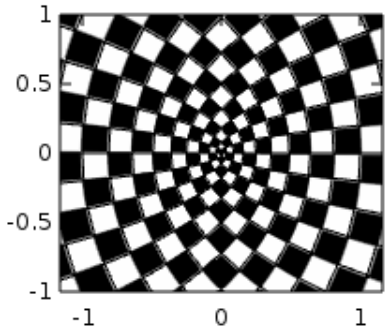
Expression – **x component** = real(w)-x

Expression – **y component** = imag(w)-y

Scale– **Scale factor** = True,1

LMB>Plot (*наступний рисунок праворуч*)





Інший спосіб побудувати ту ж сітку полягає в створенні `Surface` вузла з налаштуванням **Wireframe** = `True`.

`Results` **`-> Plot Group ->`**2D Plot Group

2D Plot Group 2:

`Plot Settings` – **View =** `View 2`

– **Plot data set edges =** `False`

`Color Legend` – **Show legends =** `False`

`2D Plot Group 2->` **`Add Plot ->`** Surface

2D Plot Group 2 – Surface 1:

`Expression` – **Expression** = `0`

`Coloring and Style` – **Coloring, Color** = `Uniform, Black`

– **Wireframe** = `True`

RMB> Deformation

2D Plot Group 2 – Surface 1 – Deformation 1:

`Expression` – **x component** = `real(w)-x`

`Expression` – **y component** = `imag(w)-y`

`Scale`– **Scale factor** = `True,1`

LMB>Plot

Буде побудовано те ж саме зображення, що раніше ліворуч.

Додайте на передостанній рисунок образи прямих $x = 0.2; 0.4; 0.6$ і $y = 0.2; 0.4; 0.6$. Для цього створіть два джерела даних типу `Cut Line 2D`.

`Results ->` Data Set - Cut Line 2D

Data Sets - Cut Line 2D 1:

`Line Data` – **Line entry method** = `Two points`

– **Point 1: x:** = `0.2` **y:**=`-3`

– **Point 2: x:** = `0.2` **y:**= `3`

– **Additional parallel lines** = `True`

– **Distances** = `-0.2, -0.4`

LMB > Plot (*наступний рисунок ліворуч*)

Data Sets -Cut Line 2D 1: RMB > Duplicate

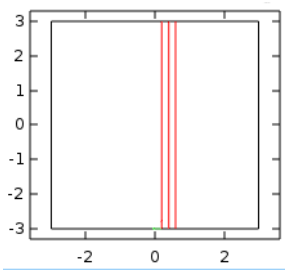
Data Sets -Cut Line 2D 2:

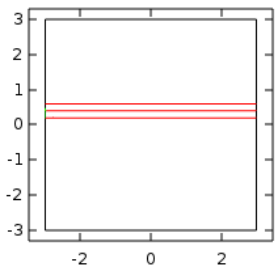
– **Point 1: x:** = `-3` **y:**=`0.2`

– **Point 2: x:** = `3` **y:**=`0.2`

– **Distances** = `0.2, 0.4`

LMB > Plot (*наступний рисунок праворуч*)





Для зображення образів цих прямих додайте в гілку **2D Plot Group 2** два `Line` вузли.

2D Plot Group 2: RMB > Line

2D Plot Group 2 – Line 1:

`Data` – **Data set** = `Cut Line 2D 1`

`Expression` – **Expression** = `0`

`Coloring and Style` – **Line type** = `Tube`

– **Tube radius expression** = `0.01`

– **Radius scale factor** = `True,1`

– **Coloring, Color** = `Uniform, Red`

2D Plot Group 2 – Surface 1 - Deformation 1: RMB > Copy

2D Plot Group 2 – Line 1: RMB > Paste Deformation

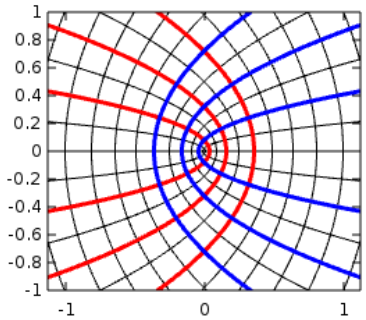
2D Plot Group 2 – Line 1: RMB > Duplicate

2D Plot Group 2 – Line 2:

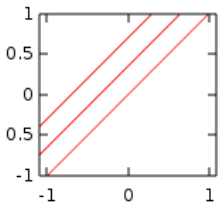
`Data` – **Data set** = `Cut Line 2D 2`

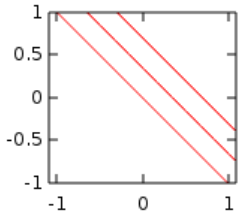
`Coloring and Style` – **Coloring, Color** = `Uniform, Blue`

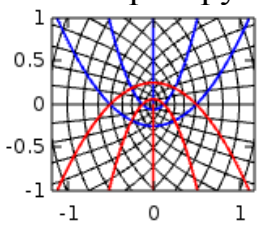
LMB>Plot



Вправа. При відображенні $w = z^2$, побудувати образи прямих, показаних на наступному рисунку ліворуч і посередині. Відповідь зображено праворуч.







Вказівка. Для входових прямих побудувати джерела даних типу `Cut Line 2D`.

2. Побудуйте образ еліпса з параметричним рівнянням $x = \cos s$, $y = 0.5 \sin s$.

Якщо потрібно будувати образ параметричної кривої (а не прямої), то слід використати ДД `Parameterized Curve 2D`.

`Results` -> Data Set – More Data Sets - Parameterized Curve 2D

Data Sets - Parameterized Curve 2D 1:

Parameter – **Maximum** = `2*pi`

Expression– **x:, y:**= `cos(s), sin(s)/2`

LMB > Plot (*наступний рисунок ліворуч*)

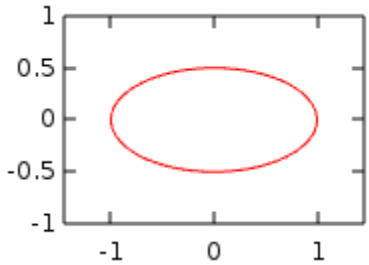
2D Plot Group 2 – Line 1: RMB > Duplicate

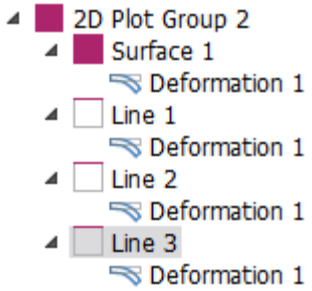
2D Plot Group 2 – Line 3:

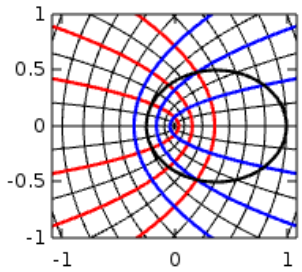
Data – **Data set** = `Parameterized Curve 2D 1`

Coloring and Style – **Coloring, Color** = `Uniform, Black`

LMB > Plot (*наступний рисунок праворуч*)



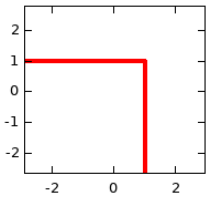


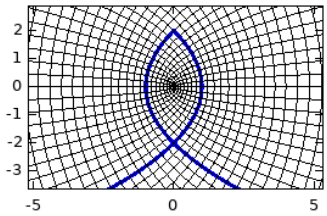


Зауваження 1. При відображенні $w = z^2$ для $z = r\,e^{i\varphi}$ маємо $w = r^2 e^{i2\varphi}$. Тобто коло радіуса r переходить в коло радіуса r^2, а його точка з аргументом α - в точку з аргументом 2α. Коли α зростає від 0 до 2π аргумент образу змінюється від 0 до 4π, тобто контур образу пробігається двічі. Більш того, образом замкнутої навколо точки $z = 0 + i \cdot 0$ кривої при відображенні $w = z^2$ буде замкнутий контур, який оминається двічі. Отже результівна крива, яка зображена на попередньому рисунку праворуч, точкою w обходиться двічі, коли точка z пробігає еліпс.

Зауваження 2. Комплексне відображення $w = f(z)$ можна представити у дійсному вигляді $u = u(x, y)$, $v = v(x, y)$. Тоді образ прямих декартової сітки $x = Const$ або $y = Const$ можна реалізувати у вигляді параметричних кривих $u = u(C_x, y)$, $v = v(C_x, y)$ або $u = u(x, C_y)$, $v = v(x, C_y)$ при різних фіксованих значеннях сталих C_x, C_y. Більш того, якщо крива на площині (x, y) задана параметрично $x = x(\tau)$, $y = y(\tau)$, то її образ теж буде параметричною кривою $u = u(x(\tau), y(\tau)) = \tilde{u}(\tau)$, $v = v(x(\tau), y(\tau)) = \tilde{v}(\tau)$.

Вправа. На площині (x, y) задана параметрична крива з рівнянням $x = 1 + (\tau - |\tau|)/2$, $y = 1 - (\tau + |\tau|)/2$ (ламана на наступному рисунку ліворуч). Використовуючи попереднє зауваження, побудувати її образ при відображенні $w = z^2$. Відповідь зображено праворуч.





3. Побудуйте образ контура квадрата $|x|\le 1/2, |y|\le 1/2$. Його можна визначити неявним рівнянням $1-|x|-|y|-||x|-|y||=0$.

Коли потрібно будувати образ кривої, яка задана неявно, то слід використовувати ДД типу `Contour`. Намалюйте образ квадрата поверх декартової координатної сітки.

Спочатку побудуйте «контурне» джерело даних.

`Results` -> Data Set – More Data Sets - Contour

Data Sets - Contour 1:

`Expression`– Expression = `1-abs(x)-abs(y)-abs(abs(x)-abs(y))`

`Levels` – Entry method= `Levels`

`Levels` – Levels = `0`

LMB>Plot (*наступний рисунок ліворуч*)

Тепер побудуйте образ квадрата.

2D Plot Group 2: RMB > Duplicate

2D Plot Group 3 – Line 1:

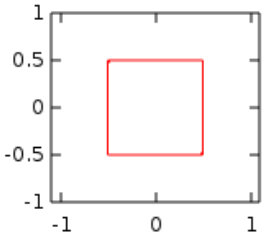
`Data` – Data set = `Contour 1`

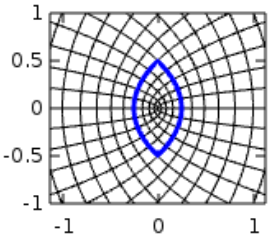
`Coloring and Style` – Coloring, Color = `Uniform, Blue`

2D Plot Group 3 – Line 2: RMB > Delete

2D Plot Group 3 – Line 3: RMB > Delete

LMB > Plot (*наступний рисунок праворуч*)

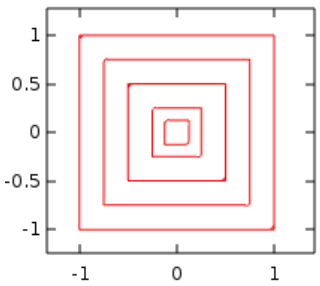


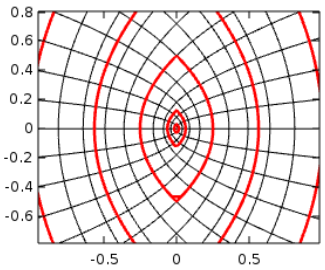


Нагадуємо, що точка `w` проходить побудовану криву двічі, коли точка `z` пробігає контур квадрата один раз.

Вправа. Побудуйте образи квадратів $1-|x|-|y|-||x|-|y||=-1;-0.5;0;0.5;0.75$.

Вказівка. В `Contour` джерелах даних можна задавати список значень рівня. Наприклад, в ДД **Contour 1** в полі **Levels** ви можете задати значення `-1,-1/2, 0,1/2,3/4`.





4. Побудуйте образи окружностей при відображенні $w=z+1/z$.

Створіть «контурне» ДД, яке складатиметься з кількох окружностей.

`Results` -> Data Set – More Data Sets - Contour

Data Sets - Contour 2:

`Expression`– Expression = `sqrt(x^2+y^2)`

Levels – **Entry method**= Levels

Levels – **Levels** = range(1,0.5,3)

LMB>Plot (*наступний рисунок ліворуч, після масштабування*)

Тепер побудуйте їх образи.

Results -> **Plot Group** ->2D Plot Group

2D Plot Group 4:

Plot Settings – **View =** View 1

– **Plot data set edges =** False

Color Legend – **Show legends =** False

2D Plot Group 3 – Line 1: RMB > Copy

2D Plot Group 4: RMB > Past Line

2D Plot Group 4 – Line 1:

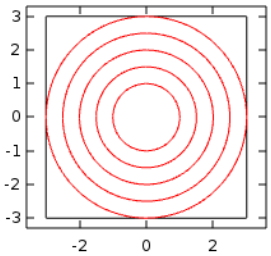
Data – **Data set** = Contour 2

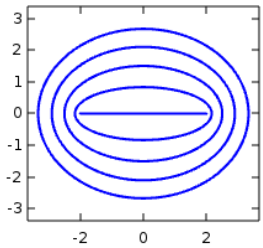
2D Plot Group 4 – Line 1 - Deformation 1:

Expression – **x component** = real(z+1/z)-x

– **y component** = imag(z+1/z)-y

LMB > Plot (*наступний рисунок праворуч*)





Окружності в площині z зручніше будувати, використовуючи полярні координати. Але для цього в моделі потрібно мати цю систему координат. Створіть її.

Definitions -> **Coordinate Systems** – Cylindrical System

У вузлі **Definitions** створюється циліндрична система координат (вузол **Cylindrical System 2 (sys2)**), позначення і розташування якої наводяться в ПВ цього вузла.

Coordinate names

First (r)	Second (phi)	Third (a)
r	phi	a

Origin

x	y
0	0

Щоб змінні системи координат стали доступними, оновіть обчислення.

Study 1: RMB > Update Solution

Створені ідентифікатори (додаючи префікс sys2) можна використовувати в моделі одночасно з декартовими координатами. Наприклад, sys2.r позначатиме полярний радіус, і цю змінну можна застосувати при побудові «контурного» ДД.

Results -> Data Set – More Data Sets - Contour

Data Sets - Contour 3:

Expression– **Expression** = sys2.r

Levels – **Entry method**= `Levels`

Levels – **Levels** = `range(1,0.25,3)`

LMB>Plot (*наступний рисунок ліворуч, після масштабування*)

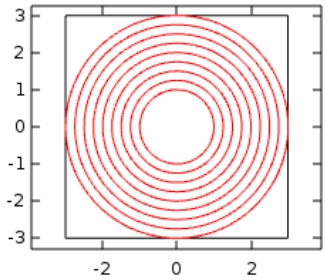
Тепер побудуйте образи кривих (окружностей) створеного ДД.

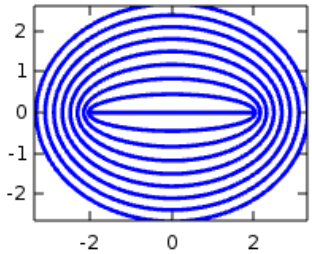
2D Plot Group 4: RMB > Duplicate

2D Plot Group 5 – Line 1:

Data – **Data set** = `Contour 3`

LMB > Plot (*наступний рисунок праворуч*)





Зауваження. Функція $W = z + 1/z$ здійснює однозначне відображення області $|z|>1$ (або області $|z|<1$) площини z на зовнішність відрізка $[-1,1]$. Тому для ДД **Contour 3** ми обрали кола радіуса більшого за одиницю.

Зауваження. Необов'язково будувати образи декартової або полярної сіток. Можна накреслити образи будь-якої системи контурів площині (x, y). Для цього потрібно намалювати цю систему контурів, а потім створити вузол деформації з вектором зсуву $\Delta x = u(x, y) - x$, $\Delta y = v(x, y) - y$.

5. Побудувати образ окружності $x^2 + y^2 = 2x$ при відображенні $w = 1/z$.

Спочатку виразимо рівняння кола через змінну z. Для цього виконаємо заміни $x = (z + \bar{z})/2$, $y = (z - \bar{z})/2i$. Це можна зробити вручну, але для навчальних цілей застосуємо оператор підстановки `subs(expr,var1,expr1,...)`. Його першим аргументом є вираз `expr`, а потім йдуть пари `vari,expri`. Перший елемент в парі – ім'я змінної, а другий – вираз, яким ви бажаєте замінити цю змінну.

Definitions - Variables 1:

Variables – **Name, Expression =** `Eq,X^2+Y^2-2*X`

Variables – **Name, Expression =** `EqZ,`
`subst(Eq,X,(z+conj(z))/2,Y, (z-conj(z))/(2*i))`

Variables – **Name, Expression =** `EqW, subst(EqZ,z,1/z)`

Остання змінна `EqW` представлятиме рівняння образа кола після заміни $w \to 1/z$

Variables

Name	Expression
z	x+i*y
w	z^2
Eq	X^2+Y^2-2*X
EqZ	subst(Eq,X,(z+conj(z))/2,Y, (z-conj(z))/(2*i))
EqW	subst(EqZ,z,1/z)

Для компіляції змінних оновіть обчислення.

Study 1: RMB > Update Solution

Входову криву (коло) і її образ намалюємо в спільній площині z.

`Results` **-> Plot Group ->**2D Plot Group

2D Plot Group 6:

Plot Settings – **View =** `View 1`

Color Legend – **Show legends =** `False`

`2D Plot Group 6`-> **Add Plot ->** Contour

2D Plot Group 6 – Contour 1: (*входове коло*)

Expression – **Expression** = `x^2+y^2-2*x`

Levels – **Entry method**= `Levels`

– **Levels** = `0`

Coloring and Style – **Contour type** = `Tube`

– **Tube radius expression** = `0.05`

– **Radius scale factor** = `True,1`

– **Coloring, Color** = `Uniform, Red`

2D Plot Group 6 – Contour 1: RMB > Duplicate

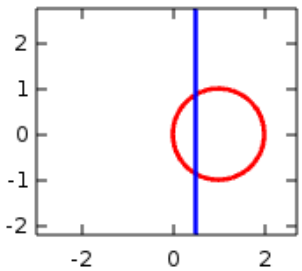
2D Plot Group 6 – Contour 2: (*образ кола*)

Expression – **Expression** = `EqW`

– **Coloring, Color** = `Uniform, Blue`

Зверніть увагу на те, що тут вузол деформації не знадобився тому, що змінна `EqW` уже являє вираз, отриманий з рівняння кола застосуванням зворотного відображення $z = 1/w$ (заміною $z \to 1/z$).

LMB>Plot



На останньому рисунку входова крива (коло) намальована червоним кольором, а її образ (пряма) – синім. На рисунку обидві криві показані в спільній області, і перетворення $w = 1/z$ можна інтерпретувати як відображення площини z в себе.

Приклад 2. Ілюстрація відображень, які здійснюються комплексними функціями.

1. Побудувати образ області $D := \{1/2 \le |z| \le 2,\ \pi/6 \le \arg z \le \pi/2\}$ при відображенні $w = 1 + 1/z$.
2. При відображенні $w = z + 1/z$ побудувати прообрази (на площині z) ліній декартової сітки площини $w = u + i \cdot v$.

3. В площині z побудувати прообраз сітки полярних координат на площині w при відображенні $w = e^{i\alpha}\dfrac{z-a}{1-\bar{a}z}$, де α - дійсне число, a – комплексне число, $|a|<1$.

Розв'язання. Створіть порожню модель, таку ж саму, як в попередньому прикладі, з єдиною відміною – СкЕ сітка буде трикутною (а не mapped).

File -> New -> Blank Model

Untitled.mph (root): Unit System – <> = None

Untitled.mph (root) : RMB > Add Component ->2D

Розрахункову область візьміть у формі квадрата, який буде зображати частину нескінченної площини $z = x + i\cdot y$.

Geometry->**Primitives->** Square

Geometry 1 - Square 1 (sq1):

Size – **Size length:** = 6

Position – **x:, y:** = -3, -3 (*два поля*)

LMB > Build Selected

Побудуйте СкЕ сітку.

Mesh 1: Mesh Settings –**Element size** = Finer

LMB > Build Al

Створіть вузол Variables, в якому задайте комплексну змінну z=x+i*y.

Definitions **->Variables** – Local Variables

Definitions - Variables 1:

Variables – **Name, Expression =** z,x+i*y (*введення в таблиці*)

Створіть вузол досліджень

Untitled.mph (root): RMB > Add Study = Add Study ->

Preset Studies – Stationary ->X

1. Побудуйте образ G області $D := \left\{1/2 \le |z| \le 2, \dfrac{\pi}{6} \le \arg z \le \dfrac{\pi}{2}\right\}$ при відображенні $w = 1 + 1/z$. Очевидно, що зворотне відображення має вигляд $z = 1/(w-1)$.

Характеристичну функцію $\chi_D(z)$ області D можна записати у вигляді

$$\chi_D(z) = (1/2 \le |z|)\&\&(|z| \le 2)\&\&(\pi/6 \le \arg(z))\&\&(\arg(z) \le \pi/2).$$

Характеристична функція $\chi_G(w)$ образа G матиме вид $\chi_G(w) = \chi_D(z(w))$. Отже, якщо у виразі $\chi_D(z)$ виконати заміну $z \to 1/(z-1)$, то це буде характеристичною функцією області G – образа D при зазначеному відображенні площини z на себе.

Створіть вирази $\chi_D(z)$ і $\chi_G(w)$ як змінні моделі.

Definitions - Variables 1:

Variables – **Name, Expression =** XD, 1/2<=abs(z) &&
abs(z)<=2 && arg(z)>=pi/6 && arg(z)<=pi/2

Variables – **Name, Expression =** XG, subst(XD,z,1/(z-1))

Для компіляції змінних виконайте обчислення.

Study 1: LMB > Compute

Тепер побудуйте зображення областей *D* і *G*. Спочатку, використовуючи характеристичну функцію $\chi_D(z)$, побудуйте зображення входової області *D*.

`Results` **-> Plot Group ->** 2D Plot Group

2D Plot Group 1:

Plot Settings – **Plot data set edges =** `False`

Color Legend – **Show legends =** `False`

`2D Plot Group 1`-> **Add Plot ->** Surface

2D Plot Group 1 – Surface 1:

Expression – **Expression** = `XD`

Coloring and Style – **Color table =** `GrayScale`

– **Reverse color table =** `True`

Quality – **Resolution** = `Extra fine`

Остання опція потрібна для покращання якості зображення.

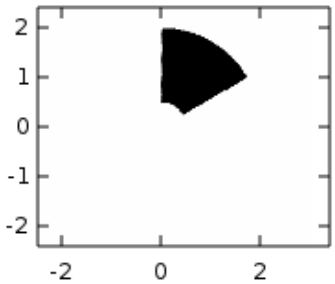
LMB>Plot (*наступний рисунок ліворуч*)

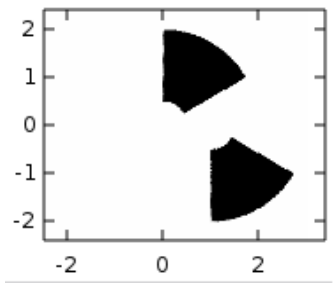
Якщо зараз побудувати зображення області *G*, тобто `Surface` графік характеристичної функції $\chi_D(1/(z-1))$, то він закриє собою попередній, і ми бачитемо лише область *G*. Щоб одночасно побачити обидві області можна побудувати графік характеристичної функції $\chi_{D\cup G}(z)$ об'єднання областей *D* і *G*. Очевидно, вона має вигляд $\chi_{D\cup G}(z) = \chi_D(z) \cup \chi_G(z)$.

2D Plot Group 1 – Surface 1:

Expression – **Expression** = `XD || XG`

LMB>Plot (*наступний рисунок праворуч*)





З останнього рисунка незрозуміло, яка область входова, а яка є образом. Для розрізнення цих зон (і тренування читача) побудуємо червоний контур навколо області *D*, і синій – навколо *G*. Для цього нам знадобляться неявні рівняння контурів цих областей. Це є нестандартною задачею, але потрібний математичний апарат ми вже застосовували в п. 2.2.2. Перечитайте його, якщо отримання наступних формул вам буде незрозумілим.

Додайте в модель нові змінні.

Definitions - Variables 1:

Variables – **Name, Expression =** `IR, (X+Y-abs(X-Y))/2`

Variables – **Name, Expression =** `w1,`
`subst(IR,X,abs(z)-1/2,Y,2-abs(z))`

Variables – **Name, Expression =** `w2,`
`subst(IR,X,arg(z)-pi/6,Y,pi/2-arg(z))`

Variables – **Name, Expression =** `wz, subst(IR,X,w1,Y,w2)`

Тут $wz = wz(z) = wz(x, y)$ є ідентифікуючою функцією області D (вона додатна всередині D, нульова на межі, від'ємна зовні D). Тоді неявне рівняння контура D матиме вигляд $wz(x, y) = 0$.

Variables – **Name, Expression =** `ww, subst(wz,z,1/(z-1))`

Тут $ww = ww(z)$ є ідентифікуючою функцією області G, неявне рівняння контура якої матиме вигляд $ww(x, y) = 0$.

Для компіляції змінних оновіть обчислення.

Study 1: RMB > Update Solution

Тепер побудуйте зображення контурів областей D і G.

`2D Plot Group 1`-> **Add Plot** -> Contour

2D Plot Group 1 – Contour 1: (*контур області D*)

Expression – **Expression** = `wz`

Levels – **Entry method, Levels** = `Levels, 0`

Coloring and Style – **Contour type** = `Tube`

– **Tube radius expression** = `0.03`

– **Radius scale factor** = `True,1`

– **Coloring, Color** = `Uniform, Red`

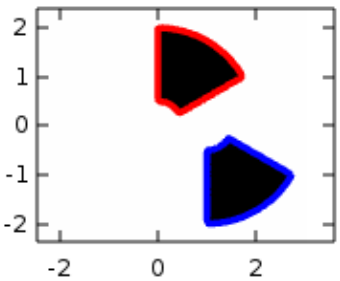
2D Plot Group 1 – Contour 1: RMB > Duplicate

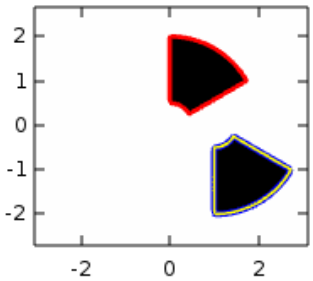
2D Plot Group 1 – Contour 2: (*контур області G*)

Expression – **Expression** = `ww`

Coloring and Style – **Coloring, Color** = `Uniform, Blue`

LMB>Plot (*наступний рисунок ліворуч*)





Контур G побудований нами за неявним рівнянням $ww(z) = 0$, отриманим з використанням зворотного відображення $z = 1/(w-1)$, яке не завжди можна просто визначити. Існує інший спосіб. Він використовує ДД типу `Contour`, яке містить точки входового контуру, а в графічному вузлі на нього «накладається» деформація, утворювана відображенням (як це робилося в попередньому прикладі).

Створіть «контурне» ДД.

`Results` -> Data Set – More Data Sets – Contour

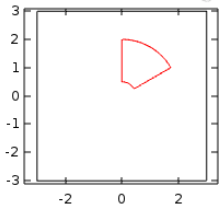
Data Sets - Contour 1: (*ДД входового контуру*)

Expression– **Expression** = `wz`

Levels – **Entry method**= `Levels`

– **Levels** = `0`

LMB>Plot



Використовуючи створене ДД і відповідну деформацію, побудуйте ще один контур області G в тій же графічній групі (для відмінності жовтий).

2D Plot Group 1: RMB > Line

2D Plot Group 1 – Line 1:

Data – **Data set** = Contour 1

Expression – **Expression** = 1

Coloring and Style – **Line type** = Line

– **Coloring, Color** = Uniform, Yellow

Quality – **Resolution** = Extra fine

2D Plot Group 1 – Line 1: RMB > Deformation

2D Plot Group 1 – Line 1 - Deformation 1:

Expression – **x component** = real(1+1/z)-x

– **y component** = imag(1+1/z)-y

Scale– **Scale factor** = True,1

LMB>Plot (*перед попереднім рисунком праворуч*)

Зауваження 1. Контурне ДД **Contour 1** являє входову криву (контур області D), а його зображення з деформацією – контур образа G.

Маючи рівняння контура, області можна малювати, застосовуючи опцію **Contour type** = Filled. Створена функція-ідентифікатор wz області D обертається на нуль тільки на її межі. Щоб зафарбувати область D (а потім її образ G) в ПВ вузла **Contour** в полі **Levels** треба двічі вказати рівень 0.

Results -> **Plot Group** -> 2D Plot Group

2D Plot Group 2:

Plot Settings – **Plot data set edges =** False

Color Legend – **Show legends =** False

2D Plot Group 2-> **Add Plot** -> Contour

2D Plot Group 2 – Contour 1: (*область D*)

Expression – **Expression** = wz

Levels – **Entry method; Levels** = Levels;0,0

Щоб зафарбувати область всередині зони, де $wz > 0$, потрібно двічі вказати нульовий рівень в полі **Levels**. Це означатиме фарбування від рівня 0 до рівня 0.

Coloring and Style – **Contour type** = Filled

– **Fill surfaces outside of contour levels** = False

– **Coloring, Color** = Uniform, Red

2D Plot Group 2 – Contour 1: RMB > Duplicate

2D Plot Group 2 – Contour 2: (*область G*)

Expression – **Expression** = wz

Coloring and Style – **Coloring, Color** = Uniform, Blue

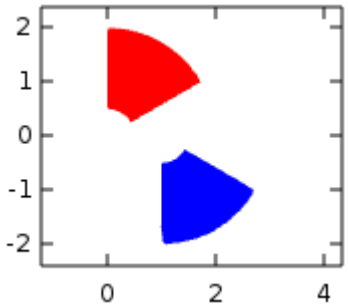
Quality – **Resolution** = Extra fine

2D Plot Group 1 – Line 1 – Deformation 1: RMB > Copy

2D Plot Group 2 – Contour 2: RMB > Paste Deformation

Копіювання вузла **Deformation 1** застосовує до області D перетворення $w = 1 + 1/z$ і, як наслідок, будується область G.

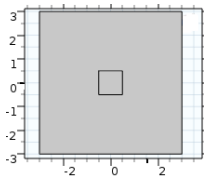
LMB>Plot



Якщо в геометрії компонента область D вже існує, то для зображення її внутрішністі або контура можна використати схожі способи. Ви можете створити ДД, яке співпадатиме з входовою множиною точок, а для зображення образу застосувати графічні вузли, що спираються на це ДД, і в них додати вузли «деформації». Продемонструємо цей спосіб на квадраті, образ якого ми будували в попередньому прикладі. Побудуємо одночасно образи області та контура квадрата при відображенні $w = z^2$.

В гілку геометрії поточного компонента додайте другий квадрат.

Geometry->**Primitives->** Square



Geometry 1 - Square 2 (sq2):

Size – **Size length:** = 1

Position – **x:, y:** = -0.5, -0.5 (*два поля*)

LMB > Build Selected

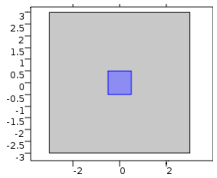
Оновіть дослідження.

Study 1: LMB > Compute

Створіть джерела даних для області і контура квадрата.

Results - Data Sets - Study 1/Solution 1 : RMB > Duplicate

Data Sets - Study 1/Solution 1 (2) : RMB > Selection



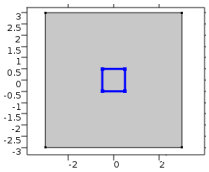
Data Sets - Study 1/Solution 1 (2) - Selection:

Geometric Entity Selection –

Geometric entity level = Domain

– **Selection** = 2 (*центральний квадрат*)

Results – Data Sets : RMB > Edge 2D



Data Sets - Edge 2D 1:

Data – **Data set** = Study 1/Solution 1(1)

Selection – **Selection** = Manual; 4,5,6,7

Тепер побудуйте образи зони і контура квадрата.

Results **-> Plot Group ->** 2D Plot Group

2D Plot Group 3:

Plot Settings – **Plot data set edges =** False

Color Legend – **Show legends =** False

2D Plot Group 3-> **Add Plot ->** Surface

2D Plot Group 3 – Surface 1:

Data – **Data set** = Study 1/Solution 1(2)

Expression – **Expression** = 1

Coloring and Style – **Coloring, Color** = Uniform, Gray

2D Plot Group 3 – Surface 1: RMB > Deformation

2D Plot Group 3 – Surface 1 - Deformation 1:

Expression – **x component** = real(z^2)-x

– **y component** = imag(z^2)-y

Scale– **Scale factor** = True,1

2D Plot Group 3: RMB > Line

2D Plot Group 3 – Line 1:

Data – **Data set** = Edge 2D 1

Expression – **Expression** = 1

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.03

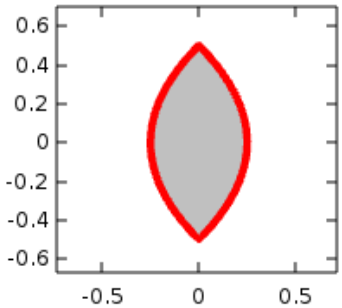
– **Radius scale factor** = True,1

– **Coloring, Color** = Uniform, Red

2D Plot Group 3 – Surface 1 - Deformation 1: RMB > Copy

2D Plot Group 3 – Line 1: RMB > Paste Deformation

LMB > Plot



2. Побудуйте прообраз (на площині z) декартової сітки площини $w = u + i \cdot v$ при відображенні $w = z + 1/z$.

Оскільки $u = \mathrm{Re}\, w = \mathrm{Re}(z + 1/z)$, то прообразами ліній $u = Const$ в комплексній площині z будуть лінії $\mathrm{Re}(z + 1/z) = Const$. Аналогічно, прообразами ліній $v = Const$ будуть криві $\mathrm{Im}(z + 1/z) = Const$. Для їх побудови створіть ще одну графічну групу з двома контурними графіками для цих сімей ліній.

Results -> **Plot Group** -> 2D Plot Group

2D Plot Group 4:

Plot Settings – **Plot data set edges =** False

Color Legend – **Show legends =** False

2D Plot Group 4 -> **Add Plot** -> Contour

2D Plot Group 4 – Contour 1:

Expression – **Expression** = real(z+1/z)

Levels – **Entry method**= Levels

– **Levels** = -10,-8,-6,-4,range(-3,0.25,3),4,6,8,10

Coloring and Style – **Coloring, Color** = Uniform, Black

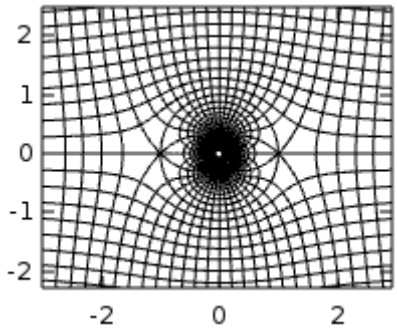
2D Plot Group 4 – Contour 1: RMB > Duplicate

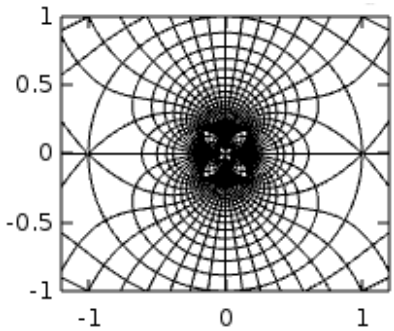
2D Plot Group 4 – Contour 2:

Expression – **Expression** = `imag(z+1/z)`

LMB>Plot (*наступний рисунок ліворуч*)

Щоб роздивитися внутрішню частину рисунка, збільшить масштаб (наступний рисунок праворуч).





Зауваження. Інший спосіб побудови цього графіка полягає в створенні двох окремих «контурних» джерел даних $\mathrm{Re}(z+1/z)=Const$ і $\mathrm{Im}(z+1/z)=Const$, з подальшою побудовою на них графічних вузлів типу `Line`.

Схожий спосіб полягає в створенні криволінійної системи координат (u,v), пов'язаної з декартовими координатами формулами перетворення $u=\mathrm{Re}[z+1/z]$, $v=\mathrm{Im}[z+1/z]$, з подальшою побудовою ліній рівня $u=Const$, $v=Const$.

`Definitions` -> **`Coordinate Systems`** – Mapped System

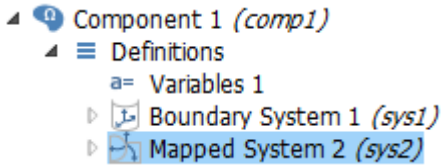
У вузлі **Definitions** створюється криволінійна система координат (вузол **Mapped System 2 (sys2)**), позначення і формули перетворення якої наводяться в ПВ її вузла.

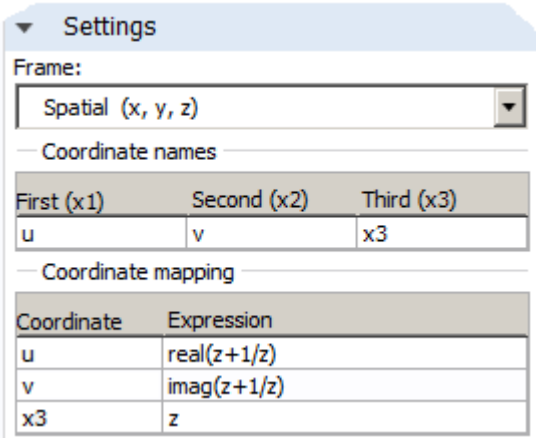
Definitions – Mapped System 2 (sys2):

`Settings` – **Coordinate names – First, Second** = `u,v` (*введення в таблиці*)

– **Coordinate mapping – Expression** = `real(z+1/z)` (*перша змінна*)

– **Coordinate mapping – Expression** = `imag(z+1/z)` (*друга змінна*)





Створені ідентифікатори (додаючи префікс `sys2`) можна використовувати в моделі одночасно з декартовими координатами. Наприклад, змінні (u,v) можна застосувати при побудові координатних ліній.

Щоб змінні (u,v) нової системи координат були доступними, оновіть обчислення.

Study 1: RMB > Update Solution

Тепер побудуйте прообрази ліній $u = Const$, $v = Const$.

`Results` **-> `Plot Group`** -> 2D Plot Group

2D Plot Group 5:

`Plot Settings` – **Plot data set edges =** `False`

`Color Legend` – **Show legends =** `False`

2D Plot Group 4 – Contour 1: RMB > Copy

2D Plot Group 5: RMB > Paste Contour

2D Plot Group 5 – Contour 1:

`Expression` – **Expression** = `sys2.u`

`Coloring and Style` – **Coloring, Color** = `Uniform, Magenta`

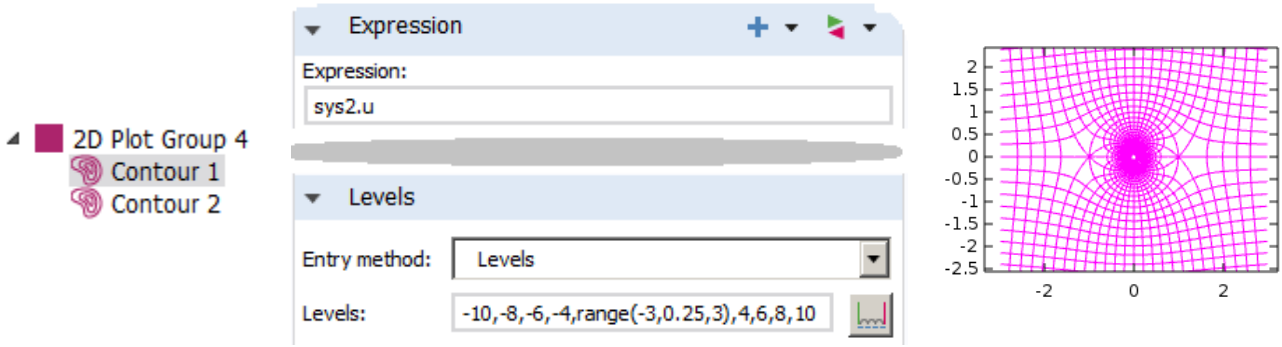
2D Plot Group 5 – Contour 1: RMB > Duplicate

2D Plot Group 5 – Contour 2:

`Expression` – **Expression** = `sys2.v`

LMB>Plot (*наступний рисунок праворуч*)

Буде побудований той же графік, що раніше.



3. Побудуйте прообраз сітки полярних координат на площині $w = e^{i\alpha}\dfrac{z-a}{1-\bar{a}z}$, де α - дійсне число, a – комплексне число, $|a|<1$.

При розв'язанні поточної задачі буде простішим застосувати криволінійну систему координат, яка відповідає заданому відображенню.

Спочатку в моделі створіть параметри для констант α і a.

Global Definitions : RMB > Parameters

Заповніть таблицю так, як показано нижче.

Parameters

Name	Expression	Value
alp	pi/6	0.5236
a	1/5+i/4	0.2+0.25i

Створіть нову систему координат.

`Definitions` -> **`Coordinate Systems`** – Mapped System

У вузлі **Definitions** створюється криволінійна система координат (вузол **Mapped System 3 (sys3)**), позначення і формули перетворення якої наводяться в ПВ цього вузла.

Definitions – Mapped System 3 (sys3):

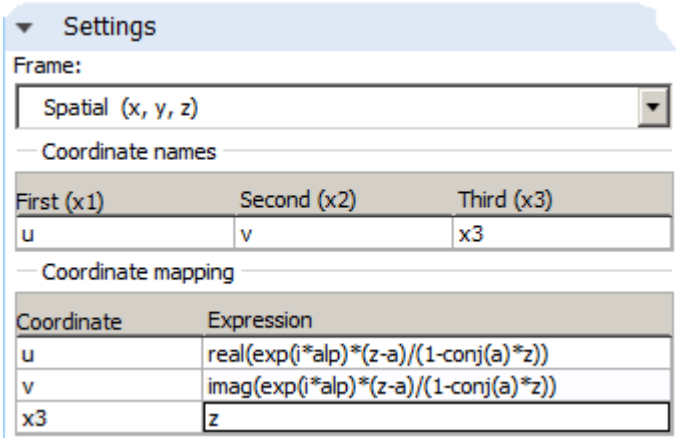
`Settings` – **Coordinate names – First, Second** = `u,v` (*введення в таблиці*)

`Settings` – **Coordinate mapping – Expression** =

`real(exp(i*alp)*(z-a)/(1-conj(a)*z))` (*вираз для u*)

`Settings` – **Coordinate mapping – Expression** =

`imag(exp(i*alp)*(z-a)/(1-conj(a)*z))` (*вираз для v*)



Створіть в геометрії коло одиничного радіуса, яке являтиме область в площині *w*.

`Geometry` -> **`Primitives`** – Circle

Geometry 1 - Circle 1 (c1):

`Size and Shape` – **Radius** = `1`

Оновіть дослідження.

Study 1: LMB > Compute

Створіть вузол досліджень, звужений на одиничне коло.

Results - Data Sets - Study 1/Solution 1 (1) : RMB > Duplicate

Data Sets - Study 1/Solution 1 (3) : RMB > Selection

Study 1/Solution 1 (3) - Selection:

`Geometric Entity Selection` – **Geometric entity level** = `Domain`

– **Selection** = `2,3` (*одиничне коло*)

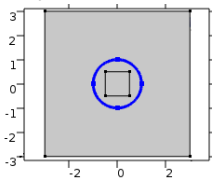
Для подальшого зображення межі кола створіть допоміжне ДД типу `Edge 2D`.

Results – Data Sets : RMB > Edge 2D

Data Sets - Edge 2D 2:

`Data` – **Data set** = `Study 1/Solution 1(1)`

`Selection` – **Selection** = `Manual; 9,10,11,12`



Тепер побудуйте лінії рівня $\sqrt{u^2+v^2}=Const$, $arctg\,\dfrac{y}{x}=Const$. Вони будуть прообразами полярної координатної сітки площини w.

2D Plot Group 5: RMB > Duplicate

2D Plot Group 6:

`Data` – **Data set** = `Study 1/Solution 1 (3)`

2D Plot Group 6– Contour 1:

`Expression` – **Expression** = `sqrt(sys3.u^2+sys3.v^2)`

`Levels` – **Entry method, Levels** = `Levels,range(0.1,0.1,1)`

`Coloring and Style` – **Coloring, Color** = `Uniform, Black`

2D Plot Group 6– Contour 2:

`Expression` – **Expression** = `atan2(sys3.v,sys3.u)`

Levels – **Entry method, Levels** = `Levels, range(-pi,pi/24,pi)`

Coloring and Style – **Coloring, Color** = `Uniform, Black`

Намалюйте контур кола.

2D Plot Group 6: RMB > Line

2D Plot Group 6 – Line 1:

Data – **Data set** = `Edge 2D 2`

Expression – **Expression** = `1`

Coloring and Style – **Line type** = `Tube`

– **Tube radius expression** = `0.01`

– **Radius scale factor** = `True,1`

– **Coloring, Color** = `Uniform, Black`

LMB>Plot (*наступний рисунок ліворуч*)

Задайте заливку клітинок отриманої координатної сітки в шаховому порядку.

`2D Plot Group 6->` **Add Plot ->** Surface

2D Plot Group 6 – Surface 1:

Expression – **Expression** = `sin(pi*(sqrt(sys3.u^2+sys3.v^2))/0.1)*sin(pi*(atan2(sys3.v,sys3.u))/(pi/24))>0`

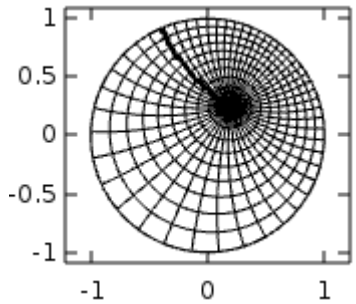
Тут в першому синусі дільник `0.1` є кроком в послідовності радіусів полярної сітки, а дільник `pi/24` – кроком в послідовності полярних кутів. Весь вираз є характеристичною функцією множини однокольорових клітинок криволінійної шахівниці.

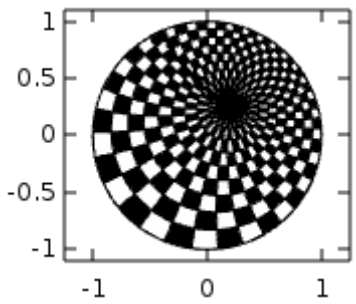
Coloring and Style – **Coloring, Color** =`Color table, GrayScale`

Quality – **Resolution** = `Extra fine`

Нагадаємо, що остання опція потрібна для покращання якості зображення клітинок шахівниці.

LMB > Plot (*наступний рисунок праворуч*)

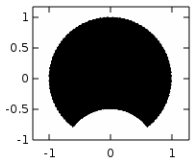




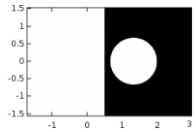
Вправи. Побудувати образи наступних областей.

1. Побудувати образ півкола $|z|<1,\ \operatorname{Im} z>0$ при відображенні $w=(2z-i)/(2+iz)$.

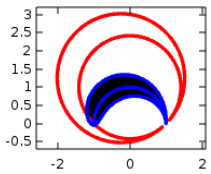
Вказівка. Застосувати зворотне перетворення $z=(-1+2iw)/(2i+w)$



2. Побудувати образ кільця $1<|z|<2$ при відображенні $w=z/(z-1)$.



3. Область D в площині (x, y) обмежена дугами окружностей $x^2+(y-1)^2=2$, $(x+1/4)^2+(y-5/4)^2=25/8$. Побудувати образ контура області D та образ самої області при відображенні Жуковського $w=(z+1/z)/2$.



Вказівка. Для побудови образа контура створіть функцію-ідентифікатор $\omega(x, y)$ області D і намалюйте контурну лінію $\omega(x, y)=0$ з вузлом `Deformation`. Для побудови образа області створіть в геометрії моделі обидва кола. У вузлі **Data Sets** створіть ДД, звужене на зону різниці цих кіл. В 2D графічній групі вузол `Surface`, який опирається на це ДД, повинен зображати характеристичну функцію $\chi_D=(\omega(x, y)>0)$ області D. Додайте до вузла `Surface` вузол `Deformation`.

Зауваження до вправи 3. Перше коло проходить через точки (1,0) і (-1,0). Друге коло проходить через точку (1,0) і має в цій точці з першим колом спільну дотичну. При відображенні Жуковського зовнішність першого кола відображається на зовнішність дуги окружності з кінцями в точках $(1,0)$ і $(-1,0)$. Друге коло відображається в замкнену криву з характерним вістрям і нагадує профіль крила літака.

Приклад 3. Створення і зображення еліптичної системи координат (α, β).

Один з можливих способів її визначення полягає в заданні співвідношення ([1], с. 219, задача 432)

$$x+i\,y=c\,\mathrm{ch}(\alpha+i\,\beta),\ 0\le\alpha<\infty,\ -\pi<\beta<\pi. \qquad (1)$$

Якщо зображатимуться лише координатні лінії, то достатньо формули (1). Але при потребі використовувати в формулах еліптичні координати (α, β) знадобиться обернена формула:

$$\alpha+i\,\beta=\mathrm{arccosh}((x+i\,y)/c). \qquad (2)$$

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Додайте в модель двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` – <> = `None`

Створіть вузол параметрів.

Global Definitions: RMB > Parameters

Parameters:

`Parameters` – Name, Expression = `c ,1`

Коли ви не збираєтесь застосовувати явні формули $\alpha=\alpha(x, y)$, $\beta=\beta(x, y)$, і вашими намірами є лише зображення координатних ліній, то підходить спосіб використання деформацій в графічних вузлах.

Definitions: RMB > Variables

Definitions – Variables 1:

`Variables` – Name, Expression = `Z, c*cosh(x+i*y)`

Щоб в графічних вузлах бачити не зображення всієї розрахункової області (нас цікавить координатна сітка), створіть і налаштуйте вузол `View` (вид) .

Component 1 – Definitions: RMB > View

Definitions - View 2:

`View` – **Lock axis =** `True`

Прапорець **Lock axis** (блокування осей) буде зберігати поточні границі виду після використання інструментів масштабування.

Definitions - View 2 - Axis:

`Axis` – **x minimum, x maximum =** `-2,2`

– **y minimum, y maximum =** `-2,2`

LMB > Update

Побудуйте геометрію. З формули (1) випливає, що область $\alpha \geq 0$ площини (α, β) відображається на всю комплексну площину $z = x + i\,y$.

Для представлення множини $\alpha \geq 0$ створіть прямокутник $0 \leq \alpha < \pi$, $-\pi < \beta < \pi$.

`Geometry` **-> `Primitives`** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

`Size` – **Width, Height** = `pi,2*pi` (*два поля*)

`Position` – **Base, x, y** = `Corner,0,-pi` (*три поля*)

LMB > Build All Objects

Побудуйте в геометрії прямокутну СкЕ сітку.

`Mesh` **-> `Generators`** – Mapped

Component 1 - Mesh 1 – Size:

`Element Size` – **Custom**= `True`

`Element Size Parameters` – **Maximum element size**= `pi/16`

LMB > Build All

Щоб змінні моделі були працездатними, потрібно виконати її компілювання. Це буде зроблено після створення і обчислення гілки досліджень `Study`.

Untitled.mph (root): RMB > Add Study = `Add Study ->`

Preset Studies – Stationary **->** X

Study 1: LMB > Compute

Якщо потрібно намалювати лише координатні лінії $\alpha = const$, $\beta = const$, то для цього достатньо формули (1). Ми зобразимо прямокутну координатну сітку і додамо до неї деформацію.

`Results` **-> `Plot Group`** **->** 2D Plot Group

2D Plot Group 1:

`Plot Settings` – **View=** `View 2`

– **Plot data set edges =** `False`

`Color Legend` – **Show legends =** `False`

`2D Plot Group 1`-> **`Add Plot`** **->** Surface

2D Plot Group 1 – Surface 1:

`Expression` – **Expression** = `0`

`Coloring and Style` – **Coloring, Color** = `Uniform, Black`

– **Wireframe** = `True`

RMB > Deformation

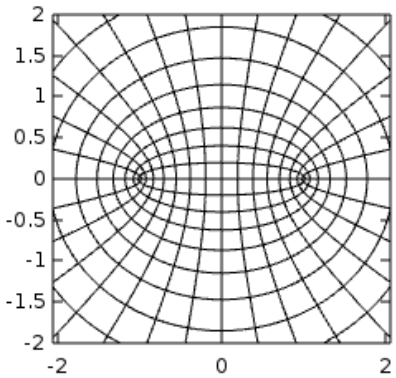
2D Plot Group 1 – Surface 1 – Deformation 1:

`Expression` – **x component** = `real(Z)-x`

– **y component** = `imag(Z)-y`

`Scale` – **Scale factor** = `True,1`

LMB > Plot



Інший спосіб зображення еліптичних координат полягає в застосуванні виразу (2). Ви створюєте вузол системи координат і задаєте в ньому явні формули $\alpha = \alpha(x, y)$, $\beta = \beta(x, y)$. Тоді незалежні змінні α і β можуть використовуватися в будь-яких місцях моделі, зокрема при побудові ліній рівня цих параметрів.

Створіть вузол криволінійної системи координат (α, β), пов'язаної з декартовими координатами формулами (2).

`Definitions` -> **`Coordinate Systems`** – Mapped System

В гілці **Definitions** з'являється вузол **Mapped System 2 (sys2)** системи координат, позначення і формули перетворення якої задаються в ПВ.

Definitions – Mapped System 2 (sys2):

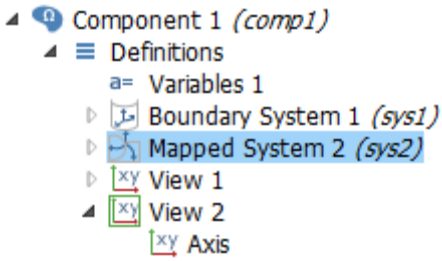
`Settings` – **Coordinate names – First, Second** = `a,b` (*введення в таблиці*)

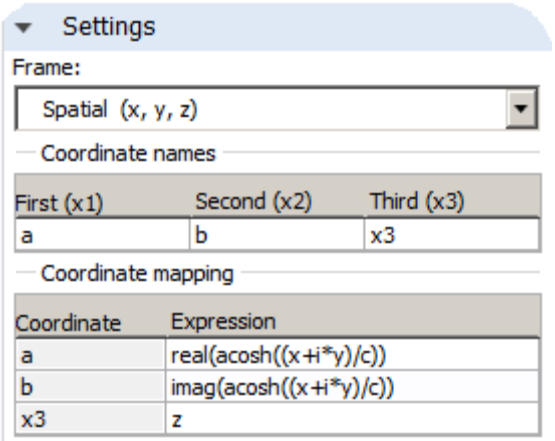
– **Coordinate mapping – Coordinate, Expression** =

`a, real(acosh((x+i*y)/c))`

– **Coordinate mapping – Coordinate, Expression** =

`b, imag(acosh((x+i*y)/c))`





Створені ідентифікатори змінних (додаючи префікс `sys2`) можна використовувати в моделі одночасно з декартовими координатами. Наприклад, змінні $\mathrm{sys2}.a, \mathrm{sys2}.b$ можна застосовувати при побудові координатних ліній.

Для побудови ліній координатної сітки нам знадобиться більша розрахункова область. Додайте до геометрії квадрат $-\pi \le x \le \pi$, $-\pi \le y \le \pi$.

`Geometry->`**`Primitives->`** Square

Geometry 1 - Square 1 (sq1):

`Size` – **Size length:** = `2*pi`

`Position` – **x:, y:** = `-pi, -pi` (*два поля*)

LMB > Build All Objects

Переобчисліть модель.

Study 1: LMB > Update Solution

Тепер можна будувати лінії $\alpha = const$, $\beta = const$.

`Results` **`-> Plot Group ->`** 2D Plot Group

2D Plot Group 2:

`Plot Settings` – **View=** `View 2`

– **Plot data set edges =** `False`

`Color Legend` – **Show legends =** `False`

`2D Plot Group 2->`**`Add Plot ->`** Contour

2D Plot Group 2 – Contour 1:

`Expression` – **Expression** = `sys2.a`

`Levels` – **Entry method, Levels** = `Levels, range(0,pi/16,pi)`

`Coloring and Style` – **Contour type** = `Line`

– **Coloring, Color** = `Uniform, Black`

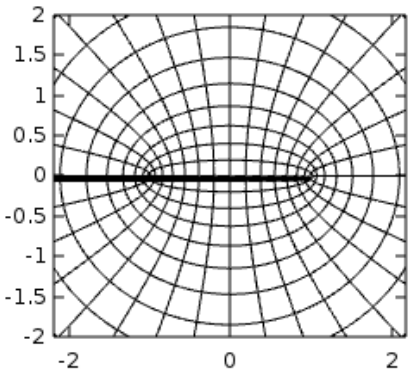
2D Plot Group 2 – Contour 1: RMB > Duplicate

2D Plot Group 2 – Contour 2:

`Expression` – **Expression** = `sys2.b`

`Levels` – **Entry method, Levels** = `Levels, range(-pi,pi/16,pi)`

LMB>Plot



Зауваження. Сенс координат (x, y) у вузлі **Variables 1** і в графічних групах різний. У вузлі **Variables 1** вони являють координати (α, β), а у вузлах **2D Plot Group 1**, **2D Plot Group 2** вони є декартовими координатами точок.

8.4. Конформні відображення.

Якщо комплексна функція $w = f(z) = u + i \cdot v$ аналітична (задовольняє умовам Коші-Рімана) і її похідна $f'(z) \neq 0$ відрізняється від нуля всюди в області *D*, то вона (функція) відображає цю область в деяку іншу Δ в площині $w = u + i \cdot v$ зі збереженням кутів між відповідними кривими. Кажуть, що функція $f(z)$ здійснює конформне відображення однієї області на іншу. Головною задачею теорії конформних відображень є побудова відображення однієї заданої області *D* площини *z* на іншу Δ, задану в площині *w*.

Нашою метою є знайомство читача з інструментами COMSOL Multiphysics, тому в прикладах ми не будемо наводити всі міркування, які призводять до побудови виразів конформних відображень, а відразу наводитимемо формули (даючи посилання). Оскільки конформне відображення реалізується комплекснозначною функцією, то її графічні зображення можна будувати за допомогою всіх прийомів, які були наведені раніше. Головна особливість таких відображень полягає в збереженні кутів між кривими, і тому ортогональні сітки однієї області переходять в ортогональні сітки іншої. Для ілюстрації конформних відображень часто обмежуються кресленням образів або прообразів контурів областей, а також образів/прообразів їх декартових або полярних сіток.

Приклад 1. Конформне відображення сектора одиничного кола на повне коло.

Сектор одиничного кола $|z| \le 1,\ 0 \le \arg z \le \beta$ відображається на одиничне коло $\Delta : |w| \le 1$ за допомогою перетворення [6, с. 434]

$$w = \frac{\left(1 + z^{\pi/\beta}\right)^2 - i \cdot \left(1 - z^{\pi/\beta}\right)^2}{\left(1 + z^{\pi/\beta}\right)^2 + i \cdot \left(1 - z^{\pi/\beta}\right)^2}.$$

Зобразити таке перетворення, побудувавши в Δ образ полярної сітки сектора *D*: $|z| \le 1,\quad 0 \le \arg z \le \pi/4$. Оскільки $\beta = \pi/4$, то останній вираз набуває вигляду

$$w = \frac{\left(1 + z^4\right)^2 - i \cdot \left(1 - z^4\right)^2}{\left(1 + z^4\right)^2 + i \cdot \left(1 - z^4\right)^2}. \qquad (1)$$

Зауваження. При конформному відображенні межа області *D* повинна переходити в межу Δ. Простіше на перший погляд перетворення $w = z^{2\pi/\beta}$ (при $\beta = \pi/4$) не задовольняє цій умові. Воно є перетворенням на коло з розрізом по радіусу на позитивній осі.

Розв'язання. Створіть модель з двовимірним компонентом і геометрією у формі сектора з центральним кутом $\beta = \pi/4$.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = None

Untitled.mph (root) : RMB > Add Component ->2D

Geometry -> **Primitives** – Circle

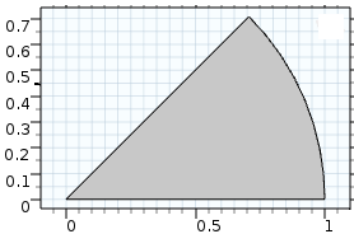
Geometry 1 - Circle 1(c1):

Size and Shape – **Radius:** = 1

– **Sector angle:** = 45

Position – **Base, x, y** = Center,0 ,0

LMB > Build All Objects



Для побудови образу полярної сітки площини z буде зручно створити відповідну систему координат.

Definitions -> **Coordinate Systems** – Cylindrical System

В гілці **Definitions** з'являється вузол **Cylindrical System 2 (sys2)**. В його ПВ задаються змінні полярної системи координат r і phi, які нарівні з позначеннями декартових координат можна використовувати у виразах.

Coordinate names

First (r)	Second (phi)	Third (a)
r	phi	a

Origin

x	y
0	0

Задайте в компоненті комплексну змінну z=x+i*y , і за формулою (1) – змінну *w* .

Definitions -> **Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = z, x+i*y

Variables – **Name, Expression**= w, ((1+z^4)^2-i*(1-z^4)^2)/
((1+z^4)^2+i*(1-z^4)^2)

Створіть вузол досліджень і виконайте його обчислення.

Untitled.mph (root): RMB > Add Study = Add Study ->
Preset Studies – Stationary -> X

Study 1: LMB > Compute

Щоб в подальшому накреслити зображення полярної сітки нам знадобляться «контурні» джерела даних, створювані за рівняннями sys2.r=Const і sys2.phi=Const, де sys2 є ім'ям полярної системи координат, побудованої вище.

Results -> Data Set – More Data Sets - Contour

Data Sets - Contour 1:

Expression– **Expression** = sys2.r

Levels – **Entry method**= Levels

Levels – **Levels** = range(0,0.05,1)

LMB>Plot (*наступний рисунок ліворуч*)

Results -> Data Set – More Data Sets - Contour

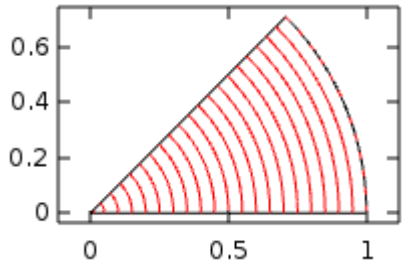
Data Sets - Contour 2:

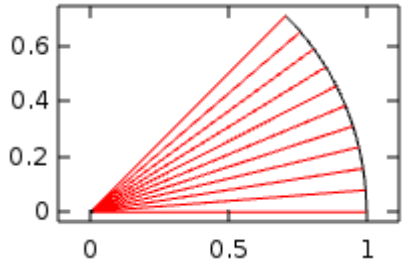
Expression– **Expression** = sys2.phi

Levels – **Entry method**= Levels

Levels – **Levels** = range(0,pi/40,pi/4)

LMB>Plot (*наступний рисунок праворуч*)





Накресліть образи кривих полярної сітки (з сектора площини z) на площині w одиничного кола $|w|<1$.

Results -> **Plot Group** -> 2D Plot Group

2D Plot Group 1:

Plot Settings – **Plot data set edges =** False

Color Legend – **Show legends =** False

2D Plot Group 1 -> Add Plot – Surface

2D Plot Group 1 - Surface 1 :

Expression – **Expression** = 0

Coloring and Style – **Coloring, Color** = Uniform, Gray

RMB > Deformation

Expression – **x component** = real(w)-x

– **y component** = imag(w)-y

Scale– **Scale factor** = True,1

2D Plot Group 1-> **Add Plot** -> Line

2D Plot Group 1 – Line 1:

Data – **Data set** = Contour 1

Expression – **Expression** = 0

Coloring and Style – **Coloring, Color** = Uniform, Black

RMB > Deformation

2D Plot Group 1 – Line 1 – Deformation 1:

Expression – **x component** = real(w)-x

– **y component** = imag(w)-y

Scale– **Scale factor** = True,1

2D Plot Group 1 – Line 1: RMB > Duplicate

2D Plot Group 1 – Line 2:

Data – **Data set** = Contour 2

LMB > Plot (*наступний рисунок ліворуч*)

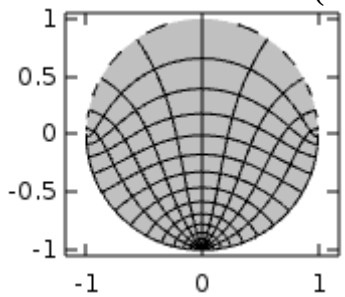
Якщо зараз деактивувати всі три вузли деформацій в гілці **2D Plot Group 1**, то ви отримаєте зображення входової області (сектора) з полярною сіткою.

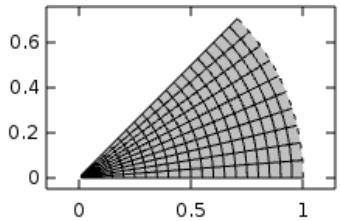
2D Plot Group 1 - Surface 1– Deformation 1 : RMB > Disable

2D Plot Group 1 – Line 1 – Deformation 1: RMB > Disable

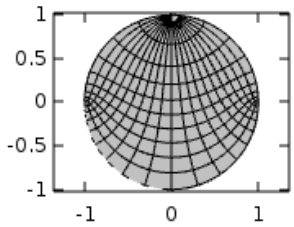
2D Plot Group 1 – Line 2 – Deformation 1: RMB > Disable

LMB > Plot (*наступний рисунок праворуч*)





Вправа. Конформне відображення правого півкола $|z| \le a$, $\operatorname{Re} z \ge 0$ на одиничне коло $|w| \le 1$ задається формулою $w = i \cdot \dfrac{z^2 + 2az - a^2}{z^2 - 2az - a^2}$. Зобразити це перетворення, побудувавши в колі $|w| \le 1$ образ полярної сітки.



■

Розглянемо на площині z однозв'язну область D, конформне відображення $w = W(z)$ якої на коло $|w| \le 1$ відомо. Вважатимемо, що зворотне відображення також задано $z = Z(w)$. Нашою метою буде побудова в D гармонічної функції, яка на межі має значення $u(\zeta)$, де ζ позначає комплексні координати точок контура D. Побудуємо функцію $U(\omega) = u(Z(\omega))$, де ω є комплексними координатами точок контура одиничного кола $|w| = 1$. Тоді на площині w аналітична в одиничному колі $|w| \le 1$ функція $F(w)$ з дійсною частиною $U(\omega)$ на межі, обчислюється за формулою (ф. (4) п. 8.2, або [2, с. 219])

$$F(w) = \frac{1}{2\pi i} \oint_{|w|=1} U(\omega) \frac{\omega + w}{\omega(\omega - w)} d\omega + i \cdot Const . \qquad (2)$$

При чому напрямок обходу контура $|w| = 1$ обирається проти годинникової стрілки. Оскільки $w = W(z)$, то функція $\Phi(z) = F(W(z))$ буде аналітичною в D і

$$\Phi(z) = F(W(z)) = \frac{1}{2\pi i} \oint_{|w|=1} u(Z(\omega)) \frac{\omega + W(z)}{\omega(\omega - W(z))} d\omega + i \cdot Const . \qquad (3)$$

На межі D дійсна частина функції $\Phi(z)$ матиме граничне значення

$$\operatorname{Re}[F(W(\zeta))] == \operatorname{Re}[F(W(Z(\omega)))] = \operatorname{Re}[F(\omega)] = U(\omega) =$$
$$= u(Z(\omega)) = u(Z(W(\zeta))) = u(\zeta).$$

Оскільки $\omega = W(\zeta)$, то $d\omega = W'(\zeta) d\zeta$, і (3) набуває вигляду

$$\Phi(z) = \frac{1}{2\pi i} \oint_C u(\zeta) \cdot \frac{W(\zeta) + W(z)}{W(\zeta) \cdot (W(\zeta) - W(z))} W'(\zeta) d\zeta + i \cdot Const ,$$

де C – контур області D, який проходиться проти годинникової стрілки. Враховуючи, що $\frac{d}{dz}\ln W(z)=\frac{W'(z)}{W(z)}$, матимемо

$$\Phi(z)=\frac{1}{2\pi i}\oint_C u(\zeta)\cdot\frac{W(\zeta)+W(z)}{W(\zeta)-W(z)}\cdot\frac{d}{d\zeta}\ln W(\zeta)\,d\zeta+i\cdot Const$$

Гармонічна функція, яка на межі області D має значення $u(\zeta)$, буде дійсною частиною функції $\Phi(z)$, тобто

$$U(z)=\mathrm{Re}\left[\frac{1}{2\pi i}\oint_C u(\zeta)\cdot\frac{W(\zeta)+W(z)}{W(\zeta)-W(z)}\cdot\frac{d}{d\zeta}\ln W(\zeta)\,d\zeta\right]. \qquad (4)$$

Приклад 2. Дана область D, яка обмежена правою гілкою лемніскати $\rho=\sqrt{\cos 2\theta}$, зсунутої вліво на 0.5 одиниць (тут ρ і θ полярні координати). Побудувати в D гармонічну функцію $U(x,y)$, яка на межі області має значення $U\big|_{\Gamma}=x^2$.

Конформне відображення на одиничне коло внутрішності правої гілки зазначеної лемніскати здійснюється функцією $w=2(z+0.5)^2-1$ [3, с. 39], а оберненим перетворенням, вочевидь, є $z=\sqrt{w+1}/\sqrt{2}-0.5$.

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = None

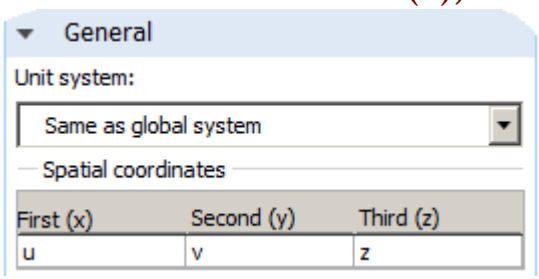
Для графічного представлення конформного відображення в площині $z=x+i\cdot y$, де розташована лемніската, побудуємо прообраз полярної сітки з площини $w=u+i\cdot v$, де розташовано одиничне коло $|w|\le 1$. Для цього нам знадобиться компонент з розрахунковою геометрією в формі одиничного кола і полярна система координат.

Untitled.mph (root) : RMB > Add Component ->2D

Декартови координати в площині w позначте u, v.

Component 1:

`General` – **Spatial coordinates – First (x), Second (y)**= `u, v`



Побудуйте розрахункову геометрію, яка складається з одного кола.

Geometry -> `Primitives` – Circle

Geometry 1 - Circle 1(c1):

`Size and Shape` – **Radius:** = 1

`Position` – **Base, x, y** = Center,0,0

Створіть полярну систему координат.

`Definitions` **-> Coordinate Systems** – Cylindrical System

В гілці **Definitions** з'являється вузол **Cylindrical System 2 (sys2)** з позначенням змінних системи координат.

Coordinate names

First (r)	Second (phi)	Third (a)
r	phi	a

Задайте в компоненті комплексну змінну `w=u+i*v` , і змінну Z за формулою $Z = \sqrt{w+1}/\sqrt{2} - 0.5$.

`Definitions` **-> Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = `w, u+i*v`

Variables – **Name, Expression**= `Z, sqrt(w+1)/sqrt(2)-0.5`

Створіть вузол досліджень і виконайте обчислення.

Untitled.mph (root): RMB > Add Study = `Add Study` ->

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Створіть два джерела даних типу `Contour`, які представлятимуть сімейства ліній $r = Const$, $\varphi = Const$ полярної системи координат (r, φ) площини *w*.

`Results` -> Data Set – More Data Sets - Contour

Data Sets - Contour 1:

Expression– **Expression** = `sys2.r`

Levels – **Entry method**= `Levels`

Levels – **Levels** = `range(0,0.05,0.95) 0.975 0.99 0.997 1`

LMB>Plot (*наступний рисунок ліворуч*)

`Results` -> Data Set – More Data Sets - Contour

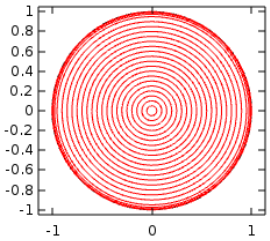
Data Sets - Contour 2:

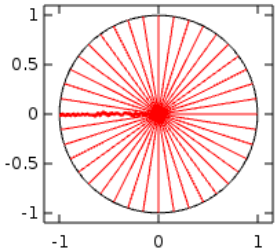
Expression– **Expression** = `sys2.phi`

Levels – **Entry method**= `Levels`

Levels – **Levels** = `range(-pi,pi/20,pi)`

LMB>Plot (*наступний рисунок праворуч*)





Побудуйте на площині *z* прообрази кривих полярної сітки площини *w*.

`Results` **-> Plot Group ->** 2D Plot Group

2D Plot Group 1:

Plot Settings – **Plot data set edges =** `False`

Color Legend – **Show legends =** `False`

`2D Plot Group 1` -> Add Plot – Surface

2D Plot Group 1 - Surface 1 :

Expression – **Expression** = 0

Coloring and Style – **Coloring**, **Color** = Uniform, Gray

RMB > Deformation

Expression – **x component** = real(Z)-u

– **y component** = imag(Z)-v

Scale– **Scale factor** = True,1

2D Plot Group 1-> **Add Plot** -> Line

2D Plot Group 1 – Line 1:

Data – **Data set** = Contour 1

Expression – **Expression** = 0

Coloring and Style – **Coloring, Color** = Uniform, Black

RMB > Deformation

2D Plot Group 1 – Line 1 – Deformation 1:

Expression – **x component** = real(Z)-u

– **y component** = imag(Z)-v

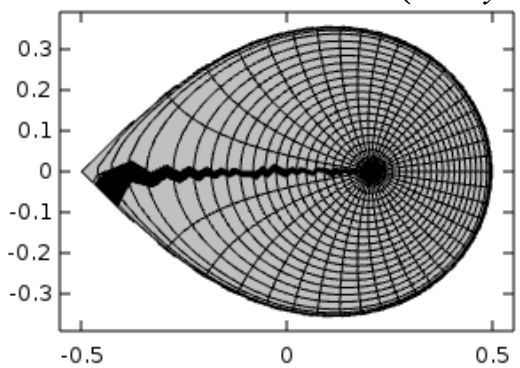
Scale– **Scale factor** = True,1

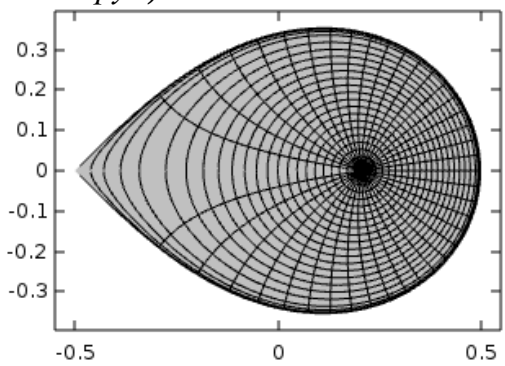
2D Plot Group 1 – Line 1: RMB > Duplicate

2D Plot Group 1 – Line 2:

Data – **Data set** = Contour 2

LMB > Plot (*наступний рисунок ліворуч*)





Отже одиничне коло $|w|<1$ при відображенні $Z=\sqrt{w+1}/\sqrt{2}-0.5$ переходить у внутрішню зону правої гілки лемніскати. А полярні координатні лінії – в криві, які зображено на рисунку.

Зауваження. Щоб на графіку позбутися товстої чорної ламаної можна з геометрії (або з джерела даних) одиничного кола $|w|\le 1$ виключити вузький прямокутник, який охоплює відрізок дійсної осі $-1\le u\le 0$ площини w. Зробіть це самостійно (попередній рисунок праворуч).

Перейдемо до розв'язання задачі побудови в зоні лемніскати гармонічної функції. Для цього нам знадобиться геометрія в формі цієї області. Створіть її у іншому компоненті поточного файлу.

Home -> **Model** - Add Component -> 2D

У вузлі **View 1** створеного компонента встановіть прапорець, який вмикає відображення напрямків зростання параметрів кривих.

Component 2 – Definitions - View 2:

View – **Show edge direction arrows** = True

Формулу (4) не можна застосовувати на межі області *D*. Тому будувати графік розв'язку будемо в області трохи менше ніж *D*. Товщину граничної смуги, яку в подальшому ми виключимо з геометрії, задамо у вигляді параметра моделі.

Global Definitions : RMB > Parameters

Parameters – **Name, Expression** = dh,0.05

При створенні геометрії у формі внутрішності правої гілки лемніскати виникають деякі труднощі. В системі немає інструментів побудови в геометрії моделі кривих, заданих в полярних координатах і, тим паче, фігур, обмежених такими кривими. Отже доведеться робити це в декартових координатах. Приймаючи полярний кут θ в якості параметра кривої, рівняння контура області можна записати в наступному вигляді:

$$x(\theta)=\rho\cos\theta-0.5=\sqrt{\cos 2\theta}\cdot\cos\theta-0.5,$$

$$y(\theta)=\rho\sin\theta=\sqrt{\cos 2\theta}\cdot\sin\theta.$$

При чому для правої гілки лемніскати параметр θ змінюватиметься в діапазоні $-\pi/4\le\theta\le\pi/4$. Для побудови граничної смуги нам знадобиться одиничний вектор нормалі $\boldsymbol{n}=(n_x,n_y)$ до цієї кривої. Його координати можна обчислювати за формулами

$$n_x=y'(\theta)/\sqrt{(x'(\theta))^2+(y'(\theta))^2},\ n_y=-x'(\theta)/\sqrt{(x'(\theta))^2+(y'(\theta))^2}.$$

Для спрощення подальших виразів, створіть ці чотири функції.

Global Definitions: RMB > Functions – Analytic

Function name = X

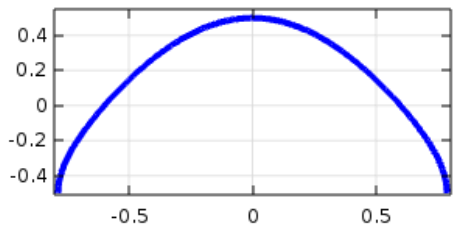
Definition – **Expression** = sqrt(cos(2*th))*cos(th)-0.5

– **Arguments** = th

Для перевірки побудуйте графік функції, пам'ятаючи, що параметр кривої не повинен виходити за межі діапазону $[-\pi/4,\pi/4]$.

Plot Parameters – **Low limit, Upper limit** = -pi/4, pi/4

LMB> Plot



Global Definitions: RMB > Functions – Analytic

Function name = Y

Definition – **Expression** = sqrt(cos(2*th))*sin(th)

– **Arguments** = th

Global Definitions: RMB > Functions – Analytic

Function name = nX

Definition – **Expression** = d(Y(th),th)/
sqrt(d(X(th),th)^2+d(Y(th),th)^2)

– **Arguments** = th

Global Definitions: RMB > Functions – Analytic

Function name = `nY`

Definition – **Expression** = `-d(X(th),th)/sqrt(d(X(th),th)^2+d(Y(th),th)^2)`

– **Arguments** = `th`

Створіть також функцію $u(\zeta)$ граничного значення, яку зручно задати як функцію координат (`x,y`).

Global Definitions: RMB > Functions – Analytic

Function name = `u`

Definition – **Expression** = `x^2`

– **Arguments** = `x,y`

Переходимо до побудови геометрії.

Component 2: (*активуйте другий компонент*)

Лемніската, як параметрична крива (навіть замкнута), буде створюватися як лінія, а не солідна фігура. Тому нам знадобиться інструмент Convert to Solid перетворення в суцільну фігуру (в 2D моделях) набору кривих, які з'єднані в замкнуту лінію. Йому потрібно принаймні дві криві. Тому лемніскату створимо з двох шматків.

`Geometry->`**`Primitives->`** Parametric Curve

Geometry 2 - Parametric Curve 1 (pc1) :

Parameter – **Name, Minimum, Maximum** = `th,-pi/4, 0`

Expressions – **x, y** = `X(th),Y(th)`

Parametric Curve 1 (pc1) : RMB > Duplicate

Parametric Curve 2 (pc2) :

Parameter – **Name, Minimum, Maximum** = `th,0,pi/4`

`Geometry ->` **`Operations - Conversions ->`** Convert to Solid

Geometry 2 - Convert to Solid 1 (csol1) :

Input – **Input Objects** = `pc1, pc2`

LMB > Build Selected (*наступний рисунок ліворуч*)

Ми вже казали, що при побудові графіка розв'язку використовуватимемо зону трохи меншу за *D*. Для цього в геометрії нам знадобиться прикордонна смуга. Щоб її створити достатньо буде побудувати еквідистанту до контура області. Її теж краще створити з двох шматків – еквідистант до кожної ділянки лемніскати.

Parametric Curve 1 (pc1) : RMB > Duplicate

Parametric Curve 3 (pc3) :

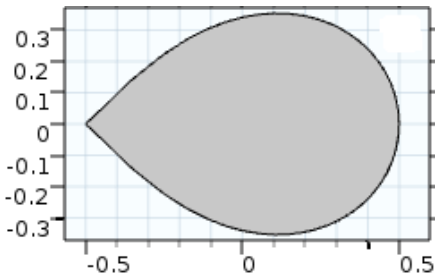
Expressions – **x, y** = `X(th)-nX(th)*dh, Y(th)-nY(th)*dh`

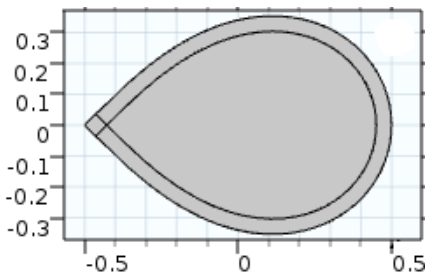
Parametric Curve 3 (pc3) : RMB > Duplicate

Parametric Curve 4 (pc4 :

Parameter – **Name, Minimum, Maximum** = `th,0,pi/4`

LMB > Build All Objects (*наступний рисунок праворуч*)





Поки що ми використали широку приграничну смугу. Пізніше ми зробимо її тонше.

Створіть оператор інтегрування по контуру області, тобто по лемніскаті.

`Definitions` > Couplin - `Component Couplings` -> Integration

Definitions - Integration 1 (intop1):

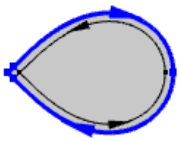
Operator name = `intC`

`Source Selection` – **Geometric entity level** = `Boundary`

– **Selection** = `Manual, 1,2,6,8`

(*дуги зовнішньої лемніскати*)

Будьте уважні, не виділіть дуги внутрішньої кривої.



Зверніть увагу на напрямок вздовж кривої інтегрування. Він спрямований по ходу годинникової стрілки.

Задайте в компоненті комплексну змінну `z=x+i*y` , допоміжні змінні, і за формулою (4) – змінну U .

`Definitions` -> **`Variables`** – Local Variables

Definitions – Variables 1:

`Variables` – **Name, Expression** = `z, x+i*y`

`Variables` – **Name, Expression** = `W, 2*(z+0.5)^2-1`

`Variables` – **Name, Expression** = `W1, d(log(W),z)`

`Variables` – **Name, Expression** = `F, intC(u(x,y)*(W+dest(W))/(W-dest(W))*W1*(tx+i*ty)) /(2*pi*i)`

`Variables` – **Name, Expression** = `U, -real(F)`

В останній формулі попереду було поставлено знак мінус у зв'язку з тим, що в операторі інтегрування `intC` зовнішній контур області (лемніската) проходиться в протилежному порівняно з формулою (4) напрямку.

Виконайте обчислення моделі

Study 1: LMB > Compute

Нових джерел даних в ДМ не з'явиться, але обчислення змінних другого компонента відбудеться (якщо не було повідомлень про помилки). Створіть ДД, яке «прив'язано» до другого компонента.

Results: Data Sets - Study 1/Solution 1 : RMB > Duplicate

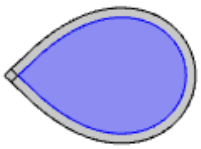
Study 1/Solution 1 (2):

`Solution` – **Component** = `Component 2(comp 2)`

Графік змінної `U` доцільно будувати над трохи звуженою геометрією, яка співпадає з усією, за винятком тонкого шару біля межі області. Створіть для цього відповідне ДД.

Results: Data Sets - Study 1/Solution 1 (2) : RMB > Duplicate

Data Sets - Study 1/Solution 1 (3) : RMB > Selection
Study 1/Solution 1 (3) - Selection:

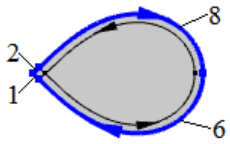


Geometric Entity Selection –
Geometric entity level = Domain
– **Selection** = 3

Тут в ДД не включено прикордонну смугу біля межі лемніскати.

Додатково ми збираємося будувати графік граничної функції u над контуром області. Для цього нам знадобиться ДД типу Edge 2D.

Results – Data Sets : RMB > Edge 2D
Data Sets - Edge 2D 1:



Data – **Data set** = Study 1/Solution 1(2)
Selection – **Selection** = Manual; 1,2,6,8

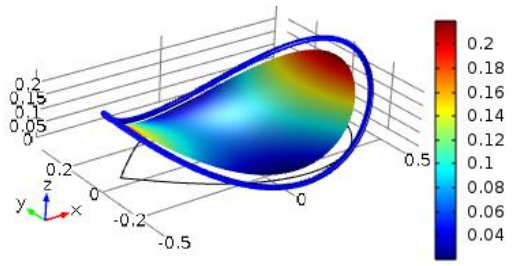
Все готово для побудови графіка U.

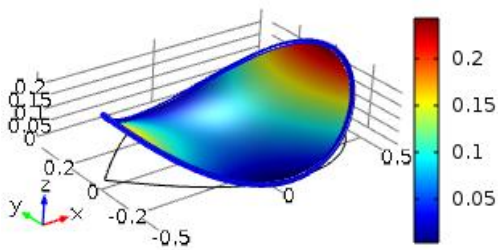
Results **-> Plot Group ->**2D Plot Group
2D Plot Group 2:
Data – **Data set** = Study 1/Solution 1(2)
2D Plot Group 2-> **Add Plot ->** Surface
2D Plot Group 2 – Surface 1:
Data – **Data set** = Study 1/Solution 1(3)
Expression – **Expression** = U
RMB > Height Expression
2D Plot Group 2– Surface 1 – Height Expression 1:
Axis – **Scale factor** = True,1

Для наочності додайте графік граничної функції $u\big|_{\Gamma} = x^2$, яка визначена лише на контурі області.

2D Plot Group 1-> **Add Plot ->** Line
2D Plot Group 2 – Line 1:
Data – **Data set** = Edge 2D 1
Expression – **Expression** = 0
Coloring and Style – **Line type** = Tube
– **Tube radius expression** = 0.01
– **Radius scale factor** = True,1
– **Coloring, Color** = Uniform, Blue
RMB > Height Expression
2D Plot Group 1 – Line 1 – Height Expression 1:
Expression – **Height data** = Expression
– **Expression** = u(x,y)
Axis – **Scale factor** = True,1
LMB>Plot (*наступний рисунок ліворуч*)

Щілина між кривою і поверхнею пояснюється виключенням з геометрії прикордонного шару.





Виключену з геометрії смугу зробіть тонше. Для цього поверніться до вузла параметрів і зменшить параметр товщини.

Global Definitions - Parameters:

`Parameters` – **Name, Expression** = `dh,0.01`

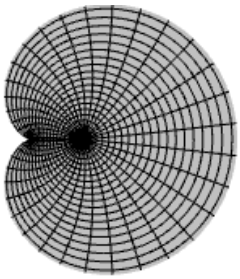
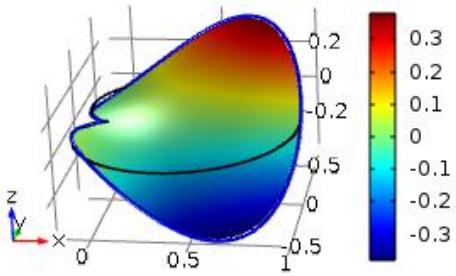
Переобчисліть модель.

Study 1: LMB > Compute

Перебудований графік вузла **2D Plot Group 1** зображено на попередньому рисунку праворуч.

Вправа. Якщо граничну функцію задати у вигляді $u(x,y)=x^2-y^2$, то буде відомий точний розв'язок задачі – це така ж сама функція $U(x,y)=x^2-y^2$. Порівняйте графіки гармонічної функції, отриманий за формулою (4), і за точною формулою.

Вправа. Дана область *D*, яка обмежена кардіоїдою $\rho=\frac{1}{2}(1+\cos\theta)$ (ρ і θ полярні координати). Конформне відображення на одиничне коло $|w|\le 1$ внутрішності кардіоїди здійснюється функцією $w=2\sqrt{z}-1$ [3, с. 39 задач 235], а оберненим перетворенням є $z=(w+1)^2/4$. Зобразити в площині кардіоїди прообрази полярної сітки з площини кола $|w|\le 1$. Побудувати в *D* гармонічну функцію $U(x,y)$, яка на межі області має значення $U|_{\Gamma}=x\cdot y$.

Вказівка. Внутрішня еквідистанта має розрив в околі особливої точки. Замкніть його будь-якою лінією, наприклад, кривою Безьє.

8.5. Приклади застосування комплексних потенціалів.

Велика кількість застосувань комплексного аналізу знаходиться в дослідженні векторних полів. Під векторним полем розуміють частину простору, з кожною точкою якого пов'язано значення деякого вектора. Оскільки комплексні величини можна інтерпретувати як точки або вектори на площині, то при їх

використанні доводиться обмежуватися двовимірними стаціонарними векторними полями. Стаціонарність означає незалежність від часу, а двовимірність – обмежує розгляданням плоскопаралельних векторних полів. Для опису таких полів достатньо двох скалярних функцій (двох компонент вектора плоского поля) двох аргументів (координат точки на площині).

Особливий інтерес становлять наступні приклади векторних полів [7].

а) *Електростатичне поле*. В просторі, що оточує електричні заряди, діють сили. Сила, діюча на одиничний пробний заряд в деякій точці, називається напруженістю електричного поля $\boldsymbol{E}$ в цій точці. Сукупність точок простору, оточуючих заряд, разом з пов'язаними з ними векторами напруженості $\boldsymbol{E} = (E_x, E_y)$, утворюють поле. Електростатичне поле всюди потенціальне, тобто $\operatorname{rot} \boldsymbol{E} = 0$. Якщо у всіх точках деякої області D щільність зарядів дорівнює нулю, то $\operatorname{div} \boldsymbol{E} = 0$ в цій області.

З умови $\operatorname{rot} \boldsymbol{E} = \dfrac{\partial E_y}{\partial x} - \dfrac{\partial E_x}{\partial y} = 0$ випливає існування функції $V = V(x, y)$ такої, що

$$E_x = -\frac{\partial V}{\partial x},\ E_y = -\frac{\partial V}{\partial y}, \tag{1}$$

тобто $\boldsymbol{E} = -\operatorname{grad} V$. Функцію $V = V(x, y)$ називають потенціалом поля. Оскільки $\operatorname{grad} V$ завжди спрямований по нормалі до ліній рівня $V(x, y) = Const$, то вектор напруженості $\boldsymbol{E}$ перпендикулярний до цих еквіпотенціальних ліній.

З умови $0 = \operatorname{div} \boldsymbol{E} = \dfrac{\partial E_x}{\partial x} + \dfrac{\partial E_y}{\partial y}$ випливає існування функції $U = U(x, y)$ такої, що

$$E_x = \frac{\partial U}{\partial y},\ E_y = -\frac{\partial U}{\partial x}. \tag{2}$$

Це означає, що вектор $\boldsymbol{E}$ є дотичним до ліній $U(x, y) = Const$. Їх називають силовими лініями поля, а $U(x, y)$ - силовою функцією поля.

Порівнюючи формули (1) і (2), бачимо, що в області D, яка не містить зарядів, силова і потенціальні функції пов'язані між собою співвідношеннями

$$\frac{\partial U}{\partial x} = \frac{\partial V}{\partial y},\ \frac{\partial U}{\partial y} = -\frac{\partial V}{\partial x}.$$

Останнє можна розглядати, як рівняння Коші – Рімана регулярності функції комплексного змінного $w = F(z) = U(x, y) + i \cdot V(x, y)$, яка називається комплексним потенціалом електростатичного поля. Через комплексний потенціал обчислюються всі головні величини, пов'язані з полем. Зокрема

$$\boldsymbol{E} = E_x + i \cdot E_y = -i \cdot \overline{F'(z)}. \tag{3}$$

Комплексний потенціал здійснює конформне відображення області D площини z на деяку частину площині w, якщо в D напруженість $\boldsymbol{E}$ відмінна від нуля. При чому на площині комплексного потенціалу w силовим і еквіпотенціальним

лініям поля відповідають прямі, паралельні координатним осям U і V. Знаючи комплексний потенціал поля $F(z)$, можна на площині $z = x + i \cdot y$ знайти силові $U(x, y) = Const$ і еквіпотенціальні $V(x, y) = Const$ лінії, тобто визначити картину поля.

Гранична умова, якій повинна задовольняти потенціальна функція V, полягає в її сталості на поверхні провідника, тобто $F(z)$ повинна на такій поверхні мати сталу уявну частину. Якщо розглядати $F(z)$ як конформне відображення площини z на площину w, контури провідників в площині z після відображення повинні переходити в прямі V=const, тобто в паралелі до дійсної осі. При чому, коли на якомусь граничному провіднику потенціал задано, наприклад V_0, то відповідна паралель матиме рівняння $V = V_0$. Інколи сюди додаються умови в нескінченних точках.

Отже будь-якому плоскому електростатичному полю в області, вільній від зарядів, відповідає регулярна функція – комплексний потенціал.

б) *Поле швидкостей руху рідини.* В кожний момент часу в будь-якій точці області D, зайнятій рухомою рідиною, цілком визначений вектор $\boldsymbol{V} = V_x + i \cdot V_y$ швидкості руху рідини. Сукупність всіх точок області D разом з пов'язаними з ними векторами $\boldsymbol{V}$ утворюють векторне поле. Якщо в деякій точці джерел рідини немає, то $\operatorname{div} \boldsymbol{V} = 0$. При відсутності вихорів вектор швидкості усталеної течії рідини задовольняє співвідношенню $\operatorname{rot} \boldsymbol{V} = 0$.

Для плоскої усталеної течії нестисливої рідини умова $\operatorname{div} \boldsymbol{V} = 0$ відсутності в області D джерел і умова $\operatorname{rot} \boldsymbol{V} = 0$ відсутності вихорів

$$\frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} = 0, \ \frac{\partial V_y}{\partial x} - \frac{\partial V_x}{\partial y} = 0$$

дозволяє стверджувати, що існує функція $\varphi(x, y)$ така, що

$$V_x = \frac{\partial \varphi}{\partial x}, \ V_y = \frac{\partial \varphi}{\partial y}, \text{ тобто } \boldsymbol{V} = \operatorname{grad} \varphi(x, y),$$

а також функція $\psi(x, y)$ така, що

$$V_x = \frac{\partial \psi}{\partial y}, \ V_y = -\frac{\partial \psi}{\partial x},$$

Функцію $\varphi(x, y)$ називають потенціалом, а $\psi(x, y)$ - функцією течії.

Легко бачити, що функції $\varphi(x, y)$ і $\psi(x, y)$ пов'язані умовами Коші-Рімана, і їх можна представляти у вигляді дійсної і уявної частини функції $w = \Phi(z) = \varphi(x, y) + i \cdot \psi(x, y)$, яка називається комплексним потенціалом течії. Він є регулярною функцією в будь-якій області течії, яка не містить джерел і вихорів, і реалізує конформне відображення області течії D площини z на площину w.

За допомогою комплексного потенціалу обчислюються всі головні величини, пов'язані з плоскою течією, наприклад, вектор швидкості течії

$$\boldsymbol{V} = V_x + i \cdot V_y = \overline{\Phi'(z)}. \tag{4}$$

На площині z лініям течії відповідають лінії $\operatorname{Im} w = \psi(x, y) = Const$, а еквіпотенціальним – лінії $\operatorname{Re} w = \varphi(x, y) = Const$.

Потік Q рідини через довільну криву C в області D обчислюється за формулою

$$Q = \int_C (\boldsymbol{V}, \boldsymbol{n}) d\,l = \int_C (\boldsymbol{V}, -i\,d\,z) = \int_C \left(V_x + i \cdot V_y,\ d\,y - i\,d\,x\right) =$$

$$= \int_C V_x d\,y - V_y d\,x = \int_C \frac{\partial \psi}{\partial y} d\,y + \frac{\partial \psi}{\partial x} d\,x = \int_C d\,\psi = \psi(z_2) - \psi(z_1), \qquad (5)$$

де z_1, z_2 - кінці C, $\boldsymbol{n}$ – одиничний вектор нормалі до кривої, круглі дужки $(\ldots,\ldots)$ позначають скалярний добуток.

Нехай C – замкнутий контур, обхід якого виконується в позитивному напрямку. Величину

$$\Gamma = \oint_C (\boldsymbol{V}, \boldsymbol{s}) d\,l = \oint_C (\boldsymbol{V}, d\,z) = \oint_C \left(V_x + i \cdot V_y,\ d\,x + i\,d\,y\right) =$$

$$= \int_C V_x d\,x + V_y d\,y = \int_C \frac{\partial \varphi}{\partial x} d\,x + \frac{\partial \varphi}{\partial y} d\,y = \int_C d\,\varphi, \qquad (6)$$

де $\boldsymbol{s}$ – одиничний дотичний вектор до кривої, називають циркуляцією вектора швидкості $\boldsymbol{V}$ вздовж кривої C.

Оскільки $\Phi'(z) d\,z = (V_x - i \cdot V_y)(d\,x + i \cdot d\,y)$, то з (5) і (6) випливає, що

$$Q = \operatorname{Im} \oint_C \Phi'(z) d\,z\ ,\ \Gamma = \operatorname{Re} \oint_C \Phi'(z) d\,z\,.$$

Поєднуючи ці формули, отримуємо

$$\Gamma + i \cdot Q = \oint_C \Phi'(z) d\,z\,. \qquad (7)$$

В гідродинаміці крайові задачі ставляться як задачі обтікання контурів C області D, в якій рухається рідина. Тому криві C повинні бути лініями течії рідини.

Якщо D смуга, то різниця значень функції течії на граничних кривих $\Delta\psi = \psi(z_2) - \psi(z_1)$ відповідно до (5) є витратами рідини – кількістю рідини, що протікає за одиницю часу через будь-який поперечний переріз смуги D. Тут $\Delta\psi$ відіграє роль різниці потенціалів в електростатиці.

Якщо D – зовнішність скінченної області, то ділянки межі є частинами ліній течії. Окрім цього на контурі області D може бути задана ненульова циркуляція Γ вектора швидкості $\boldsymbol{V}$, що є додатковою граничною умовою. Ще однією умовою для такої задачі є задання вектора швидкості на нескінченності.

в) *Поле вектора потоку тепла*. В просторі, який оточує нерівномірно нагріте тіло, відбувається перенесення тепла від більш нагрітих зон до менш нагрітих. Вектором потоку тепла називається вектор $\boldsymbol{Q} = -k \operatorname{grad} V$, де k – коефіцієнт внутрішньої теплопровідності, V - температура. Він спрямований в бік зменшення температур. Сукупність векторів $\boldsymbol{Q} = Q_x + i \cdot Q_y$ утворює

розглядуване поле. Умова відсутності джерел тепла для цього поля має вигляд $\operatorname{div}\boldsymbol{Q}=0$. Вважаючи k сталим, матимемо $\operatorname{rot}\boldsymbol{Q}=-k\operatorname{rot}\operatorname{grad}V=0$. З цих умов

$$\frac{\partial Q_x}{\partial x}+\frac{\partial Q_y}{\partial y}=0,\ \frac{\partial Q_y}{\partial x}-\frac{\partial Q_x}{\partial y}=0 \tag{8}$$

для усталеного теплового потоку в області, яка не містить теплових джерел, випливає, що роль потенціалу відіграє температура V:

$$\boldsymbol{Q}=-k\operatorname{grad}V,\ \frac{\partial V}{\partial x}=-\frac{1}{k}Q_x,\ \frac{\partial V}{\partial y}=-\frac{1}{k}Q_y.$$

З першого співвідношення (8) випливає, що вираз $\frac{1}{k}\left(Q_y d\,x-Q_x d\,y\right)$ є повним диференціалом деякої функції $U=U(x,y)$, яка називається функцією току тепла і пов'язана з температурою умовами Коші-Рімана

$$\frac{\partial U}{\partial x}=\frac{\partial V}{\partial y}=-\frac{1}{k}Q_y,\ \frac{\partial U}{\partial y}=-\frac{\partial V}{\partial x}=\frac{1}{k}Q_x.$$

Вона має сталі значення на лініях току. Функція комплексного змінного $w=\Psi(z)=U(x,y)+i\cdot V(x,y)$ називається комплексним потенціалом теплового поля і є регулярною в будь-якій однозв'язній області, в якій потік задовольняє умовам (8). Вектор потоку тепла обчислюється за формулою

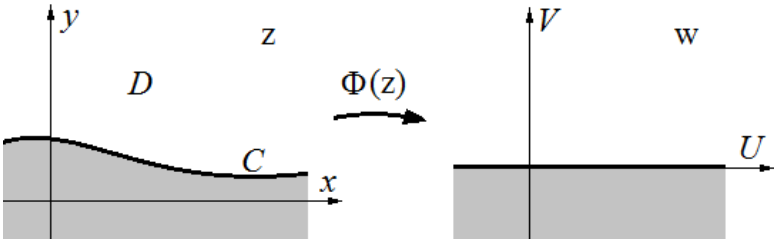
$$\boldsymbol{Q}=-k\operatorname{grad}V=-i\cdot k\,\overline{\Psi'(z)}. \tag{9}$$

Зауваження. Аналогічно до трьох згаданих векторних полів визначаються стаціонарне плоскопаралельне магнітне поле і поле постійного струму в суцільному середовищі. Наприклад, в частині простору, вільній від джерел, магнітне поле $\boldsymbol{H}$ задовольняє рівнянням $\operatorname{rot}\boldsymbol{H}=0$, $\operatorname{div}\boldsymbol{H}=0$, звідки випливає існування потенціалу V (тобто $\boldsymbol{H}=-\operatorname{grad}V$) і силової функції U, та як наслідок – існування комплексного потенціалу.

Таким чином, на площині комплексного потенціалу $w=F=U+i\cdot V$ силовим і еквіпотенціальним лініям полів розглянутої природи відповідають прямі, паралельні координатним осям U (інше позначення φ) і V (інше позначення ψ) площини w. Побудувавши їх прообрази в області D площини $z=x+i\cdot y$, можна визначити картину поля в цій області. При чому, немає потреби заглиблюватися в фізичну постановку задачі. Оскільки комплексний потенціал $F(z)$ є аналітичною функцією, то при умові $F'(z)\neq 0$ в D, він реалізовуватиме конформне відображення області D на площину (або частину площини) комплексного потенціалу w. Для його побудови достатньо знайти якесь конформне відображення $f(z)$ області D на площину комплексного потенціалу. Тоді $F(z)=k\cdot f(z)+C$, де k дійсна, а C комплексна сталі, які відшукуються з додаткових умов. В задачах електростатики і гідродинаміки фізичний зміст має лише похідна комплексного потенціалу, яка пов'язана з напруженістю електричного поля $\boldsymbol{E}$ чи з вектором $\boldsymbol{V}$ швидкості руху рідини формулами (3) або (4). Тому в цих задачах константа C не відіграє ніякої ролі і її можна покладати нулем.

Перелічимо деякі прикладні задачі, при розв'язанні яких зручно застосовувати комплексні потенціали.

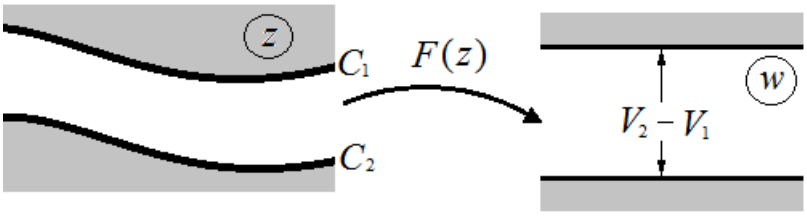
1. Визначити поле в області D, обмеженої контуром C, який проходить через нескінченно віддалену точку. Вважається заданою величина вектора поля в нескінченності, наприклад, величина вектора швидкості руху рідини $V_\infty = |\boldsymbol{V}(\infty)|$.



Для розв'язання задачі будується конформне відображення $w = f(z)$ області D площини z на півплощину $\operatorname{Im} w > 0$, і в комплексному потенціалі $\Phi(z) = k \cdot f(z)$ підбирається стала k таким чином, щоб величина поля на нескінченності співпадала з заданим значенням. Наприклад, в задачі руху рідини $\boldsymbol{V} = \overline{\Phi'(z)}$ (див. ф. (4)) потрібно виконання рівності $V_\infty = |\boldsymbol{V}_\infty| = |\Phi'(\infty)| = k \cdot |f'(\infty)|$, з якої визначається параметр k.

2. Визначити векторне поле в криволінійній смузі D між кривими C_1 і C_2, які не мають спільних скінченних точок (перетинаються лише на нескінченності). Потенціальна функція має на C_1 і C_2 сталі значення, і різниця потенціалів ΔV (потенціальної функції) між ними вважається відомою.

Якщо конформне відображення $f(z)$ криволінійної смуги D на якусь горизонтальну смугу площини w відомо, то комплексний потенціал будується за формулою $F(z) = k\ f(z) + C$, де дійсна стала k і комплексна стала C обираються з додаткових умов (зазвичай з умови відомої різниці потенціальної функції $\Delta V = V_2 - V_1$ на берегах криволінійної смуги D; ΔV одночасно представляє ширину горизонтальної смуги в площині w). Стала C потрібна лише в задачах дослідження теплового поля, коли на берегах криволінійної смуги задається значення температури (потенціальної функції), а не різниця температур/потенціалів.



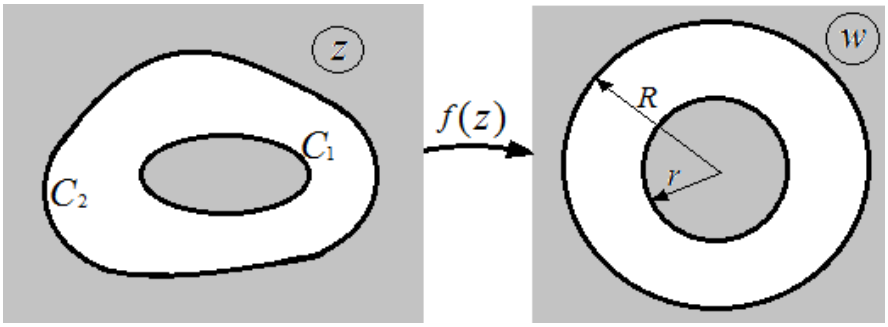
3. Визначити поле в криволінійному кільці D між двома замкнутими неперетинними скінченними кривими C_1 і C_2. Різниця потенціалів (потенціальної функції) між ними відома.

Ця задача відрізняється від попередньої тим, що криві C_1 і C_2 не мають спільних точок, область D між ними двохзв'язна і її конформне відображення на смугу в площині w неможливе. Однак задача розв'язується, якщо відомо

конформне відображення $w = f(z)$ області D на кругове кільце $r < |w| < R$. Для кругового кільця комплексним потенціалом є функція $W = k\,i \ln w + C$, де k – дійсна і C довільна комплексна сталі. Справді, для $W = U + i \cdot V$ уявна частина $V = k \ln|w| + \mathrm{Im}C$ на колах $|w| = r$ і $|w| = R$ має сталі значення і, розпоряджаючись константами k і $\mathrm{Im}C$, можна досягти будь-якої заданої різниці потенціалів ΔV між контурами кільця (або заданих значень потенціалів на C_1 і C_2). Підставляючи $f(z)$, отримуємо комплексний потенціал

$$W = k\,i \ln f(z) + C \tag{10}$$

в області D.



4. Визначити поле в зовнішності області D, обмеженої замкнутим скінченним контуром C. Вважається заданим вектор поля в нескінченності (величина і напрямок).

Якщо конформне відображення $f(z)$ області D площини z на область G площини $w = U + i \cdot V$ відоме, то аналітична функція $w = F(z) = k \cdot f(z)$, що його реалізує (з точністю до дійсного множника k, який обирається із додаткової умови), буде комплексним потенціалом поля в цій області. Тоді картина поля в COMSOL будується як лінії рівня $\mathrm{Re}\,F(z) = Const$ і $\mathrm{Im}\,F(z) = Const$. Якщо відоме конформне відображення $z = G(w)$ області G площини w на область D площини z (тобто в зворотну сторону), то картина поля будується наступним чином. В області G створюється декартова сітка. Образи її ліній при відображенні $G(w)$ будуть лініями течії і еквіпотенціальними лініями поля.

Таким чином, алгоритм розв’язання задач поточного параграфа в COMSOL складається з наступних кроків.

1. Якщо відомий комплексний потенціал $w = F(z)$, то будується 2D компонент з геометрією в формі області, в якій будуватиметься картина векторного поля.

Якщо відоме обернене до $F(z)$ відображення $z = G(w)$, то будується 2D компонент з геометрією площини w і декартовою сіткою.

2. Створюються константи моделі і допоміжні функції.

3. Якщо відомий комплексний потенціал $w = F(z)$, то створюються змінні `z=x+i*y` і змінна комплексного потенціалу `w=F(z)`.

Якщо відоме обернене відображення $z = G(w)$, то створюються змінні `w=U+i*V` і змінна `z=G(w)`.

4. Додається вузол дослідження і виконується обчислення.
5. Якщо відомий комплексний потенціал $w = F(z)$, то будуються лінії течії $\operatorname{Im} F(z) = Const$ і еквіпотенціальні лінії $\operatorname{Re} F(z) = Const$ досліджуваного поля.

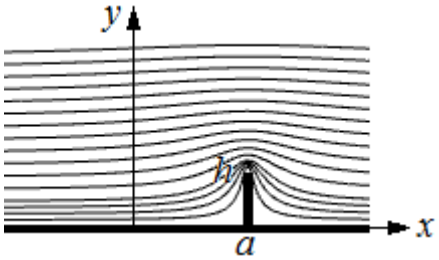
Якщо відоме обернене відображення $z = G(w)$, то у графічному вузлі **Surface** з встановленою опцією **Wireframe** = `True` за допомогою підвузла **Deformation** будується образ декартової сітки.

Головна складність схована в знаходженні конформного відображення, яке використовується при побудові комплексного потенціалу. Оскільки нашою метою є знайомство читача з інструментами COMSOL Multiphysics, в наступних прикладах ми відразу наводитимемо формули відповідного відображення (даючи посилання), не вдаючись в спосіб їх отримання.

Зауваження. Оскільки приклади цього параграфа не застосовують метод скінчених елементів, то їх можна реалізувати в будь-якому математичному пакеті, елементарні функції якого «вміють працювати» з комплексними величинами.

8.5.1. Обтікання вертикальної перешкоди над плоским дном.

Розглядається нескінченно глибока течія над плоским дном, яка обтікає перешкоду висотою `h`. Величина швидкості руху рідини на нескінченності V_∞ задана. Оскільки зона течії однозв'язна і нескінченна, то задача відноситься до типу 1 (див. п. 8.5). Наближеною моделлю є поле течії ідеальної рідини у верхній півплощині, що обтікає перпендикулярний до дна відрізок висотою `h` (область *D*).



Отже межа *D* складається з дійсної осі комплексної площини `z` та відрізка $[a, a + ih]$, і повинна бути лінією течії. Без обмежень можна покласти, що на цій лінії $\psi = 0$ і в D $\psi > 0$. Для відшукання комплексного потенціалу $w = \Phi(z) = \varphi(x, y) + i \cdot \psi(x, y)$ достатньо знайти конформне відображення області течії *D* на верхню півплощину $\operatorname{Im} w = \psi > 0$ площини *w*. Функція $f(z)$, яка реалізує схоже відображення, має вигляд $w = f(z) = \sqrt{(z-a)^2 + h^2} + a$ [2, с. 142] і в формулі $\Phi(z) = k \cdot f(z)$ залишилося підібрати сталу *k*.

Оскільки $\dfrac{d\,f}{d\,z}=\dfrac{z-a}{\sqrt{(z-a)^2+h^2}}$, то $\left|\dfrac{d\,f}{d\,z}\right|_{z=\infty}=\lim\limits_{z\to\infty}\left|\dfrac{z-a}{\sqrt{(z-a)^2+h^2}}\right|=1$. Але, відповідно до ф. (4) п. 8.5, $\boldsymbol{V}=\overline{\Phi'(z)}$. Тому $V_\infty=|\boldsymbol{V}|=|\Phi'(\infty)|=k\cdot|f'(\infty)|=k$ і шуканий комплексний потенціал набуває вигляду

$$\Phi(z)=V_\infty\cdot\left(\sqrt{(z-a)^2+h^2}+a\right). \qquad (1)$$

Таким чином, дослідження течії зводиться до побудови в області D прообразів ліній $\psi=Const$, $\varphi=Const$ при відображенні (1), де a,h дійсні числа.

Зауваження. В точці $z_1=a+i\cdot h$ маємо $|\boldsymbol{V}(z_1)|=\left|\overline{\Phi'(z_1)}\right|=\infty$. Тобто в верхівці перешкоди швидкість рідини нескінченна. А в точках $z=a$ перед і після перешкоди (в вершинах прямих кутів) швидкість $|\boldsymbol{V}(a)|=\left|\overline{\Phi'(a)}\right|=0$ нульова.

Приклад. Обтікання перешкоди висотою h над плоским дном.

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Вимкніть перевірку розмірностей фізичних величин.

Untitled.mph (root): `Unit System` –<> = None

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `a=1, H=1, Vinf=2`.

В формулі (1) використовується функція квадратного кореня, яка застосовується до комплексних чисел. В COMSOL вважається, що аргумент $\varphi=\arg z$ комплексного числа z змінюється в діапазоні $-\pi<\arg z<\pi$ і обчислення квадратного кореня за звичайною формулою $w=\sqrt{z}=\sqrt{|z|}\,e^{i\varphi/2}$ повертає комплексні числа w з аргументом $-\pi/2\le\arg w\le\pi/2$. Наприклад, якщо аргумент φ (в тлумаченнях комплексного аналізу) дорівнює $\varphi=\pi+\alpha$ $(0<\alpha<\pi)$, то в якості квадратного кореня $\sqrt{z}=\sqrt{r\,e^{i\varphi}}$ система повертає число $z_1=\sqrt{r}\,e^{i(-\pi+\alpha)/2}$. Але в теорії функцій комплексного змінного зазвичай вважається, що $0\le\arg z<2\pi$ і для $\varphi=\pi+\alpha>\pi$ $(0<\alpha<\pi)$, повинно повертатися $z_2=\sqrt{r}\,e^{i(\pi+\alpha)/2}$, тобто $z_1=-z_2$.

Щоб застосовувати формули комплексного аналізу (зокрема формулу (1)), які містять квадратний корінь, потрібно скорегувати функцію квадратного кореня за наступною формулою:

$$\mathrm{Sqrt}\,z=\begin{cases}\mathrm{sqrt}\,z\,,\ \mathrm{Im}\,z>=0\\ -\mathrm{sqrt}\,z\,,\ \mathrm{Im}\,z<0\end{cases},$$

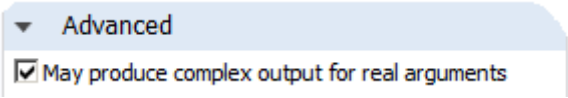
де `sqrt(z)` вбудована функція квадратного кореня.

Створіть «правильну» функцію квадратного кореня.

Global Definitions: RMB > Functions – Analytic

Function name = `Sqrt`

Definition – **Expression** = `if(imag(z)>=0,sqrt(z),-sqrt(z))`
– **Arguments** = `z`
Advanced – **May produce complex output for real arguments** = `True`



Без встановлення останньої опції не працюватиме, наприклад, вираз `Sqrt(-1)`.

Створена функція `Sqrt(z)` працюватиме так, як вважається в теорії функцій комплексного змінного, зменшуючи аргумент результівного комплексного числа в два рази.

Створіть в моделі двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Розрахункову геометрію створіть у вигляді прямокутника, в якому пізніше будуть креслитися лінії течії (вертикальний відрізок малювати не обов'язково).

`Geometry` **->** **Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1 (r1):

Size – **Width, Height** = `6,3`
Position – **Base, x, y** = `Corner,-3,0`

LMB > Build All Objects

Створіть в компоненті комплексну змінну `z=x+i*y` , і за формулою (1) змінну комплексного потенціалу w .

`Definitions` **->** **Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = `z, x+i*y`
Variables – **Name, Expression** = `w, Vinf*( Sqrt((z-a)^2+H^2)+a)`

Щоб графікам було на що посилатися (існувало джерело даних), створіть вузол досліджень і виконайте обчислення.

Untitled.mph (root): RMB > Add Study = `Add Study ->`
Preset Studies – Stationary **->** X

Study 1: LMB > Compute

Побудуйте лінії течії $\operatorname{Im} w = \psi = Const$.

`Results` **->** **Plot Group** **->** 2D Plot Group

2D Plot Group 1:

`2D Plot Group 1->` **Add Plot** **->** Contour

2D Plot Group 1 – Contour 1:

Expression – **Expression** = `imag(w)`
Levels – **Entry method**= `Levels`
Levels – **Levels** = `0.1 range(0.2,0.2,1)`
`range(1.25,0.25,2) range(2,0.4,6)`

Лінії за замовчуванням розфарбовуватимуться відповідно значенням потенціалу $\psi = \operatorname{Im} w$, а бажано – відповідно величині швидкості, яка дорівнює $|\boldsymbol{V}| = |\overline{\Phi'(z)}| = |\Phi'(z)|$. Щоб змінити вираз для фарбування у вузол **Contour 1** додайте підвузол `Color Expression`.

2D Plot Group 1 – Contour 1 : RMB > Color Expression

Нове забарвлення відповідатиме виразу, який задаватиметься в ПВ вузла **Color Expression 1**.

2D Plot Group 1 – Contour 1 – Color Expression 1 :

Expression – **Expression** = abs(d(w,z))

Оскільки величина швидкості $|V| = |dw/dz|$ в точці $z_1 = a + i \cdot h$ прямує до нескінченності, то автоматично обраний системою діапазон кольорів буде невдалим. Тому задайте цей діапазон «вручну».

Range – **Manual color range** = True

– **Minimum, Maximum** = 0,4

LMB > Plot (*наступний рисунок ліворуч*)

Створену картину течії можна доповнити еквіпотенціальними лініями $\operatorname{Re} w = \varphi = Const$.

2D Plot Group 1-> **Add Plot** -> Contour

2D Plot Group 1 – Contour 2:

Expression – **Expression** = real(w)

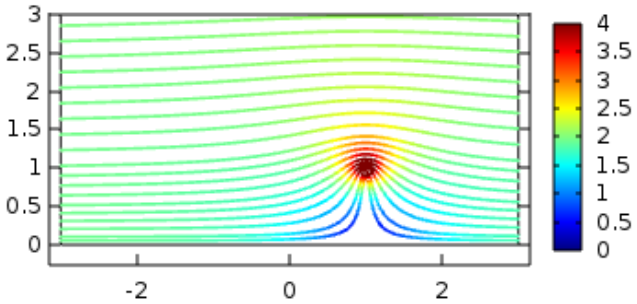
Levels – **Entry method**= Levels

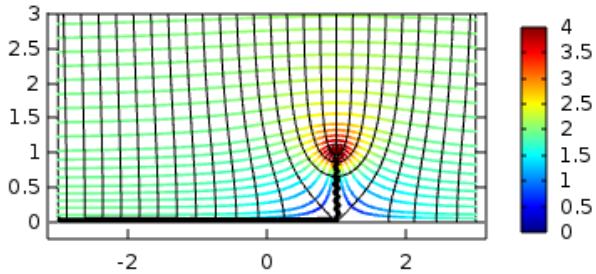
Levels – **Levels** = range(-6,0.5,7)

Coloring and Style – **Coloring, Color** = Uniform, Black

– **Color legend** = False

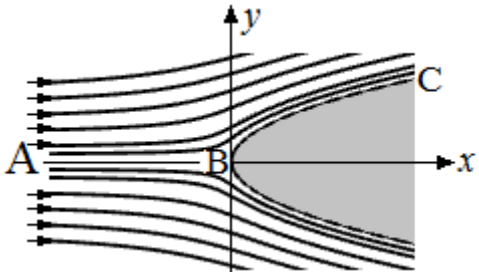
LMB>Plot (*наступний рисунок праворуч*)





8.5.2. Набігання потоку рідини на параболу.

Симетричний відносно осі O*x* потік ідеальної рідини набігає на параболу $y^2 = 2px$ зліва. Величина швидкості на нескінченності V_∞ відома. Потрібно побудувати картину течії.



З симетрії випливає, що від'ємна частина дійсної осі є лінією течії. Тому можна перейти до розглядання потоку, що обтікає контур ABC. Функція, яка виконує

конформне відображення $f(z)$ області D (частина півплощини $\operatorname{Im} z > 0$, обмеженої контуром ABC) на верхню півплощину має вигляд [7, с. 165]

$$f(z) = z - p - i \cdot \sqrt{2pz - p^2},$$

і в формулі для комплексного потенціалу $\Phi(z) = k \cdot f(z)$ залишилося підібрати сталу *k*. Далі маємо $d f / d z = 1 - i \cdot p / \sqrt{2pz - p^2}$ і $\left| d f / d z \right|_{z=\infty} = 1$. Відповідно до ф. (4) п. 8.5, $\boldsymbol{V} = \overline{\Phi'(z)}$. Тому $V_\infty = |\boldsymbol{V}| = |\Phi'(\infty)| = k \cdot |f'(\infty)| = k$ і шуканий комплексний потенціал набуває вигляду

$$\Phi(z) = V_\infty \cdot \left(z - p - i \cdot \sqrt{2pz - p^2} \right). \qquad (1)$$

Зауваження. З (1) випливає $d\Phi / d z \big|_{z=0} = V_\infty \cdot \left(1 - i \cdot p / \sqrt{2pz - p^2} \right)_{z=0} = 0$, тобто швидкість рідини в вершині параболи обертається на нуль.

Приклад. Набігання симетричного потоку рідини на параболу $y^2 = 2px$.

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Untitled.mph (root): Unit System – <> = None

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри p=0.5, Vinf=3.

Оскільки формула (1) містить корінь від комплексного виразу, то знадобиться «правильна» функція квадратного кореня (див. п. 8.5.1).

Global Definitions: RMB > Functions – Analytic

Function name = Sqrt

Definition – **Expression** = if(imag(z)>=0,sqrt(z),-sqrt(z))

– **Arguments** = z

Advanced – **May produce complex output for real arguments** = True

Додайте в модель двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Створіть геометрію у формі половини області течії (прямокутника, який імітуватиме верхню півплощину, за виключенням зони параболи).

Geometry -> **Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1 (r1):

Size – **Width, Height** = 10,3

Position – **Base, x, y** = Corner,-5,0

Накресліть параболу у вигляді параметричної кривої.

Geometry->**Primitives->** Parametric Curve

Geometry 1 - Parametric Curve 1 (pc1) :

Parameter – **Name, Minimum, Maximum** = s,0, 5

Expressions – **x, y** = s, sqrt(2*p*s) (*дуга параболи*)

LMB > Build Selected (*наступний рисунок ліворуч*)

Наступним кроком буде видалення внутрішньої зони параболи. Для цього знадобиться інструмент «розбиття об’єктів», який створює `Partition Objects` вузол. Він реалізує розділення геометричних об'єктів на окремі частини. В цьому вузлі вказується розділювана фігура і криві (або поверхні в 3D випадку), які застосовуються в якості розрізу. Результатом є частини первинного об’єкта, що в поєднанні являють початковий об’єкт.

`Geometry` **-> Booleans and Partitions** – Partition Objects

Partition Objects 1 (par1):

Partition Objects – **Objects to partition** = `r1` (*прямокутник*)

– **Tool objects** = `pc1` (*дуга параболи*)

LMB > Build Selected

Після застосування операції Partition Objects, верхня частина прямокутника буде відокремлена від нижньої. Це відразу не зрозуміло, але на наступному кроці в операції видалення частини прямокутника будуть вже розглядатися як різні фігури.

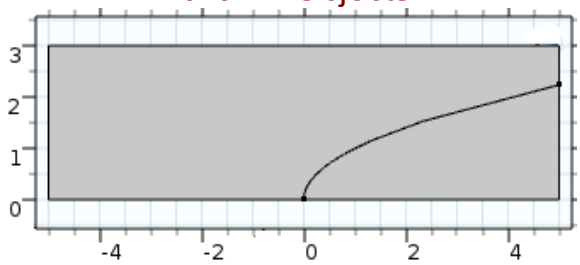
`Geometry` **-> Operations** – Delete

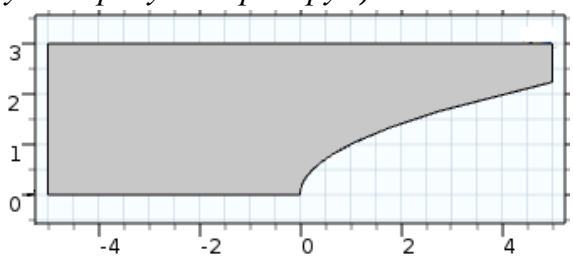
Delete Entities 1 (del1):

Entities or Objects to Delete – **Geometric entity level** = `Domain`

– **Selectin :** = `par1-2`

LMB > Build All Objects (*наступний рисунок праворуч*)





Задайте в компоненті комплексну змінну `z=x+i*y` , і за формулою (1) - змінну комплексного потенціалу w .

`Definitions` **-> Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = `z, x+i*y`

Variables – **Name, Expression**= `w,Vinf*(z-p-i*Sqrt(2*p*z-p^2))`

Створіть вузол досліджень і виконайте його обчислення.

Untitled.mph (root): RMB > Add Study = `Add Study ->`

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Оскільки задача обчислювалася лише у верхній половині зони течії, то розв’язок потрібно симетрично продовжити на нижню половину. Для цього призначене джерело даних Mirror 2D. Воно поширює розв’язок з деякої зони на область, яка отримується об’єднанням входової області з її віддзеркаленою відносно вісі симетрії копією. Віддзеркалюються також всі обчислені змінні.

`Results` -> Data Set - More Data Sets -> Mirror 2D

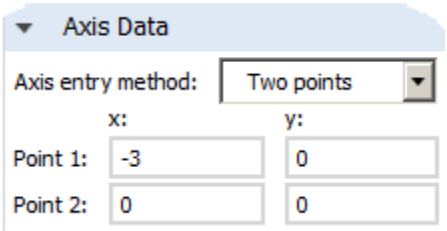
Data Sets - Mirror 2D 1 :

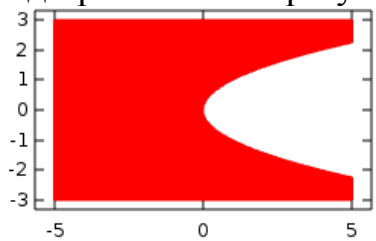
Axis Data – Point 1 – **x, y** = `-3,0`

Axis Data – Point 2 – **x, y** = 0,0

Тут точки Point 1 і Point 2 визначать вісь дзеркального відображення.

Виконайте команду Plot і отримайте зображення зони нового джерела даних, яке складається з об'єднання входової області та її дзеркального образу.





Тепер це джерело даних можна використовувати для побудови ліній течії $\operatorname{Im} w = \psi = Const$.

Results **-> Plot Group ->** 2D Plot Group

2D Plot Group 1:

Data – **Data set** = Mirror 2D 1

2D Plot Group 1-> **Add Plot ->** Contour

2D Plot Group 1 – Contour 1:

Expression – **Expression** = imag(w)

Levels – **Entry method**= Levels

Levels – **Levels** = 0 0.25 range(0.5,0.5,7)

Щоб розфарбувати лінії течії відповідно величині швидкості, додайте до вузла **Contour 1** підвузол Color Expression. В якості виразу для нового забарвлення використайте величину швидкості $|\boldsymbol{V}| = |\Phi'(z)|$.

2D Plot Group 1 – Contour 1 : RMB > Color Expression

2D Plot Group 1 – Contour 1 – Color Expression 1 :

Expression – **Expression** = abs(d(w,z))

Ми знаємо, що величина швидкості рідини змінюється в діапазоні від 0 у вершині параболи до $V_\infty = 3$. Задайте цей діапазон для палітри кольорів.

Range – **Manual color range** = True

– **Minimum, Maximum** = 0,3

LMB > Plot (*наступний рисунок ліворуч*)

Створену картину течії можна доповнити еквіпотенціальними лініями $\operatorname{Re} w = \varphi = Const$.

2D Plot Group 1-> **Add Plot ->** Contour

2D Plot Group 1 – Contour 2:

Expression – **Expression** = real(w)

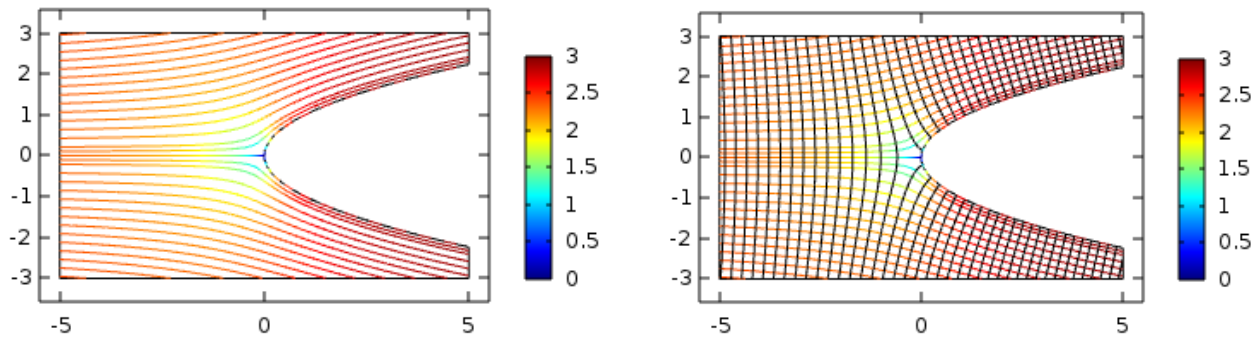
Levels – **Entry method**= Number of levels

Levels – **Levels** = 40

Coloring and Style – **Coloring, Color** = Uniform, Black

– **Color legend** = False

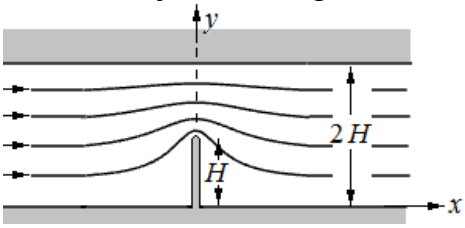
LMB>Plot (*наступний рисунок праворуч*)



З рисунків і палітри кольорів зрозуміло, що в більшій частині області течії величина швидкості рідини близька до $V_\infty = 3$. Суттєва зміна спостерігається в околі вершини параболи, де швидкість невелика (менша за одиницю).

8.5.3. Течія ідеальної рідини в каналі з перешкодою.

Дослідити картину течії ідеальної рідини в каналі з паралельними берегами завширшки $2H$, в якому є перешкода у вигляді стінки, перпендикулярної до берегів каналу і висотою в половину його ширини. Витрати Q рідини задано.



Задача зводиться до побудови конформного відображення області D (смуга з вертикальнім вирізом) на смугу завширшки Q в площині $w = \varphi + i \cdot \psi$. Воно реалізується функцією [7, с. 157]

$$w = \Phi(z) = \frac{2Q}{\pi} \ln\left(\sqrt{2}\,\mathrm{ch}\frac{\pi z}{4H} + \sqrt{\mathrm{ch}\frac{\pi z}{2H}}\right). \qquad (1)$$

Розв'язання повторює попередні приклади з новим виразом (1) конформного відображення.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = None

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `Q, 100`

`Parameters` – **Name, Expression** = `H, pi`

Untitled.mph (root) : RMB > Add Component ->2D

`Geometry` -> `Primitives` – Rectangle

Component 1 - Geometry 1 - Rectangle1 (r1):

`Size` – **Width, Height** = `10,2*H`

`Position` – **Base, x, y** = `Corner,-5,0`

LMB > Build All Objects

Побудуйте в геометрії прямокутну СкЕ сітку (це не обов'язково, але графік еквіпотенціальних ліній, який буде побудовано далі, виглядатиме краще).

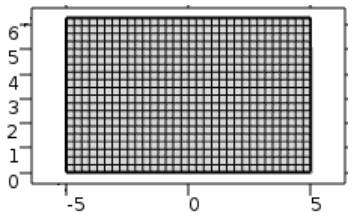
Mesh **->Generators** – Mapped

Component 2 - Mesh 1 – Size:

Element Size – **Custom**= True

Element Size Parameters – **Maximum element size** = H/10

LMB > Build All



Як і в попередньому прикладі для формули (1) потрібно створити «правильну» функцію квадратного кореня, яка зменшуватиме аргумент результівного комплексного числа в два рази.

Global Definitions: RMB > Functions – Analytic

Function name = Sqrt

Definition – **Expression** = if(imag(z)>=0,sqrt(z),-sqrt(z))

– **Arguments** = z

Advanced – **May produce complex output for real arguments** = True

Задайте в компоненті комплексну змінну z=x+i*y , допоміжну змінну zz, і за формулою (1) - змінну комплексного потенціалу w .

Definitions **->Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = z, x+i*y

Variables – **Name, Expression** = zz, pi*z/H/2

Variables – **Name, Expression** = w,

2*Q/pi*log(sqrt(2)*cosh(zz/2)+Sqrt(cosh(zz)))

Створіть вузол досліджень і виконайте обчислення.

Untitled.mph (root): RMB > Add Study = Add Study ->

Preset Studies – Stationary **->** X

Study 1: LMB > Compute

Побудуйте лінії течії $\operatorname{Im} w = \psi = Const$.

Results **->Plot Group ->** 2D Plot Group

2D Plot Group 1:

2D Plot Group 1-> **Add Plot ->** Contour

2D Plot Group 1 – Contour 1:

Expression – **Expression** = imag(w)

Levels – **Entry method**= Levels

Levels – **Levels** = range(0,Q/20,Q)

Щоб розфарбувати лінії течії відповідно величині швидкості, додайте до вузла **Contour 1** підвузол Color Expression і в якості виразу для забарвлення використайте величину швидкості $|\boldsymbol{V}| = |\Phi'(z)|$.

2D Plot Group 1 – Contour 1 : RMB > Color Expression

2D Plot Group 1 – Contour 1 – Color Expression 1 :

Expression – **Expression** = `abs(d(w,z))`

Очевидно, що величина швидкості рідини змінюється в діапазоні від 0 (у вершині прямих кутів) до нескінченності (у верхівки перешкоди). В таких випадках автоматично обраний системою діапазон кольорів буває невдалим. Задайте цей діапазон «вручну», підібравши верхню межу експериментально.

Range – **Manual color range** = `True`

– **Minimum, Maximum** = `0,40`

LMB>Plot (*наступний рисунок ліворуч*)

Картину течії доповніть еквіпотенціальними лініями $\operatorname{Re} w = \varphi = Const$.

`2D Plot Group 1->` **Add Plot ->** Contour

2D Plot Group 1 – Contour 2:

Expression – **Expression** = `real(w)`

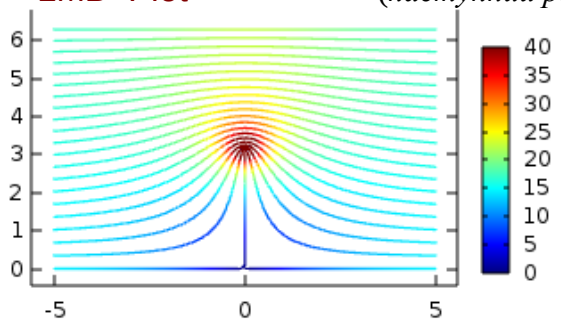
Levels – **Entry method**= `Levels`

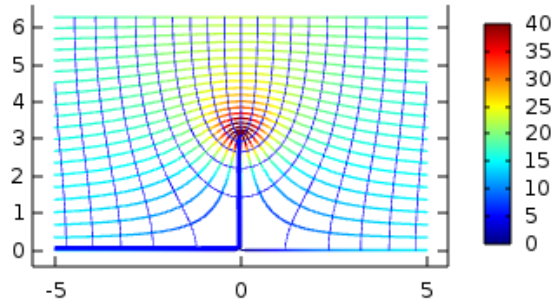
Levels – **Levels** = `range(-Q,Q/10,Q)`

Coloring and Style – **Coloring, Color** = `Uniform, Blue`

– **Color legend** = `False`

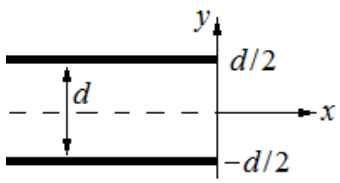
LMB>Plot (*наступний рисунок праворуч*)

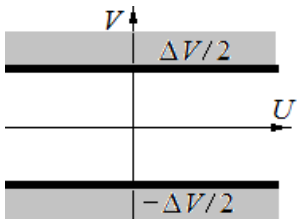




8.5.4. Електростатичне поле біля краю конденсатора.

Дослідити електростатичне поле конденсатора, утворюваного нескінченно тонкими металевими пластинами $y = -d/2$ і $y = d/2$, розташованими в області $x < 0$. Різниця потенціалів ΔV між пластинами задана.





Відомо [5, с. 390], що перетворення

$$z = f(w) = \frac{d}{2\pi}\left(w + e^{w}\right) \tag{1}$$

переводить смугу $|\operatorname{Im} w| < \pi$ площини w в площину $z = x + i \cdot y$ з двома розрізами $y = \pm d/2$, $x < 0$. Але в площині комплексного потенціалу $W = U(x, y) + i \cdot V(x, y)$ смуга повинна бути завширшки ΔV (різниця

потенціалів). Покладемо $W = \frac{\Delta V}{2\pi} w$. Тоді смуга $|\mathrm{Im}\, w| < \pi$ площини w відповідатиме смузі $|\mathrm{Im} W| < \Delta V / 2$ площини комплексного потенціалу W. Підставивши $w = 2\pi W / \Delta V$ у вираз (1) функції $f(w)$, отримаємо перетворення

$$z = \frac{d}{2\pi}\left(\frac{2\pi W}{\Delta V} + e^{\frac{2\pi W}{\Delta V}}\right), \qquad (2)$$

яке переводитиме смугу $|\mathrm{Im} W| < \Delta V / 2$ площини W в площину $z = x + i \cdot y$ з двома розрізами $y = \pm d / 2$, $x < 0$. Для створення картини поля потрібно побудувати образи ліній $V = \mathrm{Im} W = Const$, $U = \mathrm{Re}\, W = Const$ при відображенні (2). При цьому криві $V = V_i = Const$, де $-\Delta V / 2 < V_i < \Delta V / 2$, будуть представляти еквіпотенціальні лінії поля.

В прикладі візьміть $d = 2$, $\Delta V = 4$.

Приклад. Поле напівнескінченного плоского конденсатора.

Розв'язання. Створіть порожню модель з параметрами d і ΔV.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = None

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `d, 2`

`Parameters` – **Name, Expression** = `dV, 6`

Відповідно до формули (1) побудуйте функцію $f(z)$.

Global Definitions: RMB > Functions – Analytic

Function name = `F`

`Definition` – **Expression** = `d/(2*pi)*(w+exp(w))`

– **Arguments** = `w`

Створіть в моделі двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Змініть в ньому назви координат. Замість `x, y` введіть `U` і `V`.

Component 1:

`General` – **Spatial coordanates – First, Second** =`U,V`

Spatial coordinates

First (U)	Second (V)	Third (z)
U	V	z

Розрахункову геометрію створіть у вигляді прямокутника, який імітуватиме смугу $|V| \le \Delta V / 2$ на площині $W = U + i \cdot V$. Горизонтальні розміри прямокутника ми підбирали експериментально.

`Geometry` -> `Primitives` – Rectangle

Component 1 - Geometry 1 - Rectangle1 (r1):

`Size` – **Width, Height** = `18,dV`

`Position` – **Base, ϕ, ksi** = `Corner,-15,-dV/2`

LMB > Build All Objects

Побудуйте в геометрії прямокутну СкЕ сітку.

`Mesh` **->** **`Generators`** – Mapped

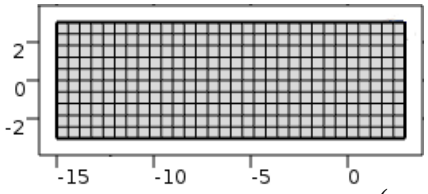
Задайте розміри клітинок сітки.

Component 2 - Mesh 1 – Size:

`Element Size` – **Custom**= `True`

`Element Size Parameters` – **Maximum element size** = `dV/20`

LMB > Build All



Створіть в компоненті змінні $W = U + i \cdot V$ і $Z = F(2\pi W / \Delta V)$.

`Definitions` **->** **`Variables`** – Local Variables

Definitions – Variables 1:

`Variables` – **Name, Expression** = `W, U+i*V`

`Variables` – **Name, Expression** = `Z, F(2*pi*W/dV)`

Створіть вузол досліджень і виконайте обчислення.

Untitled.mph (root): RMB > Add Study = `Add Study` ->

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Побудуйте в площині Z образ прямокутної сітки площини W при відображенні (2).

`Results` **->** **`Plot Group`** **->** 2D Plot Group

2D Plot Group 1:

`Plot Settings` – **Plot data set edges =** `False`

Щоб побудувати розфарбовану сітку можна використати `Surface` вузол з налаштуванням **Wireframe** = `True`.

`2D Plot Group 1`-> **`Add Plot`** **->** Surface

2D Plot Group 1 – Surface 1:

`Expression` – **Expression** = `imag(W)` (*вираз для забарвлення сітки*)

`Coloring and Style` – **Wireframe** = `True`

RMB > Deformation

2D Plot Group 2 – Surface 1 – Deformation 1:

`Expression` – **x component** = `real(Z)-U`

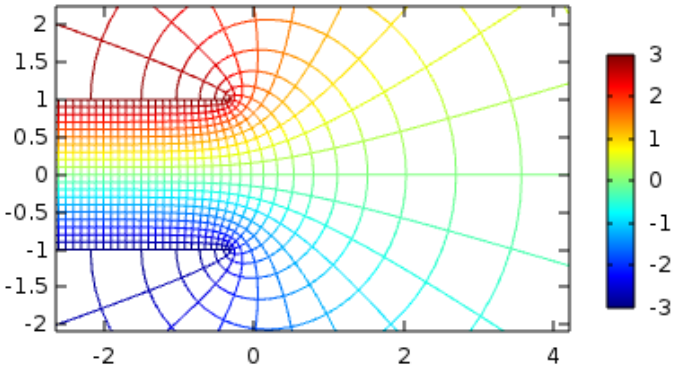
`Expression` – **y component** = `imag(Z)-V`

Якщо вузол **Deformation 1** не створювати, то буде зображатися декартова сітка. Використання вектора зсуву (`real(Z)-U,imag(Z)-V`) у вузлі **Deformation** переводить точки (U,V) площини W в точки з новими координатами $(U,V)+(\mathrm{Re}\,Z-U, \mathrm{Im}\,Z-V)=(\mathrm{Re}\,Z, \mathrm{Im}\,Z)$. Декартова сітка площини комплексного потенціалу $W = U + i \cdot V$ перетворюється на криволінійну ортогональну сітку, яка є зображенням силових і еквіпотенціальних ліній електростатичного поля.

`Scale`– **Scale factor** = `True,1`

LMB>Plot

Розмір видимої зони (масштабуванням і зсувом зображення за допомогою миші) підберіть так, щоб картина поля набула приблизно наступного вигляду.

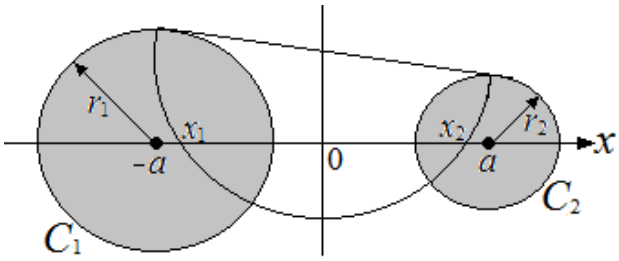


З рисунка видно, що внутрішність конденсатора розташована в зоні $-d/2<y<d/2$, $x<0$ площини Z. А палітра кольорів показує, що значення потенціалу $V=\mathrm{Im}W$ на еквіпотенціальних лініях змінюється в діапазоні $-3=-\Delta V/2<V<\Delta V/2=3$.

8.5.5. Зовнішнє електростатичне поле двох циліндрів.

Дослідити електростатичне поле двох заряджених паралельних кругових циліндрів, розташованих зовні один одного. Різниця потенціалів ΔV відома.

Проведемо вісь Ox через центри кіл, і початок координат розташуємо посередині. Відстань від центрів до початку координат позначимо через a, і радіуси кіл – r_1, r_2 $(r_1 \ge r_2, r_1+r_2<2a)$.



Задача зводиться до побудови поля в зовнішності D двох кіл C_1 і C_2. Для цього можна використати конформне відображення цієї зони на кільцеву область (задача типу 3 п. 8.5). Це робиться наступним чином. Накреслимо спільну дотичну до цих кіл і на ній, як на діаметрі, побудуємо півколо. Обчислимо абсциси точок перетину x_1, x_2 півкола з лінією центрів. Вони дорівнюють

$$x_{1,2}=\frac{1}{4a}\left(r_1^2-r_2^2\mp\sqrt{4a^2-(r_1+r_2)^2}\cdot\sqrt{4a^2-(r_1-r_2)^2}\right). \qquad (1)$$

Конформне відображення $w=f(z)$ області D (зовнішності кіл) на кільце в площині w матиме вигляд $f(z)=\dfrac{z-x_1}{z-x_2}$ [7, с. 159]. Відповідно до ф. (10) п. 8.5, комплексний потенціал $W=U+i\cdot V$ дорівнюватиме

$$W = k\,i \ln \frac{z - x_1}{z - x_2}. \qquad (2)$$

Узявши по точці на контурі кожного кола, наприклад, точки $z_1 = -a + r_1$ і $z_2 = a + r_2$, можна записати значення потенціалів контурів $V(z_i) = k \ln \left| \frac{z_i - x_1}{z_i - x_2} \right|$ $(i = 1, 2)$. Тоді різниця потенціалів $\Delta V = V(z_1) - V(z_2)$ дорівнюватиме

$$\Delta V = k \ln \left| \frac{(z_1 - x_1)(z_2 - x_2)}{(z_1 - x_2)(z_2 - x_1)} \right|. \qquad (3)$$

Останнє співвідношення визначатиме потрібне значення параметра *k*.

В прикладі візьміть $\Delta V = 2$, $a = 3$, $r_1 = 1$, $r_2 = 0.5$

Приклад. Поле двох кругових циліндрів, розташованих зовні один одного.

Р о з в ' я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = None

Задайте параметри задачі.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `dV,2` (*різниця потенціалів*)

`Parameters` – **Name, Expression** = `a, 3`

`Parameters` – **Name, Expression** = `r1, 1`

`Parameters` – **Name, Expression** = `r2,0.5`

Обчислення за формулою (1).

`Parameters` – **Name, Expression** = `x1, (r1^2-r2^2-sqrt(4*a^2-(r1+r2)^2)*sqrt(4*a^2-(r1-r2)^2))/(4*a)`

`Parameters` – **Name, Expression** = `x2, (r1^2-r2^2+sqrt(4*a^2-(r1+r2)^2)*sqrt(4*a^2-(r1-r2)^2))/(4*a)`

Обчислення параметра *k* з формули (3).

`Parameters` – **Name, Expression** = `k, dV/log(abs((-a+r1-x1)*(a+r2-x2)/(-a+r1-x2)/(a+r2-x1)))`

Потенціали V_1, V_2 контурів C_1 і C_2.

`Parameters` – **Name, Expression** = `V1, k*log(abs((-a+r1-x1)/(-a+r1-x2)))`

`Parameters` – **Name, Expression** = `V2, k*log(abs((a+r2-x1)/(a+r2-x2)))`

Крок між еквіпотенціальними лініями.

`Parameters` – **Name, Expression** = `dltV, (V1-V2)/20`

Створіть в моделі двовимірний компонент з геометрією, яка б імітувала площину *z*.

Untitled.mph (root) : RMB > Add Component ->2D

`Geometry` -> `Primitives` – Rectangle

Component 1 - Geometry 1 - Rectangle1 (r1):

`Size` – **Width, Height** = `16,8`

Position – **Base, x, y** = Corner,-8,-4

Geometry -> **Primitives** – Circle

Geometry 1 - Circle 1(c1):

Size and Shape – **Radius:** = r1

Position – **Base, x, y** = Center,-a,0

Geometry -> **Primitives** – Circle

Geometry 1 - Circle 2(c2):

Size and Shape – **Radius:** = r2

Position – **Base, x, y** = Center,a,0

Geometry -> **Booleans and Partitions** – Difference

Geometry 1 - Difference 1 (dif1):

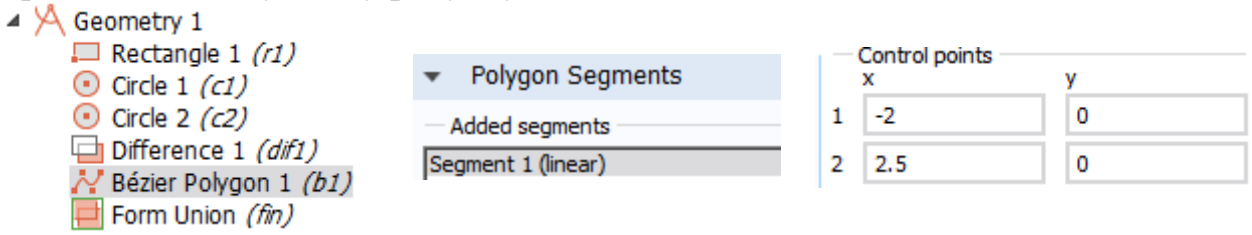
Difference – **Object to add** = r1 (*прямокутник*)

Union – **Object to subtract** = c1,c2 (*обидва кола*)

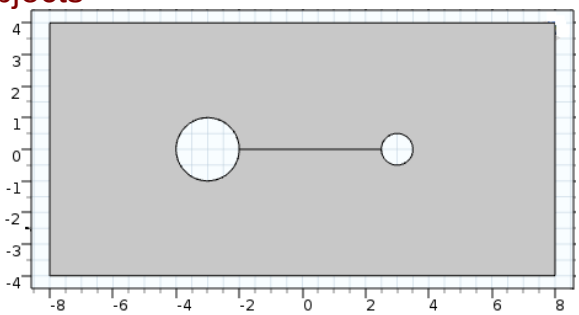
Щоб вузли СкЕ сітки розташовувалися на осі між колами, в геометрію додайте допоміжний відрізок. Для цього при активному вузлі **Geometry 1** виконайте команду

Geometry -> **Draw** – Line

Наведіть курсор миші в графічне вікно і намалюйте відрізок, який з'єднує праву точку лівого кола з лівою точкою правого. Якщо потрібно, в ПВ створеного вузла **Bézier Polygon 1(b1)** скорегуйте параметри відрізка так, як зображено на наступному рисунку.



LMB > Build All Objects



Генеруйте СкЕ сітку.

Mesh 1: **Mesh Settings** –**Element size** = Extremely fine

LMB > Build All

Задайте в компоненті комплексну змінну z=x+i*y і за формулою (2) - змінну *W* (комплексний потенціал).

Definitions -> **Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = z, x+i*y

Variables – **Name, Expression** = W, k*i*log((z-x1)/(z-x2))

Створіть вузол досліджень і виконайте обчислення.

Untitled.mph (root): RMB > Add Study = `Add Study ->`
Preset Studies – Stationary -> X

Study 1: LMB > Compute

Побудуйте картину електростатичного поля в площині z, тобто еквіпотенціальні $\mathrm{Im}W = Const$ і силові $\mathrm{Re}W = Const$ лінії поля.

`Results` -> **`Plot Group`** -> 2D Plot Group

2D Plot Group 1:

`2D Plot Group 1`-> **`Add Plot`** -> Contour

2D Plot Group 1 – Contour 1:

`Expression` – **Expression** = `imag(W)`

`Levels` – **Entry method**= `Levels`

`Levels` – **Levels** = `range(V2,dltV,V1)`

LMB>Plot

`2D Plot Group 1`-> **`Add Plot`** -> Contour

2D Plot Group 1 – Contour 2:

`Expression` – **Expression** = `real(W)`

`Levels` – **Entry method**= `Number of levels`

`Levels` – **Levels** = `30`

`Coloring and Style` – **Coloring, Color** = `Uniform, Black`
– **Color legend** = `False`

LMB>Plot (*наступний рисунок ліворуч*)

Додайте на рисунок зображення векторного поля - напруженості електричного поля $\boldsymbol{E} = E_x + i \cdot E_y = -i \cdot \overline{F'(z)}$ (див. ф. (3) п. 8.5). Для цього потрібно створити комплексну змінну E і оновити обчислення.

Definitions – Variables 1:

`Variables` – **Name, Expression** = `E, -i*conj(d(W,z))`

Study 1: RMB > Update Solution

В гілку **2D Plot Group 1** додайте графічний вузол `Arrow Surface`.

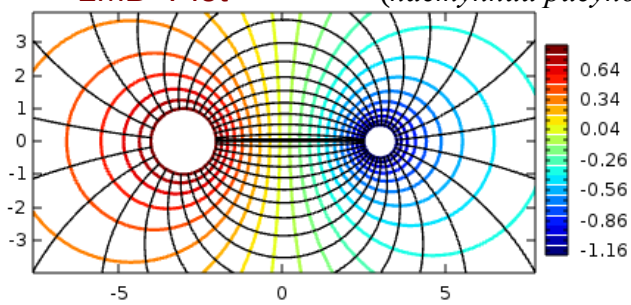
`2D Plot Group 1`-> **`Add Plot`** -> Arrow Surface

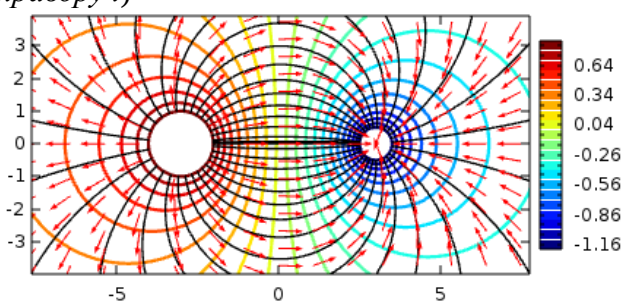
2D Plot Group 1 – Arrow Surface 1:

`Expression` – **x component** = `real(E)`
– **y component** = `imag(E)`

`Coloring and Style` – **Arrow length** = `Normalized`
– **Scale factor** = `True, 1`

LMB>Plot (*наступний рисунок праворуч*)





З палітри праворуч від картини поля видно діапазон зміни потенціалу V. Безпосередньо значення потенціалів V_1, V_2 контурів C_1 і C_2 зберігається в однойменних параметрах моделі. Обчисліть їх різницю.

`Results` **`->`** **`Derived Values`** – Global Evaluation

Derived Values - Global Evaluation 1:

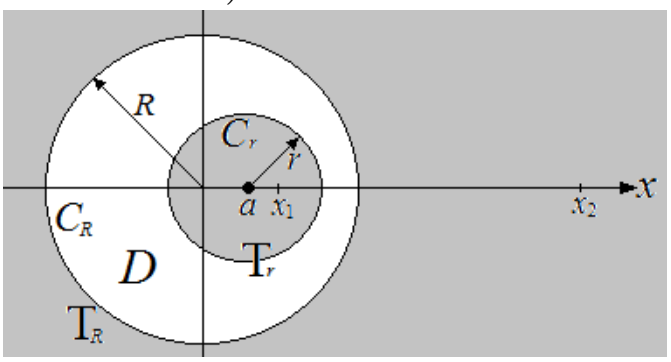
`Expression` – **Expression** = `V1-V2`

LMB > Evaluate

Буде отримано `2.000`, тобто обране в умові значення ΔV.

8.5.6. Теплове поле між двома ексцентричними трубами.

Дослідити розподіл температури між двома ексцентричними циліндричними трубами радіусів r і R $(r < R)$, нагрітими до температур T_r і T_R. Відстань між центрами труб a $(a < r, a + r < R)$.



Нехай вісь Ox проходить крізь центри кіл C_R, C_r і початок координат знаходиться в центрі більшого. Щоб побудувати конформне відображення області D, розташованій між контурами труб, спочатку потрібно на осі Ox знайти пару точок, симетричних відносно обох окружностей. Якщо абсциси цих точок позначити через x_1 і x_2, то вони задовольнятимуть системі рівнянь

$$\begin{cases} x_1 \cdot x_2 = R^2 \\ (x_1 - a)\cdot(x_2 - a) = r^2 \end{cases}$$

з розв'язком

$$x_{1,2} = \frac{a^2 - r^2 + R^2 \mp \sqrt{-4a^2R^2 + (a^2 - r^2 + R^2)^2}}{2a} \qquad (1)$$

Відомо [7, с. 161], що дрібно-лінійне перетворення $w = \dfrac{z - x_1}{z - x_2}$ конформно відображає область D між ексцентричними колами на кільце між концентричними колами. Тоді, відповідно до ф. (10) п. 8.5, комплексний потенціал W теплового поля з точністю до довільної сталої, яку нам зручно позначити через iC, матиме вигляд

$$W = U + i \cdot V = k\,i \ln \frac{z - x_1}{z - x_2} + i \cdot C. \qquad (2)$$

Враховуючи, що $\ln Z = \ln|Z| + i \cdot \arg Z$, для температури V (потенціальна функція) отримуємо

$$V = \mathrm{Im}W = k \ln\left|\frac{z - x_1}{z - x_2}\right| + C.$$

Підбираючи сталі k і C, можна досягти заданих значень температури на контурах C_R і C_r. Для цього потрібно підставити у вираз $V(z)$ значення температури на одному і іншому контурі, і розв'язати отримувану систему. Наприклад, обираючи по одній точці на кожному контурі C_r, C_R, і враховуючи, що $V = \mathrm{Im}W$, матимемо наступну систему алгебраїчних рівнянь

$$\mathrm{Im}W(a - r) = T_r,\ \mathrm{Im}W(R) = T_R, \tag{3}$$

Знайшовши звідси k і C, і підставивши їх в (2), отримаємо остаточний вираз для комплексного потенціалу.

Приклад. Теплове поле між двома ексцентричними циліндричними трубами. В модельній задачі покладіть `R=2, a=0.5, r=1, Tr=0, TR=1`.

Розв'язання. Оскільки нам доведеться розв'язувати систему алгебраїчних рівнянь при пошуку значень k і C, то можна відразу створити двовимірну модель з `Global ODEs and DAEs` інтерфейсом.

File -> New -> Model Wizard -> 2D -> Mathematics -
ODE and DAE Interfaces - Global ODEs and DAEs (ge) -> Stationary

Untitled.mph (root): `Unit System` – <> = None

Задайте параметри задачі.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `a,0.5`
– **Name, Expression** = `R,2`
– **Name, Expression** = `r,1`
– **Name, Expression** = `TR,1`
– **Name, Expression** = `Tr,0`

Абсциси `x1` і `x2` обчислюються за формулою (1).

– **Name, Expression** = `b, a^2-r^2+R^2`
– **Name, Expression** = `x1,`
`(b-sqrt(b^2-4*a^2*R^2))/(2*a)`
– **Name, Expression** = `x2,`
`(b+sqrt(b^2-4*a^2*R^2))/(2*a)`

В подальшій роботі ми будемо використовувати функцію комплексного логарифма. Вбудована функція `log(z)` працює коректно з комплексними аргументами, але «не бажає» обчислювати логарифми від'ємних дійсних чисел. Щоб подолати цю складність, створіть функцію користувача, яка вміє це робити.

Global Definitions: RMB > Functions – Analytic

Function name = `Log`

`Definition` – **Expression** = `log(z)`

– **Arguments** = z

Advanced – **May produce complex output for real arguments** = True

Функція Log(z) співпадає з вбудованою функцією log(z), але завдяки останній опції, буде працювати і з дійсними від'ємними аргументами

Створіть в моделі геометрію у вигляді області *D*.

Geometry -> **Primitives** – Circle

Geometry 1 - Circle 1 (c1):

Size and Shape – **Radius:** = R

Position – **Base, x, y** = Center,0,0

Geometry -> **Primitives** – Circle

Geometry 1 - Circle 2 (c2):

Size and Shape – **Radius:** = r

Position – **Base, x, y** = Center,a,0

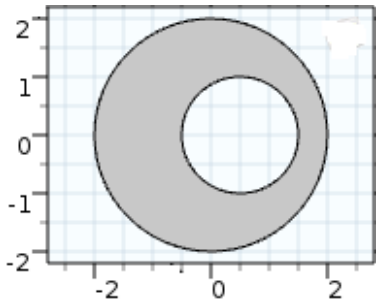
Geometry -> **Booleans and Partitions** – Difference

Geometry 1 - Difference 1 (dif1):

Difference – **Object to add** = c1 (*більше коло*)

Union – **Object to subtract** = c2 (*менше коло*)

LMB > Build All Objects



Генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Задайте в компоненті комплексну змінну z=x+i*y і за формулою (2) - змінну *W* комплексного потенціалу.

Definitions -> **Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = z, x+i*y

Variables – **Name, Expression** = W, k*i*Log((z-x1)/(z-x2))+i*C

Відносно невідомих *k* і *C* запишіть систему рівнянь (3). Для цього в ПВ вузла **Global Equations 1** заповніть таблицю наступним чином.

Global ODEs and DAEs(ge) - Global Equations 1:

Global Equations – **Name, f(u, ut, utt, t)** = k,imag(subst(W,z,a-r))-Tr

– **Name, f(u, ut, utt, t)** =C,imag(subst(W,z,R))-TR

Виконайте обчислення.

Study 1: LMB > Compute

В гілці **Results - Derived Values** з'являються два вузли **State variable k** і **State variable C** зі знайденими значеннями сталих *k* і *C*. Надрукуйте їх в спільну таблицю.

Derived Values – State variable k: LMB > Evaluate

Derived Values – State variable C: LMB > Evaluate - Table 1

State variable k	State variable C
1.6579	2.7685

Побудуйте картину теплового поля в області D. Для цього накресліть лінії сталого значення температури $V = \mathrm{Im}W = Const$ і лінії течії тепла $\mathrm{Re}\, W = Const$

`Results` **-> Plot Group ->** 2D Plot Group

2D Plot Group 1:

`2D Plot Group 1`-> **Add Plot ->** Contour

2D Plot Group 1 – Contour 1:

Expression – **Expression** = `imag(W)`

Levels – **Entry method**= `Levels`

Levels – **Levels** = `range(Tr,0.1,TR)`

LMB>Plot

2D Plot Group 1 – Contour 1: RMB > Duplicate

2D Plot Group 1 – Contour 2:

Expression – **Expression** = `real(W)`

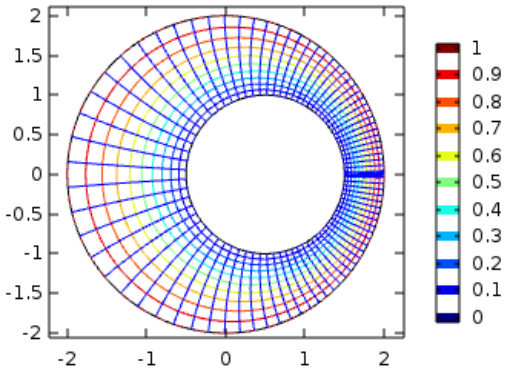
Levels – **Entry method**= `Number of levels`

– **Total levels** = `80`

Coloring and Style – **Coloring, Color** = `Uniform, Blue`

– **Color legend** = `False`

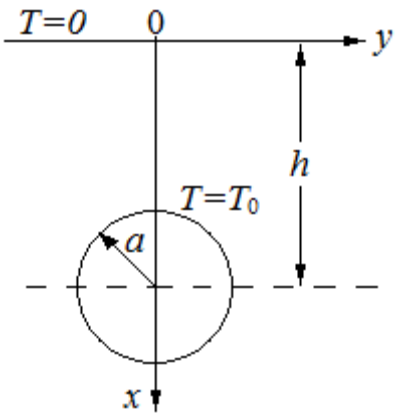
LMB>Plot

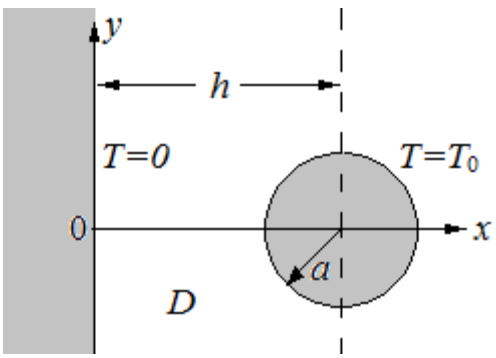


З палітри кольорів видно, що температура внутрішнього циліндра дорівнює 0 (тобто `Tr`), а зовнішнього – 1 (тобто `TR`), як і повинно бути.

8.5.7. Теплове поле навколо труби в грунті.

Труба радіуса a розташована на заданій глибині h $(h > a)$ в грунті. Знайти усталене розподілення температури, беручи температуру поверхні нульовою, а температуру труби T_0 (наступний рисунок ліворуч). Щоб не плутати координати, будемо моделювати задачу, схема якої зображена на наступному рисунку праворуч.





Покладемо $c=\sqrt{h^2-a^2}$. Відомо, що дрібно-лінійне перетворення $w=\dfrac{z-c}{z+c}$ відображає розглядувану область D (праву півплощину з круговим отвором) на кругове кільце [1, с. 40]. Тоді відповідно до ф. (10) п. 8.5, комплексний потенціал W теплового поля з точністю до довільної сталої, яку ми позначимо iC, матиме вигляд

$$W=U+i\cdot V=k\,i\,\ln\frac{z-c}{z+c}+i\cdot C. \qquad (1)$$

Враховуючи, що $\ln Z=\ln|Z|+i\cdot\arg Z$, температура (потенціальна функція V) дорівнюватиме

$$V=\operatorname{Im}W=k\ln\left|\frac{z-c}{z+c}\right|+C.$$

З умови $V(0)=0$ отримуємо C=0. З умови $V(h+a)=T_0$ (температура правої точки кола $z=h+a$) матимемо

$$T_0=k\cdot\ln\left|\frac{h+a-c}{h+a+c}\right|.$$

Останнє співвідношення визначає потрібне значення параметра k.

В модельній задачі покладіть $a=1$, $h=3$, $T_0=1$.

Приклад. Теплове поле в правій півплощині з круговим отвором.

Р о з в ’ я з а н н я . Створіть порожню модель.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = None

Задайте параметри задачі.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `H,3`

`Parameters` – **Name, Expression** = `a,1`

`Parameters` – **Name, Expression** = `T0,1`

`Parameters` – **Name, Expression** = `c, sqrt(H^2-a^2)`

Обчислення параметра k.

`Parameters` –**Name,Expression**=`k,T0/log(abs((H+a-c)/(H+a+c)))`

Створіть в моделі двовимірний компонент з геометрією, яка б імітувала площину *z*.

Untitled.mph (root) : RMB > Add Component ->2D

`Geometry` **-> Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1 (r1):

Size – **Width, Height** = `5,6`

Position – **Base, x, y** = `Corner,0,-3`

`Geometry` -> **Primitives** – Circle

Geometry 1 - Circle 1(c1):

Size and Shape – **Radius:** = `a`

Position – **Base, x, y** = `Center,H,0`

`Geometry` **-> Booleans and Partitions** – Difference

Geometry 1 - Difference 1 (dif1):

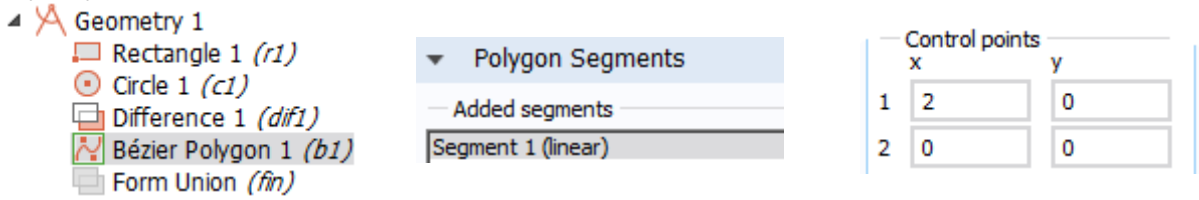
Difference – **Object to add** = `r1` (*прямокутник*)

Union – **Object to subtract** = `c1` (*коло*)

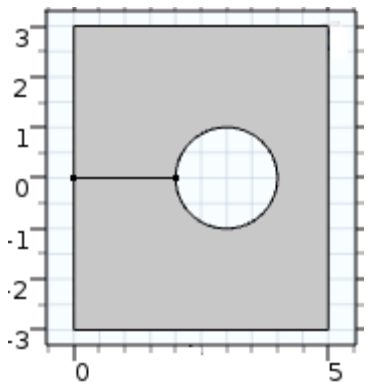
Щоб примусити вузли СкЕ сітки розташуватися на осі О*x*, в геометрію додайте допоміжний відрізок. Для цього при активному вузлі **Geometry 1** виконайте команду

`Geometry` **-> Draw** – Line

Наведіть курсор миші в графічне вікно і намалюйте відрізок, який з'єднує точку (0,0) з лівою точкою кола. Якщо потрібно, в ПВ створеного вузла **Bézier Polygon 1(b1)** скорегуйте параметри відрізка так, як зображено на наступному рисунку.



LMB > Build All Objects



Генеруйте СкЕ сітку.

Mesh 1: **Mesh Settings** –**Element size** = `Extremely fine`

LMB > Build All

Задайте в компоненті комплексну змінну `z=x+i*y` і відповідно до формули (1) змінну *W* (комплексний потенціал).

`Definitions` **-> Variables** – Local Variables

Definitions – Variables 1:

Variables – **Name, Expression** = `z, x+i*y`

Variables – **Name, Expression** = `W, k*i*log((z-c)/(z+c))`

Створіть вузол досліджень і виконайте обчислення.

Untitled.mph (root): RMB > Add Study = `Add Study ->`

Preset Studies – Stationary -> X

Study 1: LMB > Compute

Побудуйте картину теплового поля в площині z, тобто лінії сталого значення температури $\operatorname{Im} W = Const$ і лінії течії тепла $\operatorname{Re} W = Const$.

`Results` -> **Plot Group** -> 2D Plot Group

2D Plot Group 1:

`2D Plot Group 1`-> **Add Plot** -> Contour

2D Plot Group 1 – Contour 1:

Expression – **Expression** = `imag(W)`

Levels – **Entry method**= `Levels`

Levels – **Levels** = `range(0,0.05,T0)`

LMB>Plot

2D Plot Group 1 – Contour 1: RMB > Duplicate

2D Plot Group 1 – Contour 2:

Expression – **Expression** = `real(W)`

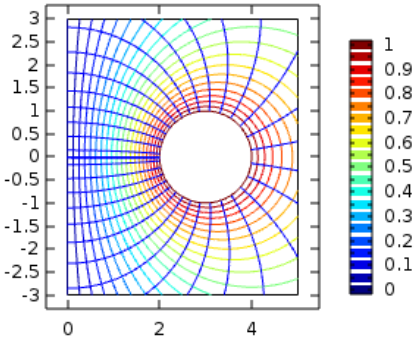
Levels – **Entry method**= `Number of levels`

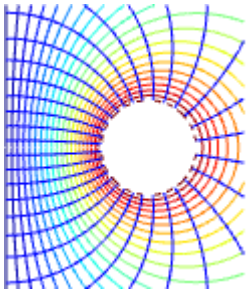
– **Total levels** = `30`

Coloring and Style – **Coloring, Color** = `Uniform, Blue`

– **Color legend** = `False`

LMB>Plot (*наступний рисунок ліворуч*)





Тут сині криві зображають лінії току тепла, а ексцентричні кола є лініями рівня температури, яка змінюється від 0 на поверхні півплощини до 1 (значення `T0`) на поверхні труби.

Вправа. Побудуйте графік теплового поля, як на попередньому рисунку праворуч, без стороннього зображення стовщеного відрізка $y = 0, 0 \le x \le 2$.

Вказівка. Щоб прибрати таку особливість, намалюйте в цьому місці геометрії тонкий прямокутник і видаліть його з розрахункової області.

8.5.8. Обтікання кола.

Дослідити задачу обтікання ідеальною рідиною кругового циліндра радіуса R з заданою швидкістю на нескінченності $V_\infty = 1$ і напрямком вздовж додатної дійсної осі (задача типу 4 п. 8.5). Якщо циркуляції рідини немає, то потік буде симетричним, і його можна розглядати як обтікаючим верхнє півколо і відповідні ділянки горизонтальної осі, тобто можна перейти до розглядання задачі типу 1 п. 8.5. В цьому випадку комплексним потенціалом поля є функція $\Phi(z)$, яка виконує конформне відображення відповідної області на верхню півплощину з додатковими умовами $\Phi(\infty) = \infty$ і $\Phi'(\infty) = 1$. Такою функцією є

$$w = \varphi + i \cdot \psi = \Phi(z) = z + \frac{R^2}{z},$$

і вона перетворює півколо $|z| = R$ у відрізок $-2R < \varphi < 2R$. Насправді функція $\Phi(z)$ виконує перетворення всієї зовнішності кола на зовнішність цього відрізка. Якщо потік буде несиметричним, то це означає існування вихра (зовні області D, тобто всередині кола) з циркуляцією Γ. Його комплексний потенціал дорівнюватиме $\frac{\Gamma}{2\pi i}\ln z$. Оскільки швидкість потоку вихра на нескінченності дорівнює нулю (швидкість пропорційна похідній комплексного потенціалу, див. ф. (4) п. 8.5), то потік з комплексним потенціалом

$$w = z + \frac{R^2}{z} + \frac{\Gamma}{2\pi i}\ln z$$

обтікатиме коло $|z| = R$, матиме задану циркуляцію Γ, і на нескінченності матиме одиничну швидкість. В залежності від величини циркуляції Γ (менше чи більше значення $4\pi|V_\infty|R$) можуть існувати різні картини течії.

У випадку негоризонтальної течії на нескінченності зі швидкість $\boldsymbol{V}_\infty$, комплексний потенціал матиме вид [2, с. 262]

$$\Phi(z) = \overline{\boldsymbol{V}_\infty} z + \frac{\boldsymbol{V}_\infty R^2}{z} + \frac{\Gamma}{2\pi i}\operatorname{Ln} z. \qquad (1)$$

Побудуйте лінії струменя такої течії.

Приклад. Обтікання ідеальною рідиною кругового циліндра.

Розв'язання. Створіть порожню модель.

File -> New -> Blank Model

Untitled.mph (root): `Unit System` – <> = None

Додайте в модель таблицю параметрів і заповніть її.

Global Definitions : RMB > Parameters

`Parameters` – **Name, Expression** = `R, 1`

`Parameters` – **Name, Expression** = `Vinf, 1`

`Parameters` – **Name, Expression** = `G1, 2*pi*abs(Vinf)*R`

`Parameters` – **Name, Expression** = `G2, 5*pi*abs(Vinf)*R`

Parameters

Name	Expression	Value
R	1	1
Vinf	1	1
G1	2*pi*abs(Vinf)*R	6.2832
G2	5*pi*abs(Vinf)*R	15.708

Параметри G1 і G2 будуть зображати два різні значення циркуляції, а вектор швидкості на нескінченності Vinf задано дійсним числом, тобто він горизонтальний.

Додайте двовимірний компонент.

Untitled.mph (root) : RMB > Add Component ->2D

Геометрію створіть у вигляді квадрата з круговим отвором.

Geometry **-> Primitives ->** Square

Size – **Size length** = 8

Position – **Base, x, y** = Center, 0, 0

Geometry -> **Primitives** – Circle

Size and Shape – **Radius:** = R

За допомогою операції логічного віднімання видаліть з області квадрата зону кола.

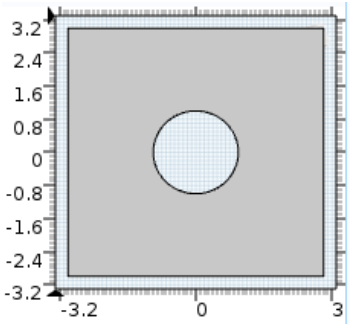
Geometry **-> Booleans and Partitions** – Difference

Component 1 - Geometry 1 - Difference 1:

Difference – **Object to add** = sq1 (*квадрат*)

– **Object to subtract** = c1 (*коло*)

LMB > Build All Objects



Активуйте вузол компонента **Component 1** і створіть таблицю змінних моделі.

Definitions -> **Variables** – Local Variables

Definitions - Variables 1: (*наступні введення в таблиці*)

Variables – **Name, Expression =** Z,x+i*y

– **Name, Expression =** W, conj(Vinf)*Z+Vinf*R^2/Z

Щоб змінні моделі працювали потрібно виконати обчислення, а для цього треба мати вузол Study.

Untitled.mph (root): RMB > Add Study -> Stationary

Home **->** Compute

Після завершення обчислень ніякі графічні вузли не створюються.

Побудуйте лінії струменя безциркуляційного потоку і ортогональну до них сім'ю ліній. Комплексний потенції цієї течії ми позначили змінною W.

Results **-> Plot Group ->** 2D Plot Group

2D Plot Group 1 **-> Add Plot** – Contour

2D Plot Group 1 – Contour 1: (*лінії струменя безциркуляційного потоку*)

Expression – **Expression** = imag(W)

Levels – **Total levels** = 30

Coloring and Style – **Contour type** = Tube

Coloring and Style – **Tube radius expression** = 0.03

Coloring and Style – **Radius scale factor** = True,1

Coloring and Style – **Coloring, Color** = Uniform, Red

Coloring and Style – **Color legend** = False

LMB>Plot (*наступний рисунок ліворуч*)

Додайте на рисунок еквіпотенціальні лінії.

2D Plot Group 1 – Contour 1: RMB > Duplicate

2D Plot Group 1 – Contour 2: (*ортогональні лінії до ліній струменя*)

Expression – **Expression** = real(W)

Coloring and Style – **Color** = Blue

LMB>Plot (*наступний рисунок всередині*)

Додайте на останній рисунок векторне поле швидкостей безциркуляційного потоку. Оскільки швидкість дорівнює $\boldsymbol{V} = \overline{\Phi'(z)}$ (див. ф. (4) п. 8.5), то для неї потрібно створити відповідну змінну.

Definitions - Variables 1:

Variables – **Name, Expression =** V, conj(d(W,Z))

Study 1: LMB > Compute

2D Plot Group 1-> **Add Plot** -> Arrow Surface

Expression – **x component** = real(V)

– **y component** = imag(V)

Arrow Positioning – **x grid cpoints - Points** = 10

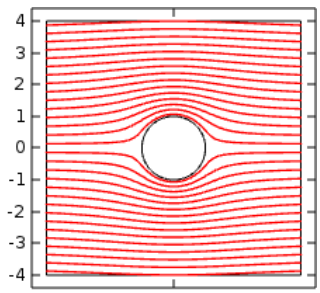
– **y grid cpoints - Points** = 15

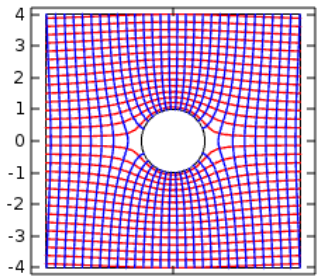
Coloring and Style – **Scale factor** = True, 1

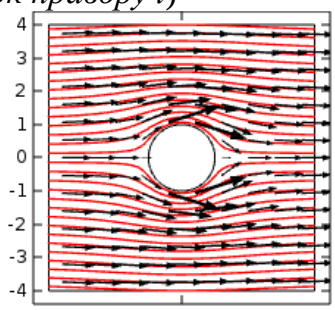
– **Color** = Black

2D Plot Group 1 – Contour 2: RMB > Disable

LMB>Plot (*наступний рисунок праворуч*)







Тепер побудуйте лінії струменя потоку з циркуляцією. В таблиці змінних за формулою (1) створіть два комплексних потенціали, які відповідатимуть двом різним значенням циркуляції G1 і G2, і відповідно двом різним картинам течії.

Definitions - Variables 1:

Variables – **Name, Expression =** WC1, W+G1/(2*pi*i)*log(Z)

Variables – **Name, Expression =** `WC2, W+G2/(2*pi*i)*log(Z)`

Study 1: LMB > Compute

Побудуйте лінії струменя потоку з невеликою циркуляцією `G1=2*pi*abs(Vinf)*R`. Його комплексний потенціал ми позначили змінною `WC1`.

`Results` **->** **Plot Group** **->** 2D Plot Group

`2D Plot Group 1` **->** **Add Plot** – Contour

2D Plot Group 2 – Contour 1:

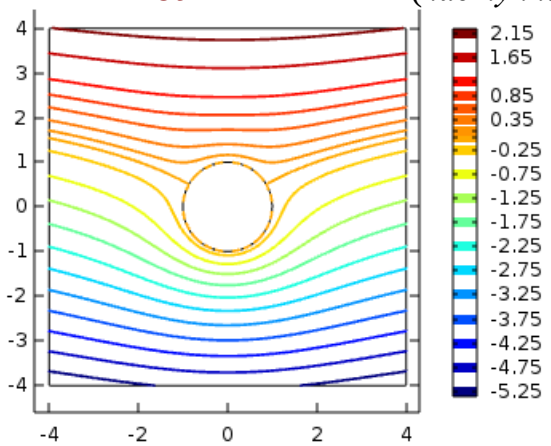
Expression – **Expression** = `imag(WC1)`

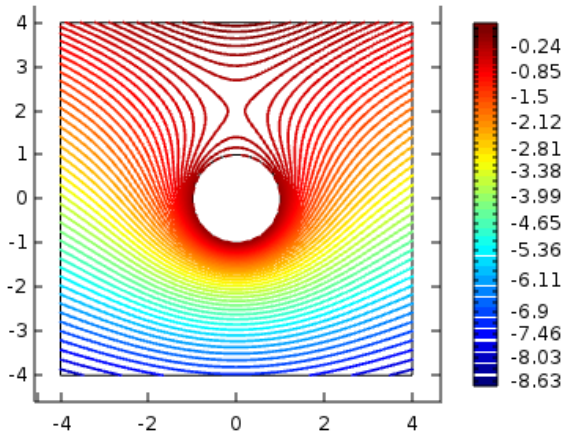
Levels – **Entry method**= `Levels`

Levels – **Levels** = `range(-5.25,0.5,-0.25) 0 0.15 0.35 0.6 0.85 range(1.15,0.5,2.5)`

Coloring and Style – **Color legend** = `True`

LMB>Plot (*наступний рисунок ліворуч*)





При циркуляції $\Gamma > 4\pi|\boldsymbol{V}_\infty|R$ навколо циліндра утворюються замкнені лінії струменя. Комплексний потенціал потоку при циркуляції `G2=5*pi*abs(Vinf)*R` , яка задовольняє вказаній нерівності, ми позначили змінною `WC2`.

2D Plot Group 2 : RMB > Duplicate

2D Plot Group 3 – Contour 1:

Expression – **Expression** = `imag(WC2)`

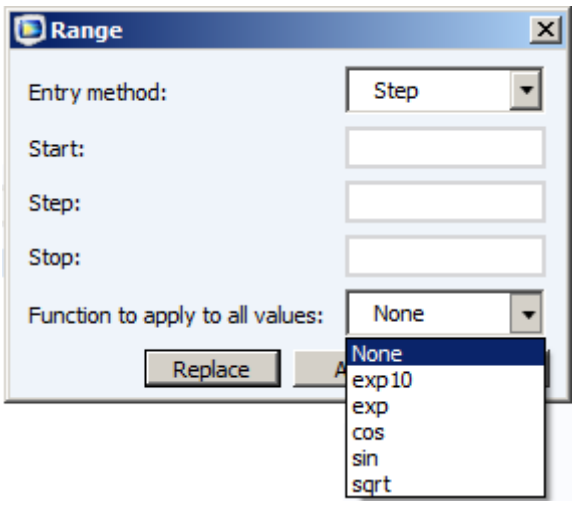
Levels – **Entry method**= `Levels`

Levels – **Levels** = `0.25-0.05*range(0,1,50)-(range(0,1,50)^2+range(0,1,50))/400`

LMB>Plot (*попередній рисунок праворуч*)

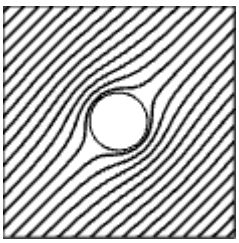
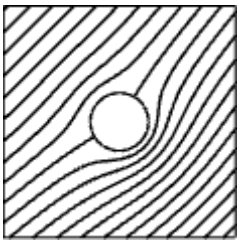
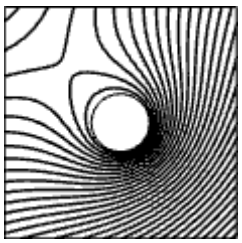
Як бачите, навколо кола зображено кілька замкнених ліній течії. Щоб їх побачити ми «наказали» системі відобразити лінії струменя $\operatorname{Im} WC2 = C_n$ при нерівномірно розподілених на інтервалі $-8.625 \le C_n \le 0.25$ значеннях константи C_n. Формула для обчислення цих значень має вигляд $C_n = 0.25 - 0.05 \cdot n - \dfrac{n^2+n}{400}$ $(n = 0,1,\ldots,50)$. Ми підібрали її експериментально. Щоб коректно задати цю формулу в полі списку **Levels**, замість змінної `n` потрібно було записати функцію `range(0,1,50)`.

Способи введення списків ми обговорювали в розділі «Математичні вирази» першої глави. В зв'язку з цим зробимо ще кілька зауважень. При створенні списків можна будувати деякі формули. Наприклад, можна використовувати формули `Const*range(0,1,5)`, `Const^range(0,1,5)` або `sin(range(0,0.2,1.0))`. Замість функції `sin` можна використовувати лише обмежену кількість функцій, назви яких можна побачити з випадного списку віконця `Range`.



Воно відкривається при клацанні мишею по кнопці Range, розташованій вгорі-праворуч від будь-якого поля діапазонів. Не можна використовувати функції користувача, або навіть множення різних діапазонів, наприклад, `range(0,1,5)* range(1,1,6)`, яке можна було б проінтерпретувати як $n\cdot(n+1)$ при $(n=0,1,\ldots,5)$, але n^2+n при $n=\mathrm{range}(0,1,5)$ працює.

Вправа. Побудуйте картини течії рідини навколо циліндра при векторі швидкості на нескінченності $\boldsymbol{V}_\infty=(1+i)/\sqrt{2}$ і значеннях циркуляції $\Gamma=0;\ \pi|\boldsymbol{V}_\infty|R;\ 6\pi|\boldsymbol{V}_\infty|R$. Відповіді зображено на наступних рисунках.

Література.

1. Лебедев Н.Н., Скальская И.П., Уфлянд Я.С. Сборник задач по математической физике. – М.: Гос. изд. технико-теоретической литературы, 1955. – 420 с.

2. Лаврентьев М.А., Шабат Б.В. Методы теории функций комплексного переменного. – М.: Наука, Главная редакция физико-математической литературы, 1973, 736 стр.

3. Волковыский Л.И., Лунц Г.Л., Араманович И.Г. Сборник задач по теории функций комплексного переменного. – М.: Гос. изд. физ.-мат. литературы, 1961. – 367 с.
4. Гурса Э. Курс математического анализа. Т. 2. – М. Л.: Главная редакция общетехнической литературы и номографии, 1936, 564 с.
5. Тихонов А.Н., Самарский А.А. Уравнения математической физики. – М.: Наука, 1977. – 736с.
6. Полянин А.Д. Справочник по линейным уравнениям математической физики. М.: ФИЗМАТЛИТ, 2001. – 576с.
7. Фукс Б. А. , Шабат Б. В. Функции комплексного переменного и некоторые их приложения. – 3-е изд. – М.: Наука, 1964, 388 стр.