Доля П.Г.
Харківський національний університет імені В. Н. Каразіна
факультет математики і інформатики
кафедра теоретичної і прикладної інформатики
2019 р.

Основи моделювання в COMSOL Multiphysics.

Зміст

1. Початок роботи в COMSOL Multiphysics. 3
1.1. Вступ. 3
1.2. Опис інтерфейсу на прикладі розв'язання рівняння Пуассона. 5
1.3. Функції і графіки. 19
1.4. Термінологія, позначення і скорочення. 38
1.5. Початкові відомості про моделювання в COMSOL Multiphysics. 45
1.5.1. Структура моделей COMSOL Multiphysics. 45
1.5.2. Керування деревом моделі 69
1.5.3. Налаштування графіки. 76
1.5.4. Параметри, змінні, зонди (probes). 98
1.5.5. Побудова геометрії. 119
1.6. Демонстраційні приклади. 130
1.6.1. Охолодження куба. 130
1.6.2. Коливання вільно опертої балки. 137
1.6.3. Осесиметричні коливання круглої мембрани. 142
1.7. Математичні вирази та оператори. 147
1.8. Довідкова система і документація 169
1.9. Задачі для самостійного розв'язання. 170
2. Візуалізація та обробка даних. 174
2.1. Основи графічного зображення даних. 174
2.1.1. Графіки функцій. 176
2.1.2. Графіки параметрично заданих кривих і поверхонь. 196
2.1.3. Використання сіткових джерел даних та їх трансформацій. 202
2.1.4. Траєкторії точок, лінії течії і анімація. 227
2.1.5. Допоміжні графіки. 245
2.2. Обчислення інтегралів, довжин, площ та об'ємів. 256
2.2.1. Обчислення інтегралів, середніх і екстремальних значень. 256
2.2.2. Геометричні застосування інтегрального числення. 268
2.3. Графічні ілюстрації до розв'язків прикладних задач. 283
2.3.1. Центр ваги тіла. 283
2.3.2. Дія зосередженої сили на пружну півплощину. 287
2.3.3. Напруження в нескінченній платівці з круглим отвором. 292
2.3.4. Прогин балки. 297

2.3.5. Дотична і нормаль до еліпса. 302
2.3.6. Траєкторія руху точки на ободі колеса. 307
2.3.7. Рух стичного кола. 311
2.3.8. Математичний більярд в прямокутнику. 315
2.3.9. Кривошипно-шатунний механізм. 319
2.3.10. Стрічка Мебіуса. 325
2.3.11. Коливання струни з закріпленими кінцями. 329
2.3.12. Одновимірна теплопровідність. 335
2.3.13. Остигання квадратної платівки і куба. 343
2.3.14. Хвиля Релея. 348
2.3.15. Біжні хвилі в водоймищі скінченної глибини. 352
2.3.16. Подовжні і крутні коливання. 357
Література. 368
3. Класичні задачі математичної фізики. 369
3.1. Крайова задача для ЗДР. 371
3.2. Вільні коливання напівнескінченної струни. 374
3.3. Вільні коливання вагомої нитки з вантажем на кінці. 378
3.4. Розподіл температури в кільці. 384
3.5. Власні коливання струни. 389
3.6. Телеграфне рівняння. 393
3.7. Вісесиметричні коливання нескінченної тремтівки. 396
3.8. Мінімальна поверхня. 401
3.9. Безвихрова течія ідеальної рідини. 408
3.10. Моделювання точкового джерела. 417
3.11. Скін-ефект. 422
3.12. Електростатичний потенціал квадратної рамки. 428
3.13. Магнітне поле електромотора. 432
3.14. Постійний струм в проводійному середовищі. 439
3.15. Дифракція плоскої хвилі на перешкоді. 445
3.16. Розповсюдження тепла в платівці з прямокутним отвором. 451
3.17. Теплопровідність в неоднорідному середовищі. 457
3.18. Власні коливання круглої тремтівки. 463
3.19. Розповсюдження тепла в радіоактивному стрижні. 466
3.20. Теплопровідність в шестикутній призмі з отвором. 471
3.21. Дві крайові задачі коливання квадратної мембрани. 477
3.22. Коливання струни з обчислювальною початковою формою. 485
3.23. Система двох ДРЧП. 491
3.24. Згин круглої плити, затисненої за контуром. 504
3.25. Система нелінійних параболічних рівнянь. 509
3.26. Рівняння Пуассона в генеральній формі. 512
3.27. Рівняння Кортевега – де Фріза. 516
3.28. Сферично-симетрична задача теплопровідності 522
Література. 529

Числове моделювання фізичних процесів є важливим етапом проектування конструкцій і технологій. Оскільки багато фізичних законів формулюється за допомогою диференціальних рівнянь в частинних похідних (ДРЧП), то для їх дослідження потрібен універсальний інструмент, який, крім іншого, здатний поєднувати різні фізичні моделі. Таке об'єднання можна назвати «мультифізикою», що означає спільне використання декількох моделей.

COMSOL Multiphysics це потужне програмне середовище, яке створено для моделювання фізичних процесів і здатне розв'язувати широкий спектр наукових і інженерних задач з багатьох сфер людської діяльності. При розв’язанні ДРЧП в системі здебільшого використовується метод скінченних елементів (МСкЕ). Система має налаштовані інтерфейси (прикладні режими) для розв’язання задач з різноманітних сфер, таких як механіка, електромагнетизм, гідродинаміка, хімія, акустика, теплопровідність, дифузія, оптика, опір матеріалів, теорія пружності і багатьох інших. COMSOL Multiphysics дозволяє поєднувати несхожі фізичні моделі і моделі різних розмірностей, а також дозволяє додавати до моделей власні рівняння. З самого початку, не використовуючи визначені інтерфейси, можна записати задачу в формі системи рівнянь, яка може включати алгебраїчні рівняння (АР), звичайні диференціальні рівняння (ЗДР) та ДРЧП. Можливості системи COMSOL Multiphysics величезні, і викласти їх в межах одного посібника нереально. Тут ми дамо вступ в розв’язання класичних задач. При цьому ми намагатимемося познайомити вас з системою на простих але нетривіальних прикладах,. Це полегшить знайомство з програмою.

Читати посібник ми радимо послідовно. Головні можливості системи викладаються в процесі розв’язання задач. Практично в кожному прикладі розглядаються нові елементи моделювання (опції, параметри, команди і т.і.). При цьому нові етапи моделювання викладаються докладно, але їх повторне використання в наступних задачах описується стисло. Не розібравши попередні приклади, ви, можливо, деякі з етапів згодом не зможете відтворити.

Якщо ви ще не вирішили, чи потрібно вам знайомитися з COMSOL Multiphysics, то подивіться приклади в наступних главах посібника. Ілюстрації до їх розв’язків надають уявлення про можливості системи.

1. Початок роботи в COMSOL Multiphysics.

1.1. Вступ.

В даному посібнику буде описуватися робота системи COMSOL Multiphysics версії 5.1, яка вийшла в 2015 році. До 2005 року вона називалася FEMLAB (Finite Element Method Laboratory). Історія програми починається з пакету-додатку PDE Toolbox MatLab, який призначений для розв’язання диференціальних рівнянь в частинних похідних і який досі входить до складу основної поставки MatLab. Друга версія так само, як і перша, не могла працювати без MatLab, але вже постачалася як самостійний продукт. Починаючи з третьої версії, FEMLAB міг працювати як окрема програма або

використовуватися як пакет додаток. Головним режимом роботі 4-ї та 5-ї версій є робота з графічним інтерфейсом користувача, але можливість спільної роботи з MatLab збереглася. При цьому синтаксис запису математичних виразів успадкований від першої версії і дуже близький до синтаксису MatLab.

Перша версія програми (PDE Toolbox MatLab) могла розв'язувати системи ДРЧП відносно функцій двох просторових змінних та змінної часу. Наступні версії могли вже розв'язувати тривимірні задачі. Пакет використовує метод скінченних елементів (МСкЕ). В англомовній літературі він називається Finite Element Method (FEM). В цьому методі розрахункова область розбивається на велику кількість маленьких ділянок, які називаються скінченними елементами. Потім будуються фінітні функції з носіями, обмеженими кількома такими елементами. Ці функції утворюють базис в просторі кускових функцій на скінченно-елементній сітці. Шукані величини апроксимуються лінійною комбінацією базисних функцій і крайова (гранична) задача замінюється задачею визначення коефіцієнтів розкладання апроксимуючих функцій по базисним.

На сайті розробників www.comsol.com/models розміщена галерея моделей, де можна докладно познайомитися з можливостями програми.

COMSOL Multiphysics вміє розв'язувати одновимірні (1D), двовимірні (2D) і тривимірні (3D) задачі. При цьому задачі можуть бути стаціонарними (шукані величини не залежать від часу), або динамічними (є залежність від часу). Інколи доводиться розглядати нульвимірні (0D) задачі, тобто розв'язувати просторово незалежні рівняння. Це означає, що задача зводиться до пошуку фізичних величин в точках. Чим вище розмірність задачі, тим складніше і триваліше розв'язання. У вісесиметричному випадку розмірність задачі можна понизити на одиницю. Для цього система виділяє випадки одновимірних і двовимірних осесиметричних задач (1D і 2D Axisymmetric). Таким чином, можна обирати один з шести можливих варіантів розмірності або їх поєднання.

Дуже важливою частиною програми є прикладні режими, які містять закінчені математичні постановки задач з різноманітних сфер фізики і техніки. Електронна версія документації по цим режимам встановлюється разом з COMSOL в двох форматах – PDF та HTML. Зокрема у встановлювальній папці системи в каталозі `..\doc\pdf\` містяться файли з документацією, яка включає опис математичних постановок задач, процес моделювання та отримання результатів.

Головним режимом роботи програми є робота з графічним інтерфейсом користувача (режим COMSOL Desktop). Але існують і інші режими, наприклад, робота в командному вікні (режим COMSOL Batch) без використання графічного інтерфейсу. В цьому режимі заощаджується пам'ять за рахунок того, що вона не використовується під графічний інтерфейс. Існує режим роботи з доступом до COMSOL Multiphysics з MatLab або, навпаки, з доступом з COMSOL Multiphysics до MatLab. Після встановлення системи, якщо до цього був встановлений MatLab, у вас на робочому столі з'явиться два ярлика. Перший «COMSOL Multiphysics 5.1» запускає систему з графічним

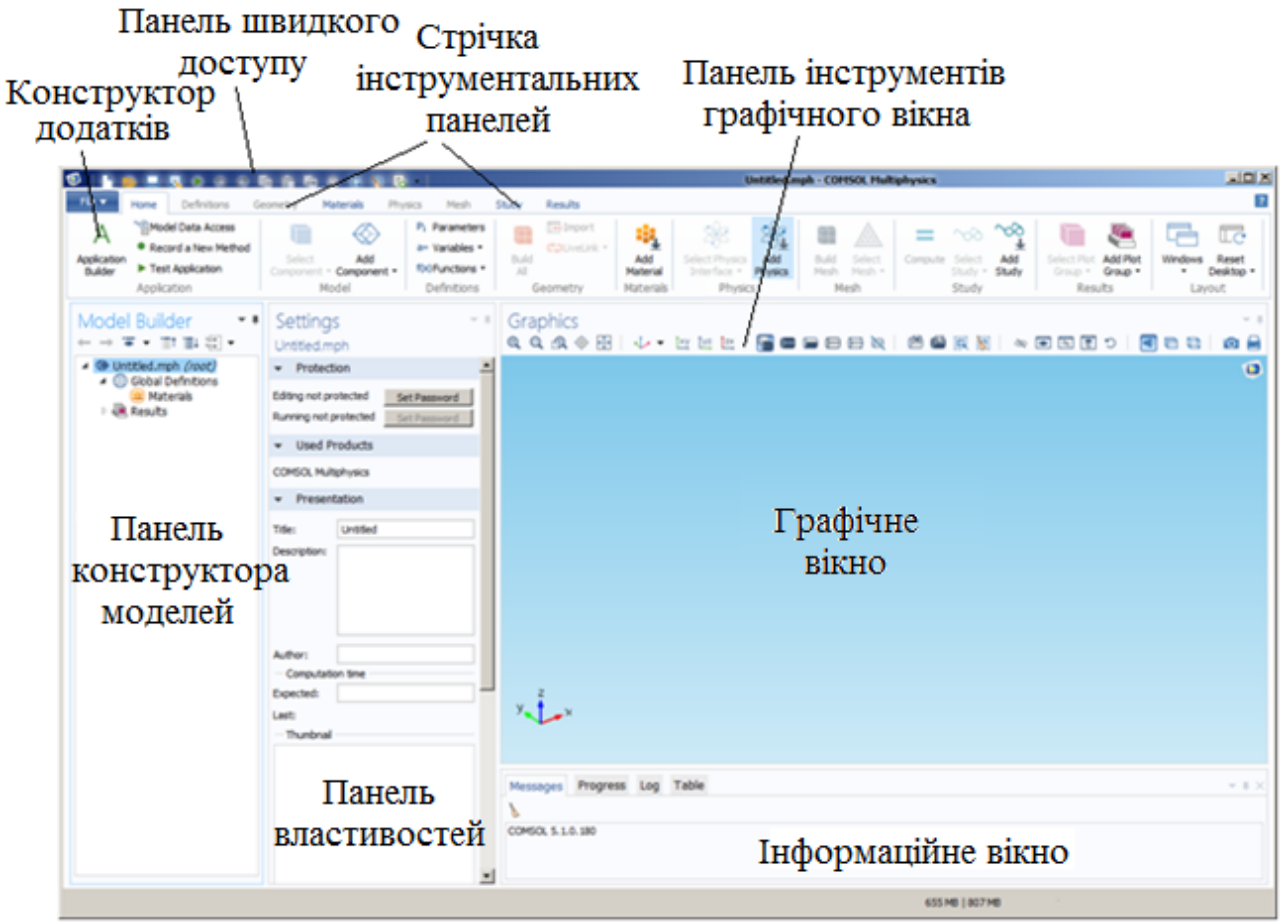
інтерфейсом користувача. Другий ярлик, з назвою «COMSOL 5.1 with MATLAB», запускає MatLab з можливістю доступу до моделей COMSOL. Програма має інструменти для взаємодії з найбільш поширеними системами САПР, такими як Pro/ENGINEER (Creo Parametric), SOLIDWORKS, Autodesk Inventor, Autodesk Revit, Solid Edge. Існує можливість керувати моделями COMSOL з Excel.

В документації на прикладі мови програмування Java докладно описана можливість доступу до COMSOL API. В п'ятій версії програми з'явилося ще одне нововведення – середовище розробки додатків (Application Builder), яке дозволяє конвертувати існуючу модель COMSOL в придаток.

Для багатоядерних та багатопроцесорних комп'ютерів існує можливість використовувати всі процесори і всі ядра.

1.2. Опис інтерфейсу на прикладі розв'язання рівняння Пуассона.

Після запуску програми ви побачите наступне вікно (ми припускаємо, що користувач не міняв налаштувань системи після інсталяції).



Панель швидкого доступу містить кнопки, які виконують особливо часто використовувані команди, такі як Save, Open, Undo, Copy, Paste та ін. Користувач має можливість самостійно налагоджувати цю панель.

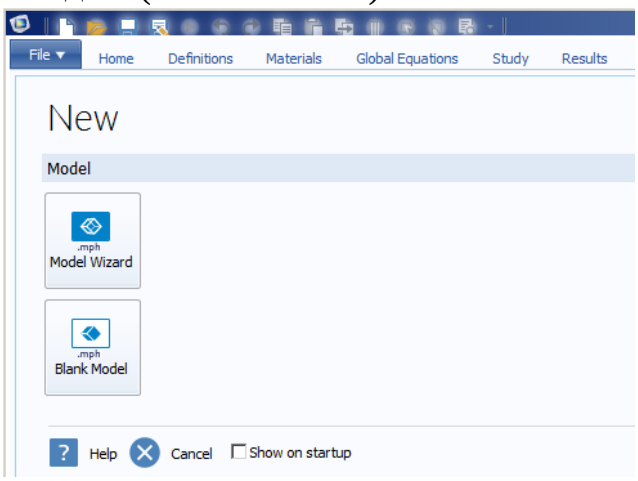
Стрічка інструментальних панелей, яка розташована вгорі, містить закладки з кнопками, які виконують більшість команд моделювання.

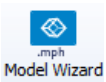
Панель конструктора моделей (Model Builder) надає графічне відображення процесу моделювання у вигляді дерева, елементи якого керують всім процесом постановки і розв’язання задачі. Кожний аспект задачі репрезентується у вигляді гілок та вузлів цього дерева. Через їх властивості можна задавати форму об’єктів, рівняння, граничні і початкові умови, будувати скінченно-елементну сітку, змінювати параметри солвера (метода розв'язання), відображати результати обчислювання в текстовому або графічному вигляді. Гілки можуть мати відгалуження та вузли, які уточнюють або розширюють можливості батьківських вузлів. Клацання правою кнопкою миші по елементу дерева моделі відкриває контекстне меню, команди якого дозволяють додати новий вузол, видалити поточний, змінити налаштування, які пов’язані з цим вузлом, і т.д. Виділення вузла в дереві приводить до відкриття панелі властивостей/налаштувань (Settings), зв’язаної з поточним вузлом, і яка дозволяє вводити властивості відповідного елементу моделі. Ми будемо називати її панеллю властивостей (ПВ) або вікном властивостей (ВВ). Введення вихідних даних з використанням цього дерева в термінології розробників COMSOL Multiphysics називається моделюванням задачі.

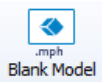
В графічному вікні створюються креслення областей/тіл і будуються графіки.

В інформаційному вікні відображаються числові значення розв’язків задач, а також повідомлення , які з’являються в процесі розв’язання.

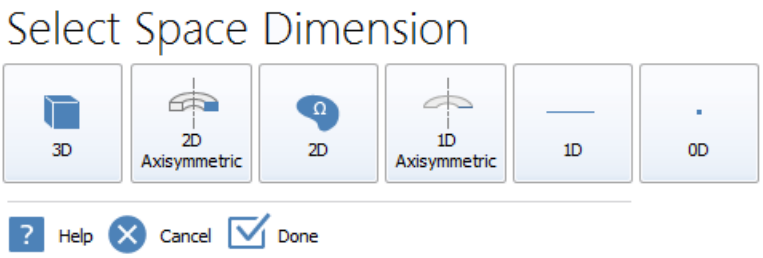
Робота в пакеті починається зі створення шаблону моделі задачі. Потрібно вибрати пункт меню File->New і у відчиненому вікні вибрати одну з двох можливостей – створення моделі за допомогою майстра (Model Wizard) або створення порожньої моделі (Blank Model).



Обидві кнопки 

і 

мають однакові функціональні можливості, але перша приводить до здійснення кроків моделювання в певній послідовності. Натиснемо кнопку Model Wizard. Відчиняється вікно, в якому обирається просторова модель задачі.



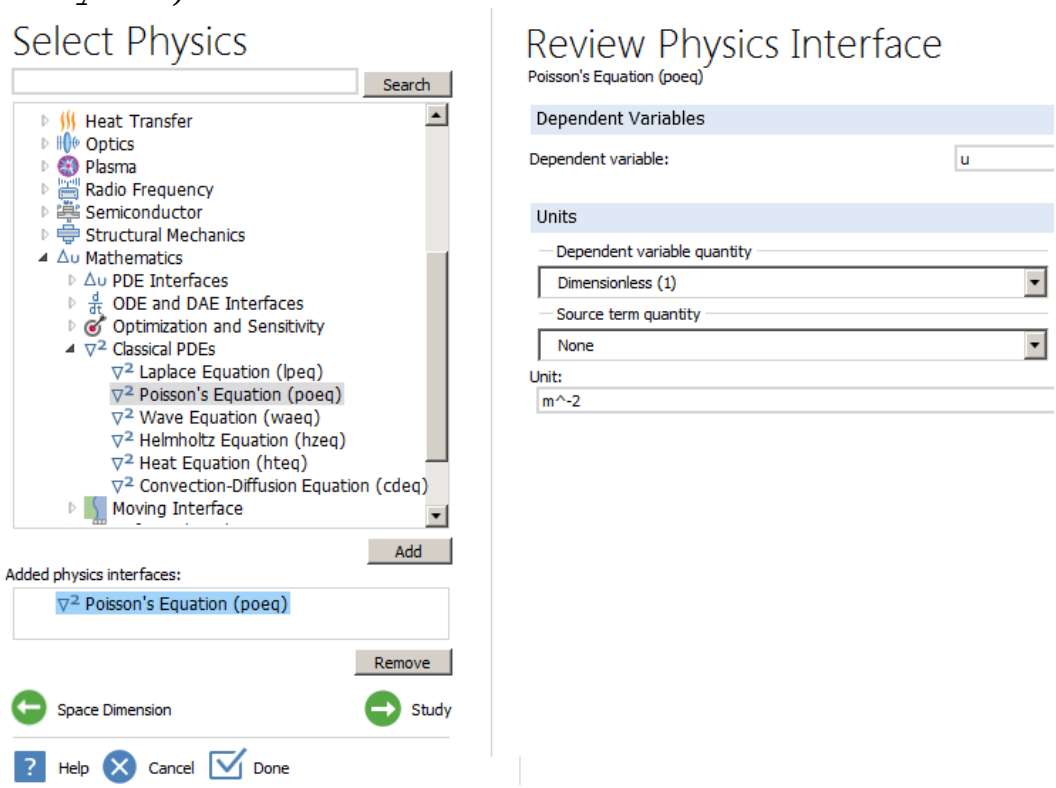
З написів на кнопках зрозуміло, яка модель буде досліджуватися. Варіант `0D` свідчить про створення просторово-незалежної моделі. Це означає, що будуть визначатися значення фізичних величин в точках. Інші варіанти вказують на створення моделі, в якій шукані величини будуть залежати від просторових координат. При цьому вони (величини) можуть не залежати від часу (стаціонарна модель) або бути функціями часу (динамічна модель). Обрання стаціонарної або динамічної моделі виконується на одному з наступних кроків роботи майстра моделей.

Припустимо, що ми бажаємо розв'язати рівняння Пуассона з граничними умовами Діріхле

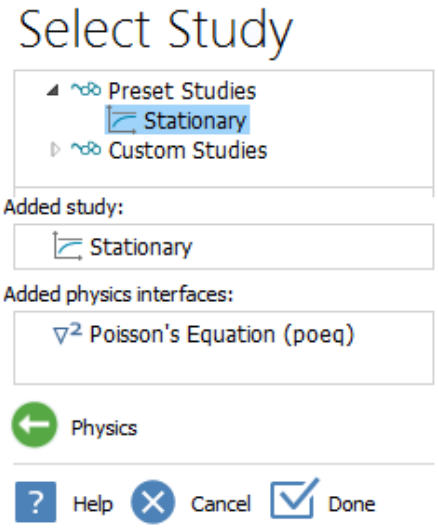
$$\Delta u = 16 \cdot \left(x^2 + y^2\right), \quad u\big|_{\partial\Omega} = 1, \tag{1}$$

де область Ω зображає коло одиничного радіуса з центром в початку координат. Вибір цієї задачі пояснюється тим, що її точний розв'язок відомий $u = \left(x^2 + y^2\right)^2$, і ми зможемо порівняти його з розв'язком, отриманим за допомогою COMSOL Multiphysics.

У вікні `Select Space Dimension` обираємо варіант `2D`. Для цього натискаємо на відповідну кнопку. Відчиняється вікно вибору фізичної моделі (`Select Physics`).



Воно містить список визначених інтерфейсів (прикладних режимів). Розкриваємо гілку Mathematics підгілку Classical PDEs та обираємо Poisson's Equation(poeq). Натискаємо кнопку Add. В поле **Added physics interfaces**, яке розташовано нижче, додається назва режиму. Праворуч в панелі Review Physics Interface в полі **Dependent Variable** можна змінити ім'я шуканої функції. Залишимо літеру u. Нижче в розділі **Units** в полі **Dependent variable quantity** обираємо Dimensionless (безрозмірний), тобто величина u буде безрозмірна. Натискаємо внизу кнопку Study. Відкривається вікно Select Study. Вибираємо варіант Stationary, який означає, що шукані функції моделі не залежать від часу (в рівняння Пуассона час не входить).



Завершуємо все натисканням на кнопку Done, розташовану внизу панелі Select Study. В результаті створюється вікно задачі з панеллю Model Builder в його лівій частині. Панель містить дерево, елементи якого керують процесом постановки і розв'язання задачі. Ми будемо називати його деревом моделі, а введення вхідних даних з використанням цього дерева будемо називати моделюванням задачі. Моделювання починається з налаштування параметрів вузлів цього дерева і, якщо потрібно, в додаванні нових вузлів.

Приклад 1. Продемонструємо процес моделювання на прикладі задачі (1).

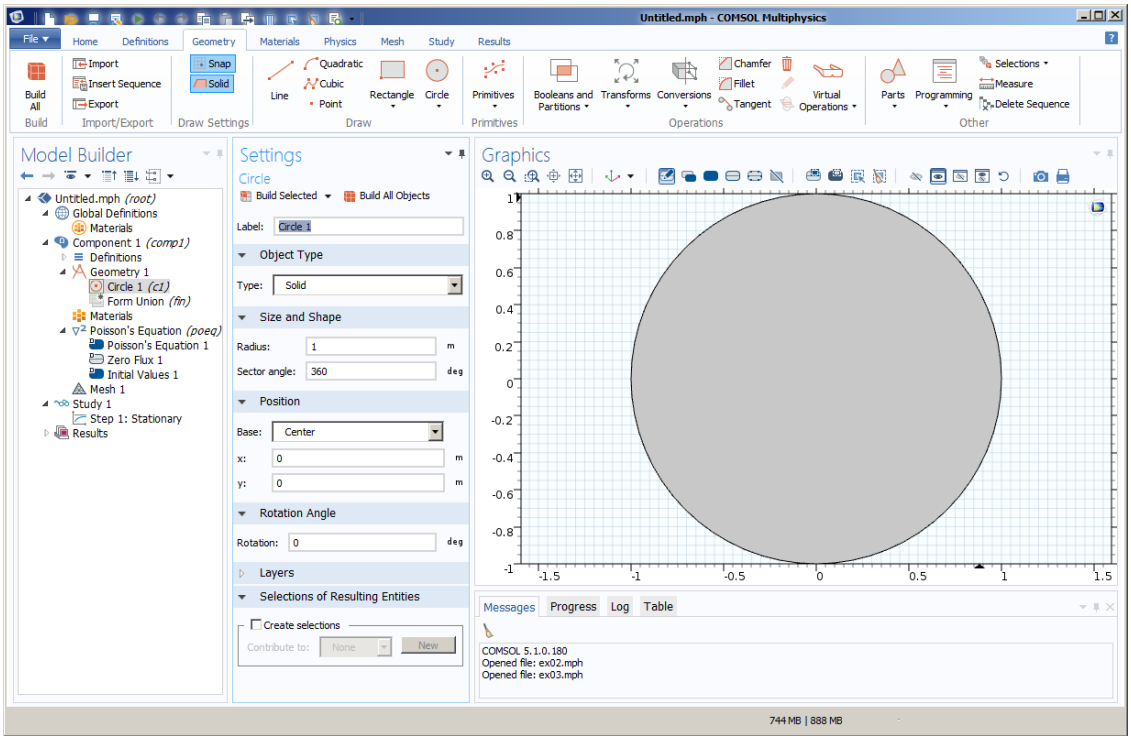
Введення умов здійснюється в чотири етапи:

1. Задання геометрії області Ω.
2. Побудова скінченно-елементної сітки.
3. Задання параметрів рівняння.
4. Задання граничних умов.

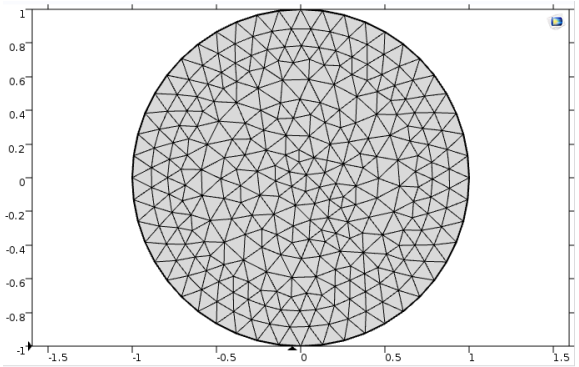
Якщо в панелі Select Physics буде обрана фізична модель не з гілки Mathematics, то можуть знадобитися інші етапи, наприклад, вибір матеріалу конструкції.

1. *Створення двовимірної області* Ω. В дереві моделі виділіть вузол **Geometry 1**. Він використовується для креслення геометрії моделі. Після виділення цього вузла графічне вікно зміниться і набуде вигляду, який налаштований для креслення плоских геометричних фігур. На стрічці панелей перейдіть до закладки Geometry. Вона містить інструменти креслення.

Клацніть на ній один раз мишею по кнопці Circle, наведіть курсор миші в зону креслення, натисніть ліву кнопку, посуньте мишу і відпустіть кнопку. В результаті буде намальовано коло (по центру і радіусу). Вам не обов'язково креслити коло точно такого розміру і положення, як треба. Параметри кола можна скорегувати наступним чином. В дереві моделі в гілці **Geometry 1** з'явиться дочірній вузол **Circle 1**. Поки він активний (виділений) праворуч від нього буде відкрита панель властивостей цього вузла (нагадаємо, що цю панель ми коротко будемо позначати ПВ). В ній в розділі **Position** ви можете змінити координати центру x, y, а в розділі **Size and Shape** можете скорегувати радіус. Змініть значення координат центру так, щоб вони були рівні $x = 0, y = 0,$ а радіус дорівнював 1. Вгорі панелі Setting Circle натисніть кнопку Build All Objects. В результаті вікно програми набуде вигляду, показаному на наступному рисунку. Оскільки всі вікна плаваючі, то у вашій системі їх положення може відрізнятися від нашого.



2. *Побудова скінченно-елементної сітки*. В дереві моделі є вузол **Mesh 1**. Виділіть його і на стрічці панелей перейдіть на закладку Mesh. Клацніть мишею по кнопці Build Mesh. Буде виконана тріангуляція області і графічне вікно набуде наступного вигляду



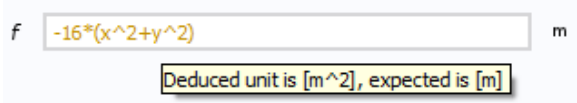
Закладка `Mesh` містить команди, які дозволяють вносити зміни в скінченно-елементну сітку. Зокрема, кнопка цієї панелі

містить випадний список команд, які дозволяють модифікувати сітку. У нашому прикладі можна погодитися з розбиттям області за замовчуванням, і сітку не змінювати.

3. *Задання параметрів рівняння*. В дереві моделі виділіть вузол **Poisson's Equathion 1**. Праворуч відкриється панель `Setting Poisson's Equathion`. В розділі **`Equation`** цієї панелі наведено рівняння Пуассона. В нашому випадку воно має вигляд $\nabla\cdot(-c\nabla u)=f$, де $\nabla=\left[\frac{\partial}{\partial x},\frac{\partial}{\partial y}\right]$ – оператор градієнта, а точка позначає скалярний добуток операторів. В координатній формі рівняння має наступний вигляд:

$$-\frac{\partial}{\partial x}\left(c\frac{\partial u}{\partial x}\right)-\frac{\partial}{\partial y}\left(c\frac{\partial u}{\partial y}\right)=f\,.$$

Нижче в розділі **`Diffusion Coefficient`** залиште 1 в якості значення коефіцієнта `c` цього рівняння, а в розділі **`Source Term`** введіть вираз для `f` – правої частини рівняння. Синтаксис запису математичних виразів близький до синтаксису MatLab. Вираз повинен мати вигляд $-16*(x\text{^}2+y\text{^}2)$ (з мінусом, оскільки мінус використовується в лівій частині рівняння). Зверніть увагу, що колір формули став помаранчевим. Це є попередженням, що можливо не всі дані введені коректно. В нашому випадку це пов'язано з невідповідністю фізичних одиниць вимірювання. Щоб зрозуміти, що відбувається, слід підвести курсор миші до формули. Спливе пояснююча підказка.

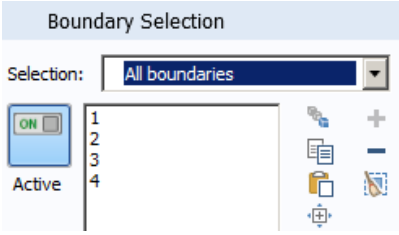


Зараз нас одиниці виміру не цікавлять, тому проігноруймо попередження. Якби ви ввели формулу невірно, то колір став би червоним (ознака помилки). Тоді обов'язково слід звернути увагу на спливаюче повідомлення і виправити помилку.

Зауваження. Для відключення контролю одиниць вимірювання треба у вузлі **Component 1** в списку **Unit system** вибрати `None`.

4. *Задання граничної умови*. Клацніть в дереві моделі правою кнопкою миші по гілці **Poisson's Equation (poeq)** (не по вузлу рівняння, а по імені його

батьківської гілки) і в контекстному меню оберіть Dirichlet Boundary Condition. В дереві моделі з'явиться вузол **Dirichlet Boundary Condition 1**, а праворуч відкриється його панель властивостей (ПВ). У графічному вікні межа кола розбита на 4 частини. Послідовно виділіть всі частини, клацаючи по ним мишею (при цьому дуги змінюють товщину і колір). Можна також в розділі **`Boundary Selection`** в списку **Selection** вибрати `All boundaries`. Номери ділянок окружності, на яких буде задана умова Діріхле, з'являться в секції вибору нижче списку **Selection**.



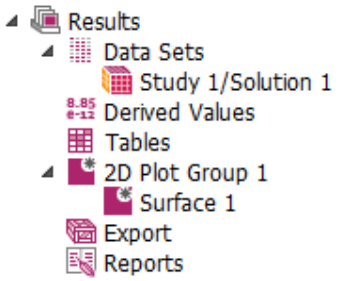
В розділі **`Dirichlet Boundary Condition`** цієї панелі в полі `r` виправіть значення 0 на 1. Цим задається гранична умова $u\big|_{\partial\Omega} = 1$.

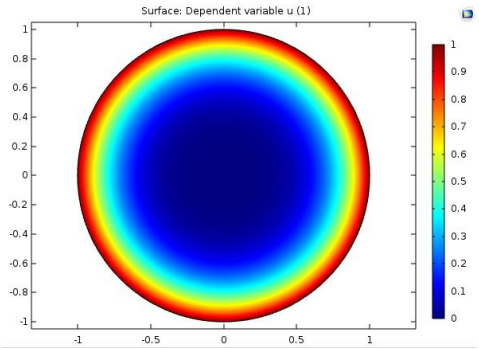
Зараз все готово для розв'язання задачі. В дереві моделі правою кнопкою миші клацніть вузол **Study 1** і в випадному меню виберіть команду Compute. В правому нижньому кутку головного вікна ви побачите індикатор виконання (Progress Bar).

Команду Compute можна також виконати натисканням на клавішу `F8` або клацанням миші по кнопці

, яка розташована на закладці `Home` стрічки панелей. Якщо всі умови задачі введені коректно, то в графічній області ви побачите кругову зону, на якій колір точок відповідає значенню/величині розв'язку $u(x, y)$. Таке функціональне забарвлення області ми іноді будемо називати залитим (або суцільним) контурним графіком.

Зверніть увагу, що малюнок теж належить дереву моделі, а саме гілці **Results**. До обчислення ця гілка містила порожні гілки. Тепер вона виглядає наступним чином

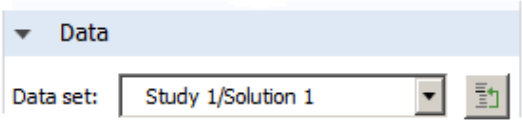




З'явився вузол **Study 1/Solution 1**, що представляє дані обчислення, і гілка **2D Plot Group 1** з вузлом **Surface 1**. Саме вузлу **Surface 1** належить побудований графік. Якщо ви виділите який-небудь інший вузол дерева моделі, наприклад, вузол **Geometry**, то вміст графічного вікна зміниться. Повторне

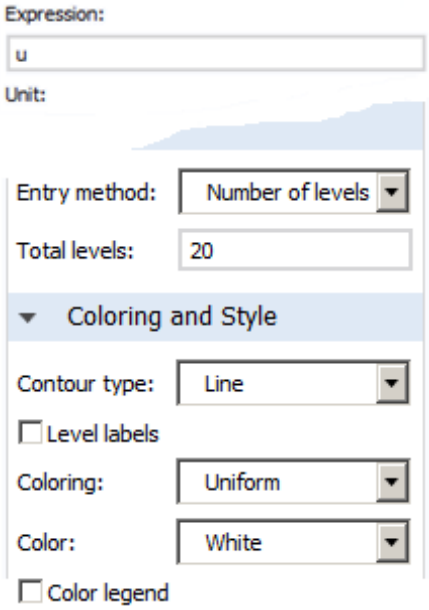
виділення вузла **Surface 1** знову відобразить залитий контурний графік в графічному вікні.

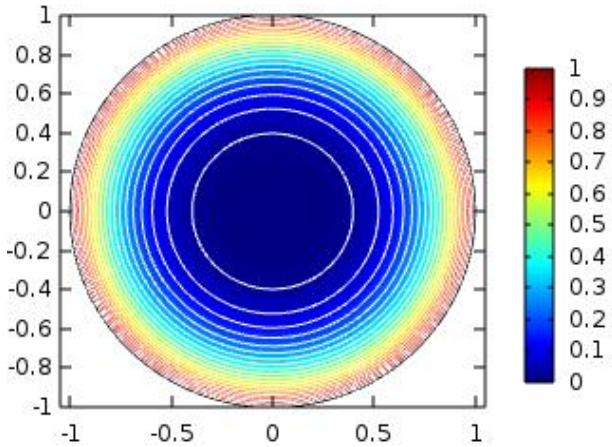
Гілка типу **2D Plot Group** призначена для побудови двовимірних графіків, а її вікно властивостей містить про них загальні відомості. Зокрема, в полі **Data Set** вказується джерело даних, яке використовується для побудови графіка.



Дані повинні «бути присутніми» в підгілці **Data Sets** гілки **Results**. В нашому випадку вони отримані в результаті обчислень і знаходяться у вузлі **Results - Data Sets – Study 1/Solution 1**. Сама гілка **2D Plot Group 1** повинна містити хоча б один дочірній вузол, який представляє графік конкретного типу. Зокрема, вузол типу **Surface** зображує залитий контурний графік, показаний вище.

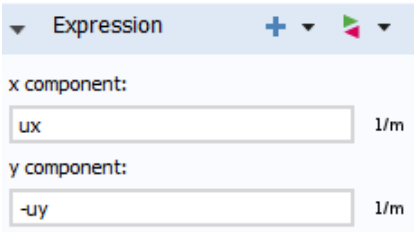
Графіки можуть накладатися один на одного. Для цього в гілку **2D Plot Group 1** можна додавати інші графічні вузли. Наприклад, клацніть по вузлу **2D Plot Group 1** правою кнопкою миші і в контекстному меню виберіть команду Contour. До групи графіків **2D Plot Group 1** додасться вузол **Contour 1**, який буде представляти графік ліній рівня виразу, записаного в полі **Expression** вікна властивостей цього вузла (в нашому випадку це `u` - ім'я розв’язку). В цьому вікні в розділі **`Levels`** в полі **Total Levels** можна задати кількість ліній рівня. В списку **Contour Type** виберіть тип графіка – `Line`, що означає зображення контурів лініями, а нижче в списку **Color** задайте колір ліній рівня, наприклад, білий. В цій **п**анелі **в**ластивостей (ПВ) також зніміть перемикач **Color legend** для того, щоб в графічному вікні не відображалася друга шкала кольорів. Частина ПВ вузла **Contour 1**, налаштованого таким чином, показана на наступному рисунку ліворуч.

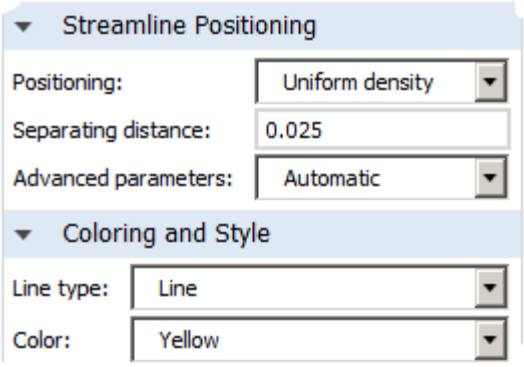




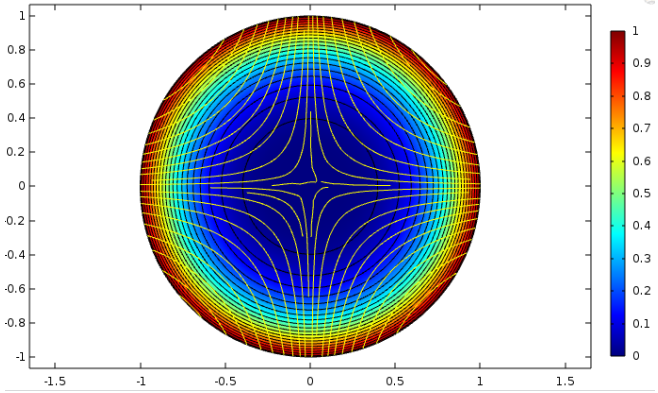
Для побудови графіка натисніть кнопку Plot, яка знаходиться вгорі панелі `Setting Contour`. В результаті, другий графік накладається на перший (див. попередній рисунок праворуч).

Додамо в цю групу графік ліній течії векторного поля $u'_x, -u'_y$. Для цього клацніть правою кнопкою миші по гілці **2D Plot Group 1** і в контекстному меню оберіть StreamLine. У вікні властивостей створеного вузла **StreamLine 1** в розділі **`Expression`** в полях **x component** і **y component** введіть `ux` і `-uy` (частинні похідні розв'язку `u` по `x` і по `y` позначаються `ux` і `uy`).



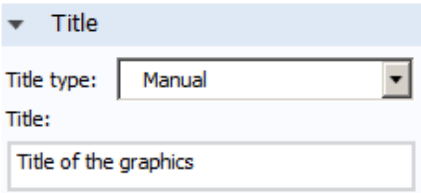


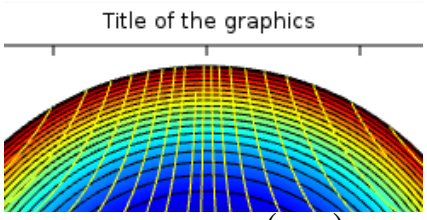
В розділі **`Streamline Positioning`** в списку **Positioning** виберіть `Uniform density` і в полі **Separating distance** задайте відстань між лініями. В розділі **`Coloring and Style`** в списку **Color** виберіть колір векторних ліній (див. попередній рисунок праворуч). Для побудови графіка натисніть кнопку Plot вгорі панелі `Setting StreamLine`. Третій графік буде накладено на перші два. Результуюче зображення наведено на наступному рисунку.



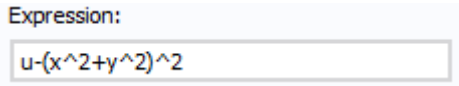
Таким чином, кожний 2D графік зображає вираз, записаний в його полі **Expression**, і графіки однієї групи накладаються в графічному вікні один на одного.

Зверніть увагу, що в вікнах властивостей є поля **Label**, в яких вводять назву вузлів дерева моделі (можна використовувати національні шрифти). Наприклад, в поле **Label** вузла **Surface 1** введіть текст `Поверхня`. В поле **Description** (якщо встановлено його перемикач) вводиться назва графіка. Вона відображається вгорі графічного вікна (дозволяється тільки латинський шрифт). В розділі **`Title`** в списку **Title type** вводиться назва заголовка, яка стоїть перед основним текстом. Якщо накладається декілька графіків, то заголовок краще задавати у вікні властивостей графічної групи **2D Plot Group 1** в розділі **`Title`**. Для цього в списку **Title type** треба вибрати `Manual` і в текстовому полі **Title** ввести заголовок об'єднаного графіка.

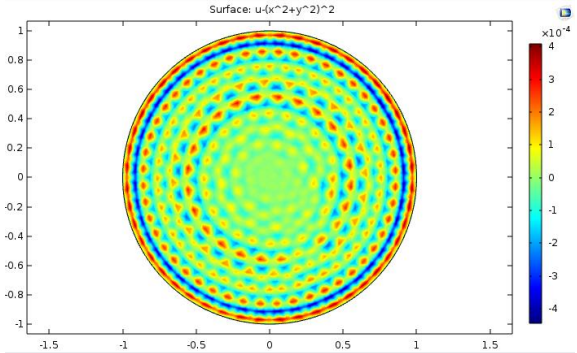




Порівняємо отриманий наближений розв'язок $u(x,y)$ з точним $w(x,y)=(x^2+y^2)^2$. Для цього побудуємо контурний графік різниці наближеного і точного розв'язків $u(x,y)-w(x,y)$. В дереві моделі клацніть правою кнопкою миші по гілці **Results** і в контекстному меню оберіть команду 2D Plot Group. З'явиться новий вузол **2D Plot Group 2**. Додайте в нього підвузол **Surface 1** так само, як ви це робили раніше (клацанням правою кнопкою миші і вибором з контекстного меню команди **Surface**). У панелі властивостей (ПВ) цього вузла в поле **Expression** введіть вираз $u-(x^2+y^2)^2$.



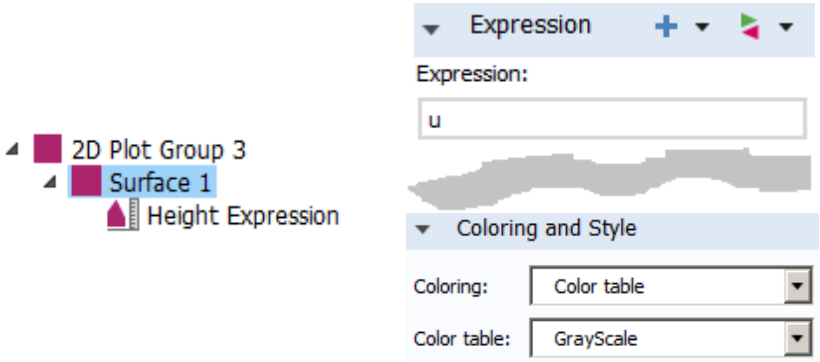
Побудуйте графік цього виразу клацанням по кнопці Plot в панелі властивостей (ПВ) вузла **Surface 1**.



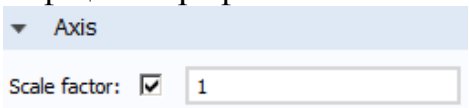
По шкалі кольорів, розташованій праворуч від рисунка, видно, що значення виразу $u-(x^2+y^2)^2$ змінюється в діапазоні від $-4\cdot10^{-4}$ до $+4\cdot10^{-4}$. Це означає, що абсолютна похибка (різниці наближеного і точного розв'язків) не перевищує $4\cdot10^{-4}$.

Якщо ви побажаєте зменшити похибку, то вам слід зменшити розмір скінченних елементів сітки. Для цього треба повернутися до вузла **Mesh 1** дерева моделі і до закладки `Mesh` на стрічці. Потім слід вибрати відповідний інструмент, згустити сітку і заново виконати обчислення.

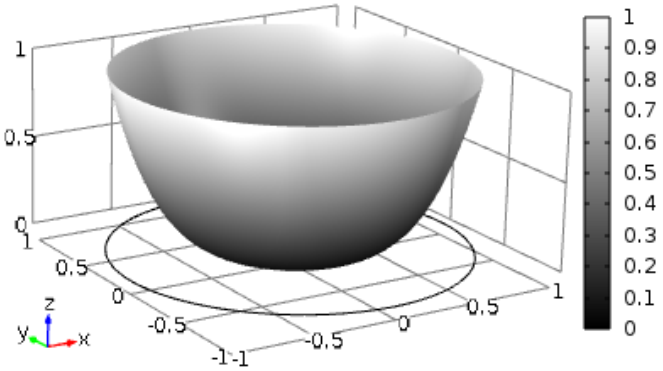
Є можливість намалювати графік розв'язку у вигляді поверхні. Для цього додайте ще одну гілку **2D Plot Group 3** і в ній підгілку **Surface 1**, в якій створіть вузол **Height Expression 1**. Він створюється так само, як і будь-який інший вузол, клацанням правої кнопки миші по батьківському вузлу і вибором в контекстному меню команди Height Expression. В ПВ вузла **Surface 1** в розділі **`Coloring and Style`** в списку **Color table** виберіть таблицю кольорів `GrayScale`.



В ПВ вузла **Height Expression 1** в розділі **`Axis`** в поле **Scale factor** встановіть одиничний масштабний коефіцієнт графіка.

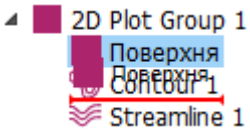


Нарешті натисніть кнопку Plot , яка будує графік поверхні функції.

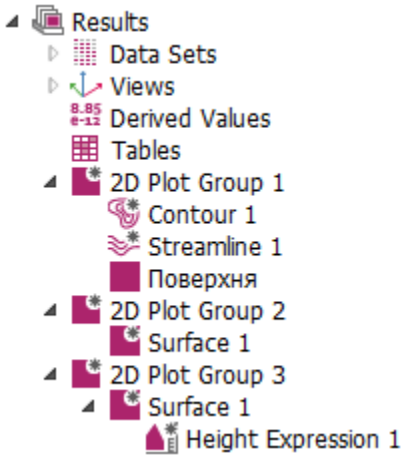


Те, що це поверхня функції `u` визначається установкою в полі **Expression** вузла **2D Plot Group 3 - Surface 1**, бо у дочірньому вузлі **Height Expression 1** в полі **Height data** встановлено `From parent`, що означає деформування поверхні по даним батьківського вузла.

Положення вузлів в дереві моделі не фіксоване. Зокрема, графічні вузли можна переміщати в межах групи графіків або навіть переміщати в іншу графічну групу. Наприклад, захопіть лівою кнопкою миші вузол **Поверхня** (натисніть кнопку і не відпускайте) і перетягніть вузол вниз гілки **2D Plot Group 1**. В процесі переміщення положення вузла позначається червоним відрізком.



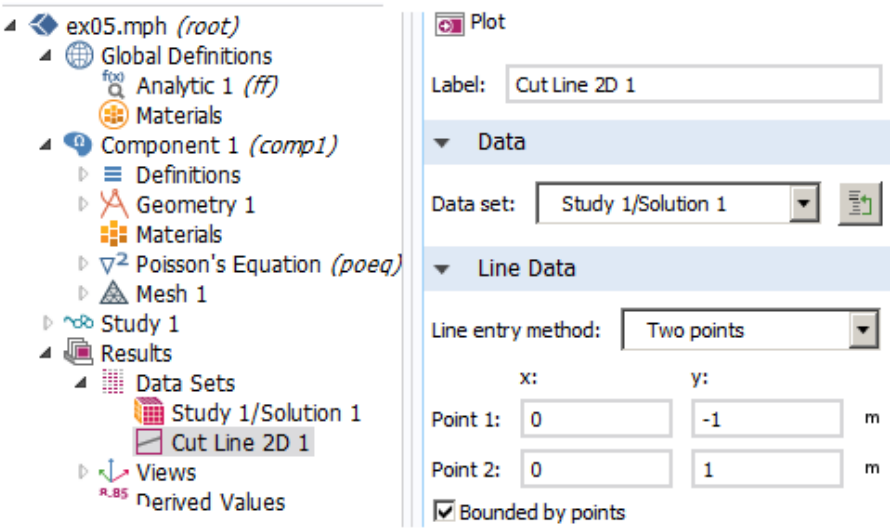
В результаті, гілка **Results** дерева моделі набуде наступного вигляду.



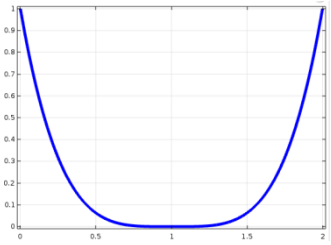
Графік може бути не пов'язаний з розв’язком `u`. Наприклад, додамо до дерева нашої моделі ще одну гілку **2D Plot Group 4** з вузлом **Surface**. В його ПВ в поле **Expression** введіть вираз `x^2-y^2` і натисніть кнопку Plot. В зоні одиничного кола буде побудовано залитий контурний графік гіперболічного параболоїда.

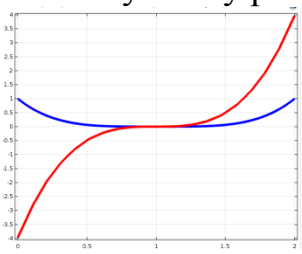
Зверніть увагу на те, що в розділі **`Data Set`** вікна властивостей вузла **2D Plot Group 4** все одно повинно бути вказано джерело даних `Study 1/Solution 1`. Воно зберігає не тільки значення шуканої функції `u`, а й координати вузлів скінченно-елементної сітки. Тому графік виразу `x^2-y^2` побудований на цій сітці (в одиничному колі).

Будувати графік розв’язку `u` можна не на всій області. Наприклад, побудуємо його вздовж вертикального діаметру нашого кола. Для цього в гілці **Data Sets** треба створити інше джерело даних – множину точок, над яким буде будуватися графік. Клацніть правою кнопкою миші по гілці **Data Sets** і в контекстному меню оберіть команду Cut Line 2D (лінія перерізу двовимірної області). В його ПВ в розділі **`Data`** в списку **Data set** виберіть джерело даних `Sudy 1/Solution 1`, а в списку **Line entry method** виберіть `Two points` (переріз по двом точкам). Введіть координати точок початку і кінця лінії перерізу в полях **x, y**. Панель властивостей вузла **Cut Line 2D** заповніть так, як показано на наступному рисунку (показана частина цього вікна).

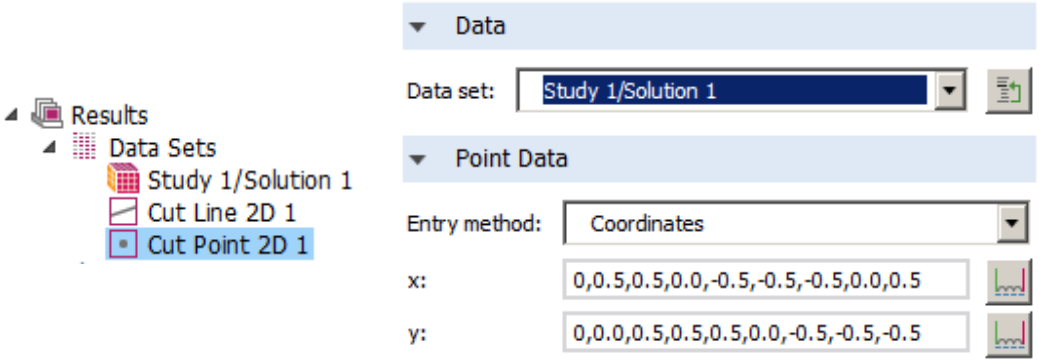


Тепер додайте графічну гілку **1D Plot Group 5**. В її ПВ в списку **Data set** виберіть `Cut Line 2D 1`. Додайте підвузол **Line Graph**. В його ПВ в списку **Data set** виберіть `From parent`, а в поле **Expression** введіть `u` (ім'я розв'язку). Зверніть увагу, що в розділі **`x-Axis Data`** в списку **Parameter** обрано значення `Arc length`, тобто наш графік віднесений до довжини дуги перерізу (діаметра). Вгорі ПВ вузла **Line Graph 1** натисніть кнопку Plot. В графічному вікні буде побудований графік, показаний на наступному рисунку ліворуч. Для побудови графіка частинної похідною розв'язку вздовж цього відрізка додайте в цю графічну групу вузол **Line Graph 2** і в його ПВ в поле **Expression** введіть `uy` (похідна розв'язку по змінній `y`). Решту полів заповніть так само, як і для вузла **Line Graph 1**. Після натискання кнопки Plot графічне вікно оновиться і в ньому буде побудований другий графік. Результуюче графічне вікно, яке відповідає вузлу **1D Plot Group 5**, показано на наступному рисунку праворуч.

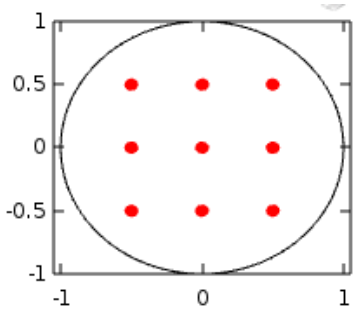




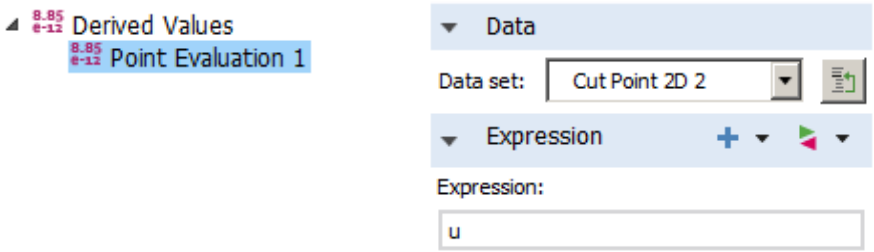
Знайдемо числові значення розв'язку в деяких точках. Для цього треба додати відповідне джерело даних в дерево моделі. Клацніть правою кнопкою миші по вузлу **Results - Data Sets** і у випадному списку виберіть команду Cut Point 2D. В гілку **Results - Data Sets** буде додано вузол **Cut Point 2D 1**. В його ПВ в розділі **`Data`** в списку **Data set** виберіть розв'язок `Sudy 1/Solution 1`, а в розділі **`Point Data`** введіть через кому *x* - координати і *y* - координати точок. Нижче показана частина ПВ вузла **Cut Point 2D 1**.



Натиснувши на кнопку Plot вгорі ПВ вузла **Cut Point 2D 1**, ви отримаєте зображення точок.



Тепер клацніть правою кнопкою миші по вузлу **Results - Derived Values** і у випадному списку виберіть команду Point Evaluation. З'явиться вузол **Results - Derived Values - Point Evaluation 1**. В його ПВ в списку **Data set** слід вибрати `Cut Point 2D 1` (створене раніше точкове джерело даних). Також в цій ПВ потрібно ввести вираз, значення якого буде обчислюватися. В нашому випадку ми будемо обчислювати значення розв'язку в точках. Тому в поле **Expression** вводимо `u` (нижче показана частина ПВ вузла **Point Evaluation 1**).



Натисніть на кнопку Evaluate вгорі ПВ вузла **Point Evaluation 1** і в правому нижньому куту програми з'явиться таблиця значень розв'язку в кожній вказаній вами точці.

Dependent variable u, Point: (0, 0)	Dependent variable u, Point: (0.5, 0)	Dependent variable u, Point: (0.5, 0.5)	Dependent variable u
-0.00000020173	0.062406	0.24998	0.062505

Створену модель можна зберегти в файл, який має розширення імені `*.mph`. Зберігається вся модель разом з результатами обчислень (якщо вони виконувалися) і графікою. Пізніше ви можете відкрити такий файл і продовжити роботу з моделлю.

▪

Зауваження 1. В гілці **Results - Data Sets** джерела даних типу `Cut Point` використовуються для обчислення значень виразів в точках та побудови в точках «часових» графіків; вузли типу `Cut Lines` в основному призначені для обчислення і візуалізації виразів вздовж деяких ліній, наприклад, вздовж лінії

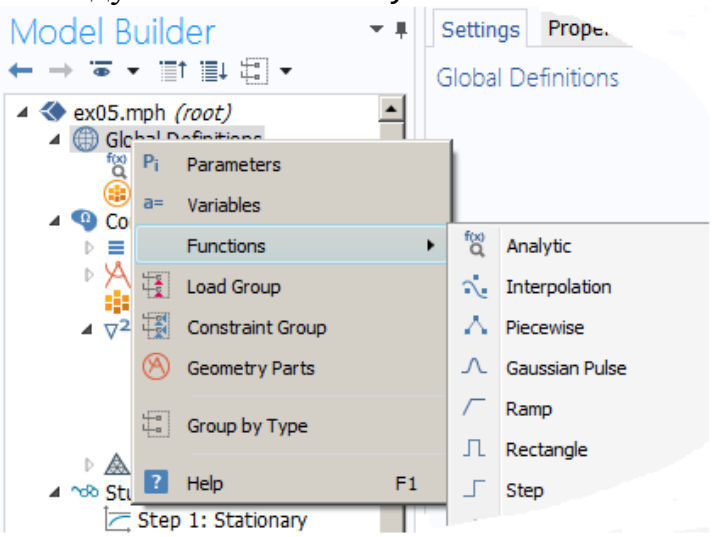
перерізу будь-якої двовимірної області. Вузли типу `Cut Plane` створюються для візуалізації розв’язків/виразів по перерізах тривимірних областей.

Зауваження 2. Отримати числове значення розв’язку в точці можна по-іншому. Активуйте графічний вузол, в якому побудований графік з функціональної забарвленням (не вузол групи, а вузол графіка), і на графіку виконайте подвійне клацання миші в околі бажаної точки. З'явиться таблиця (якщо її ще немає), в яку додасться рядок зі значеннями координат точки клацання і значенням виразу в ній. Кожне нове клацання буде додавати один рядок в цю таблицю.

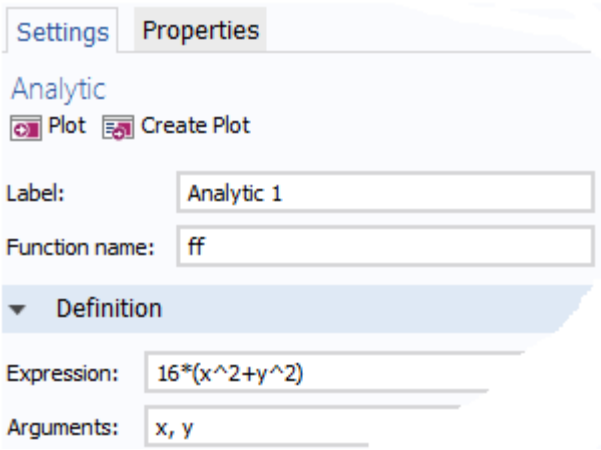
1.3. Функції і графіки.

Система COMSOL Multiphysics не використовує вбудованої мови програмування на зразок MatLab, але в ній є можливість записувати досить складні математичні вирази. Окрім того, користувач може створювати власні функції. Кожній функції в дереві моделі повинен відповідати свій вузол. Функції створюються в гілках **Global Definitions** або **Component - Definitions** дерева моделі вибором відповідного шаблону зі списку.

Припустимо, що в прикладі 1 попереднього параграфа праву частину рівняння (1) ми збираємося вводити у вигляді аналітичної функції. Для її створення клацніть правою кнопкою миші по вузлу **Global Definitions** дерева моделі і виберіть команду Functions->Analytic.



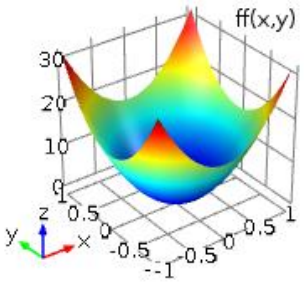
Для функції буде створено власний вузол і відкриється панель властивостей, призначена для задання функції за допомогою аналітичного виразу. В поле **Function name** введіть ім'я функції, наприклад, `ff`. В полі **Arguments** через кому перелічіть імена аргументів функції, а в полі **Expression** запишіть математичний вираз, використовуючи імена аргументів, константи і знаки математичних операцій. Заповнені поля в ПВ нашої функції показані на наступному рисунку.



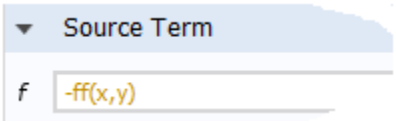
Перевірку коректності задання функції можна виконати шляхом побудови її графіка. Для цього розкрийте розділ **`Plot Parameters`** і в таблиці задайте діапазон зміни аргументів функції на графіку (це не обмежує діапазон аргументів при подальшому використанні).

Argument	Lower limit	Upper limit
x	-1	1
y	-1	1

Після цього натисніть кнопку Plot вгорі ПВ. В графічному вікні буде побудований графік нашої функції $\mathrm{ff}(x, y)$.



Перейдіть у вікно властивостей вузла **Component 1 - Poisson's Equation (poeq) - Poisson's Equation 1** і в розділі **`Source Term`** в поле **f** введіть $-\mathrm{ff}(x, y)$.



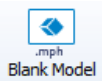
На закладці Home стрічки панелей натисніть кнопку Compute і отримайте той же розв'язок, що і раніше.

В системі є шаблони для створення аналітичних, інтерполяційних, кускових, пилкоподібних, східчастих і інших функцій. Можна використовувати функції, створені в Matlab. У кожного типу функцій є свої особливості. Наприклад, Piecewise (кускові) функції допускають використання тільки одного аргументу. З особливостями створення різних функцій ми познайомимо вас у міру їх появи в наших прикладах.

Після створення функцій часто виникає потреба подальшої роботи з їх графіками (поєднання, накладання, тощо). З розібраного раніше прикладу ви бачили, що в гілці **Results** дерева моделі можна будувати і обробляти графіки розв'язків. Але якщо треба працювати лише з графіками функцій, то це можна робити без моделювання і розв'язання фізичних задач. Для цього потрібно виконати всього три кроки:

- створити функцію;
- в гілці **Results->Data Sets** створити джерело даних (`Data set`); в деякому сенсі джерело даних можна розглядати як область визначення функції, а точніше як область зміни її аргументів при побудові графіка;
- створити групу графіків **Plot Group** з графічним вузлом того чи іншого типу, наприклад, **Line Graph**; в ПВ цього вузла вказати функцію і джерело даних (`Data set`).

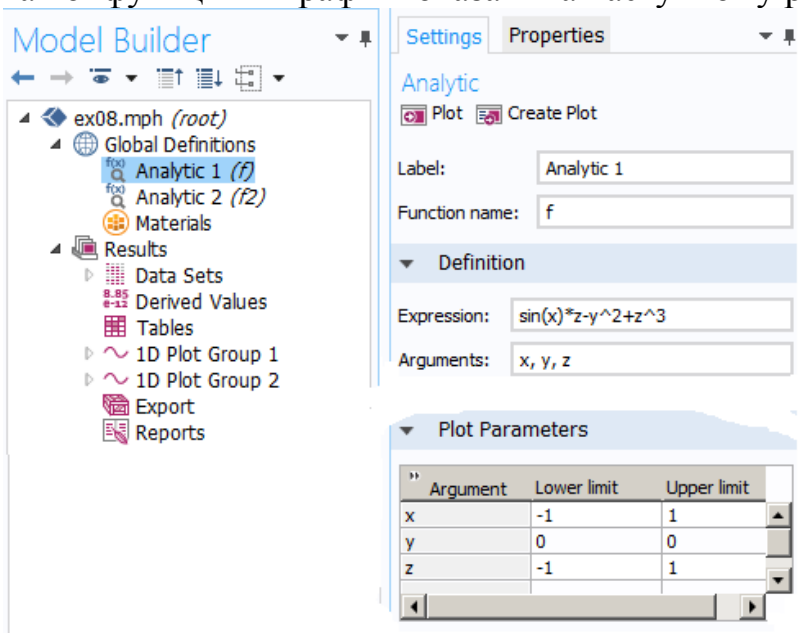
Кожен з кроків можна виконати декількома способами. Для прикладу побудуємо кілька графіків функцій.

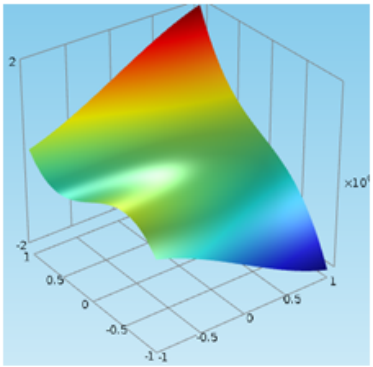
Приклад 1. Починаємо зі створення порожньої моделі. Вибираємо пункт меню File->New і у відкрившомуся вікні вибираємо створення порожньої моделі (`Blank Model`), тобто тиснемо кнопку 

.

Створюємо функцію. Клацніть правою кнопкою миші по вузлу дерева моделі **Global Definitions** і виберіть команду Functions->Analytic. Для функції створиться вузол і відкриється панель властивостей, призначена для задання функції за допомогою аналітичного виразу. В поле **Function name** введіть ім'я функції, наприклад, `f`. В полі **Arguments** через кому перелічіть імена аргументів функції `x,y,z` (функція може мати кілька аргументів). В поле **Expression** введіть математичний вираз `sin(x)*z-y^2+z^3`. Для контролю побудуйте графік функції двох змінних, задавши фіксоване значення одного з аргументів в розділі **`Plot Parameters`** (щоб присвоїти `y = 0` задайте нульові нижню і верхню межі зміни `y`), і натисніть кнопку

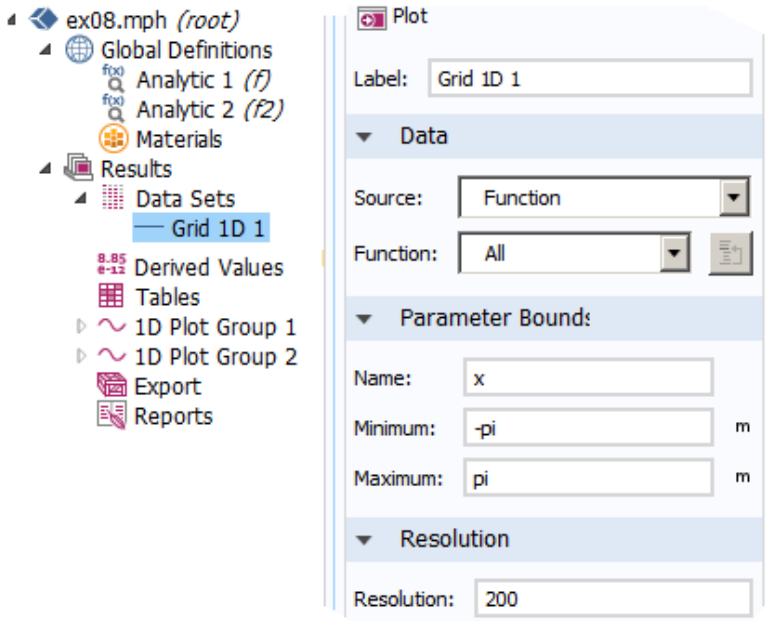
. Заповнені поля ПВ нашої функції і її графік показані на наступному рисунку.





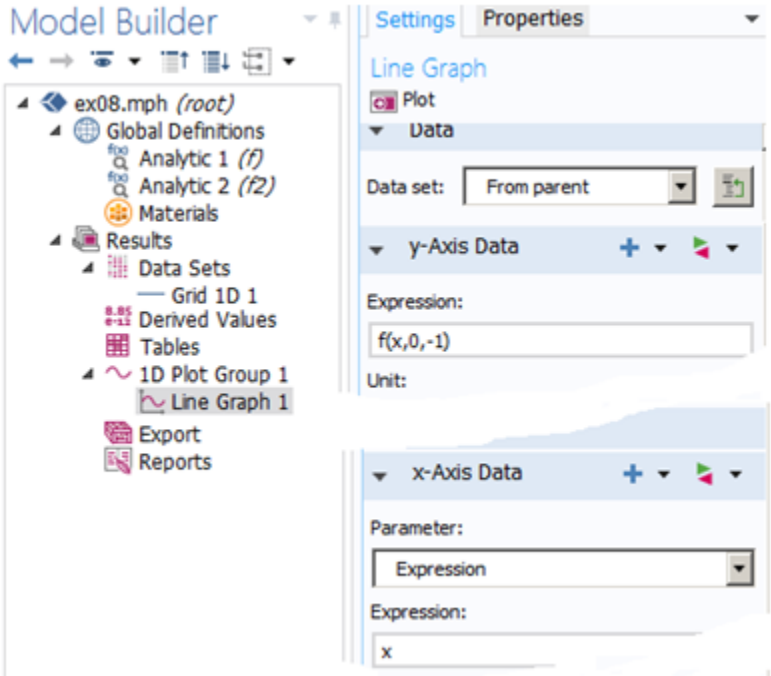
Аналогічно створіть другу функцію однієї змінної `f2` з виразом `x^3`.

Створюємо одновимірне *джерело даних*, яке буде використовуватися для побудови графіків функцій. Для цього у випадному меню гілки **Results - Data Sets** оберіть команду More Data Sets-> Grid 1D і заповніть ПВ створеного вузла **Grid 1D** так, як показано нижче.



Вибір в списку **Source** значення `Function` означає, що на множині точок **Grid 1D** буде обчислюватися значення функції. У списку **Function** обираємо потрібну функцію (або `All`, якщо збираємося будувати кілька графіків). В полі **Resolution** вказується кількість точок дискретизації відрізка, який визначається значеннями, заданими в полях **Minimum** і **Maximum**.

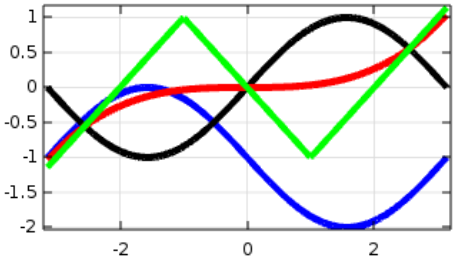
Створюємо групу графіків. Для цього в гілку **Results** додайте гілку типу **1D Plot Group** і в ній створіть вузол типу **Line Graph**. В ПВ гілки **1D Plot Group 1** в списку **Data Set** виберіть `Grid 1D 1` (одновимірна множина, на якій збираємося будувати графік). У вікні властивостей вузла **Line Graph 1** в розділі **`y-Axis Data`** в полі **Expression** введіть вираз `f(x,0,-1)`. В розділі **`x-Axis Data`** в списку **Parameter** виберіть `Expression`, і в поле **Expression** введіть `x`. В розділі **Coloring and Style** задайте стиль, колір і товщину лінійного графіка.



Для побудови графіка натисніть кнопку Plot. Діючи аналогічно, у вузол **1D Plot Group 1** додайте ще три підвузли типу **Line Graph**, в яких в полях **Expression** розділів **`y-Axis Data`** введіть вирази

```
f(x,1,1)
f2(x)/30
x-abs(x+1)+abs(x-1).
```

Решту властивостей задайте такими же, як у вузла **1D Plot Group 1**. Після побудови всіх кривих графічне вікно повинно набути наступного вигляду.



Побудуємо кілька плоских кривих по їх параметричним рівнянням.

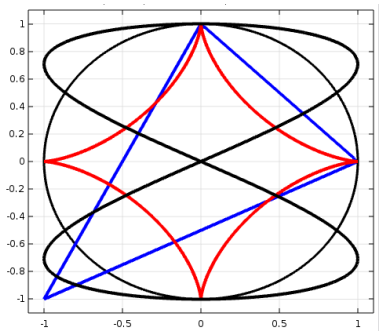
В тій же моделі в гілку **Results** додайте вузол графічної групи **1D Plot Group 2** і створіть в ньому підвузол **Line Graph**. В ПВ гілки **1D Plot Group 2** в списку **Data Set** оберіть `Grid 1D 1`. У вікні властивостей вузла **Line Graph 1** в розділі **`y-Axis Data`** в полі **Expression** введіть вираз, наприклад, `cos(x)`. В розділі **`x-Axis Data`** в списку **Parameter** обиріть `Expression`, і в поле **Expression** введіть `sin(x)`. Цим ми задаємо параметричне рівняння кола. В розділі **`Coloring and Style`** задайте стиль, колір і товщину кривої. Для побудови кривої натисніть кнопку Plot. В цю ж гілку **1D Plot Group 2** додайте ще три підвузли типу **Line Graph**. В їх розділах **`x-Axis Data`** в списках **Parameter** оберіть `Expression`, і в полях **Expression** в розділах **`y-Axis Data`** і **`x-Axis Data`** введіть наступні пари виразів (параметричні рівняння кривих):

a) `abs(x)/2-3*abs(x-2)/2+abs(x-3)-1`
`abs(x)-3*abs(x-1)/2+abs(x-3)/2-1`

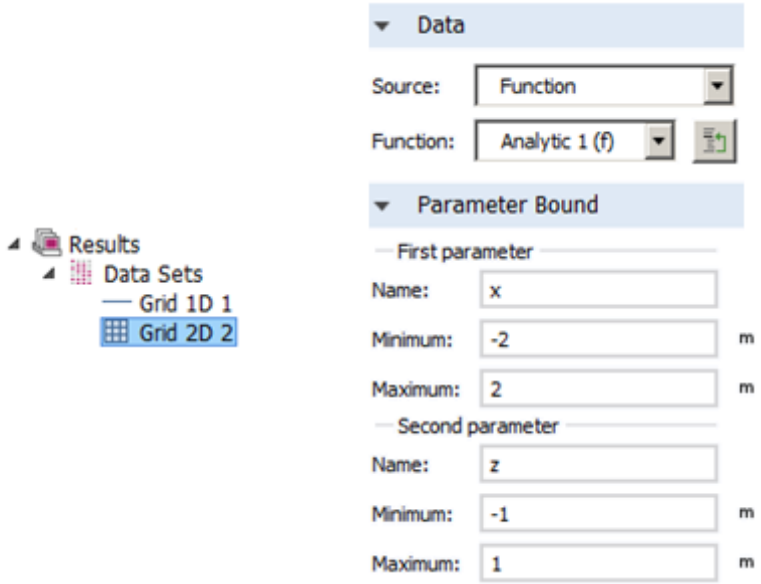
b) `cos(pi*x)^3`
`sin(pi*x)^3`

c) `cos(x)`
`sin(2*x)`

Після побудови всіх кривих кнопкою Plot графічне вікно набуде наступного вигляду.

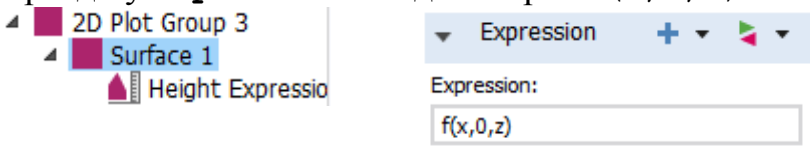


Можна будувати графіки функцій двох змінних. Для цього в гілці джерел даних **Results->Data Sets** потрібно створити двовимірну множину точок, яка буде використовуватися при побудові графіків. У випадному меню гілки **Data Sets** оберіть команду More Data Sets->Grid 2D і заповніть ПВ створеного вузла **Grid 2D 2** так, як показано нижче (не пропустіть налаштування в полі **Function**, яке повинно бути `Analitic 1(f)` або `All`).

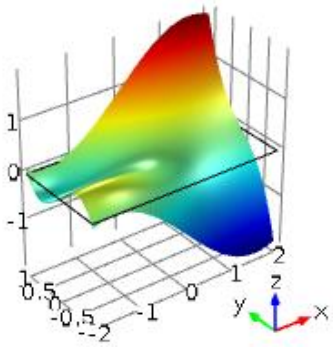


Зверніть увагу, що у другого параметра ми змінили ім'я. Для цього в розділі **`Second parameter`** в поле **name** ми ввели `z`.

В гілку **Results** додайте гілку графічної групи **2D Plot Group** і в ній створіть вузол типу **Surface**. В ПВ створеної гілки **2D Plot Group 3** в списку **Data Set** оберіть `Grid 2D 2`. У вікні властивостей вузла **Surface 1** в полі **Expression** розділу **`Expression`** введіть вираз `f(x,0,z)`.

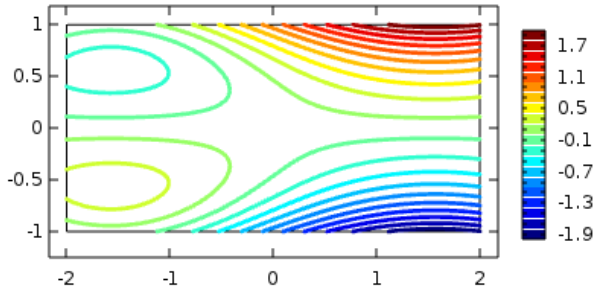


Щоб побудувати графік поверхні (а не суцільний кольоровий графік) до вузла **Surface 1** додайте підвузол **Height Expression**. В його ПВ в поле **Scale factor** (масштабний множник) введіть одиницю і натисніть кнопку Plot. Буде побудований наступний графік

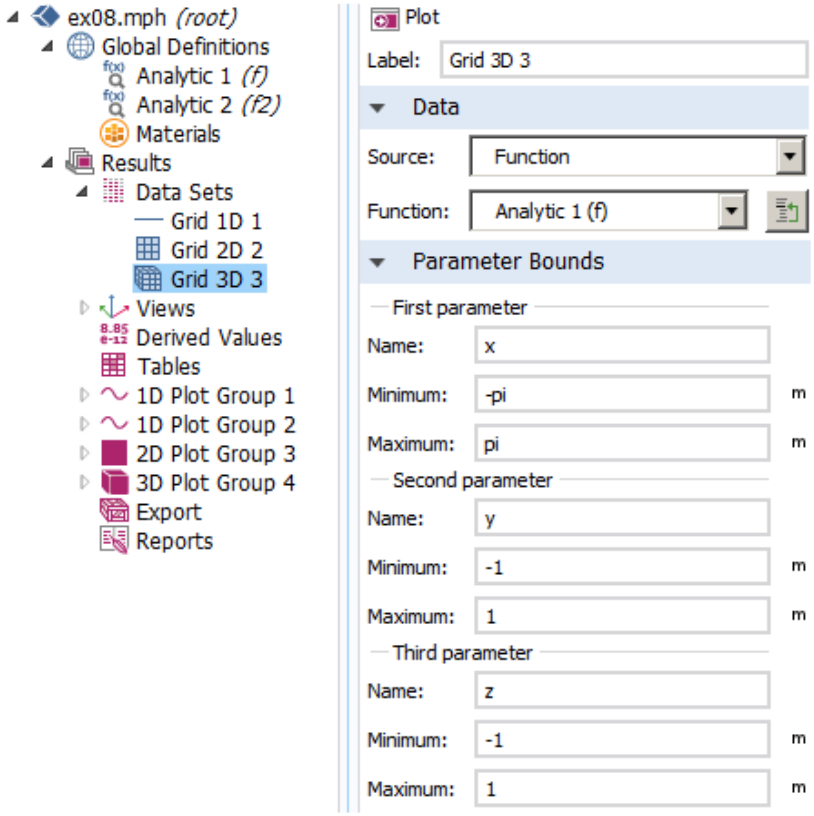


Замість вузла **Surface 1** в гілку **2D Plot Group 3** можна додати вузол **Contour 1**. Для цього клацніть правою кнопкою миші по вузлу **Surface 1** і в випадному меню виберіть команду Delete для видалення вузла або виберіть команду

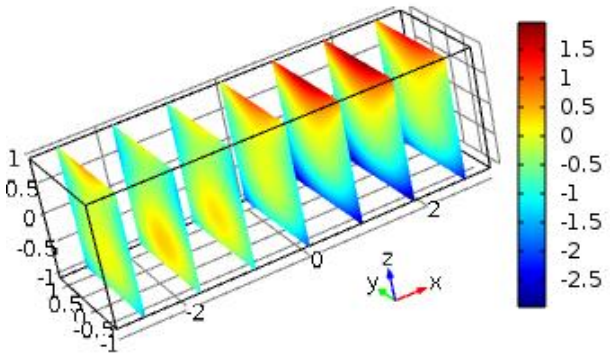
Disable для його деактивації, а потім додайте вузол типу **Contour**. В ПВ вузла **Contour 1** в поле **Expression** розділу **`Expression`** введіть попередній вираз `f(x,0,z)`, в поле **Total levels** розділу **`Levels`** введіть кількість ліній `20`, а в списку **Contour type** розділу **`Coloring and Style`** виберіть `Line`. В результаті буде побудований контурний графік функції 2 змінних.



Для графічного зображення функції 3 змінних можна використовувати «пошаровий» (slice) графік (графік перерізів). Для цього в гілці **Data Sets** командою випадного меню More Data Sets->Grid 3D створіть тривимірну множину точок **Grid 3D** (точніше вузол в дереві моделі). Заповніть його ПВ так, як показано на наступному рисунку.

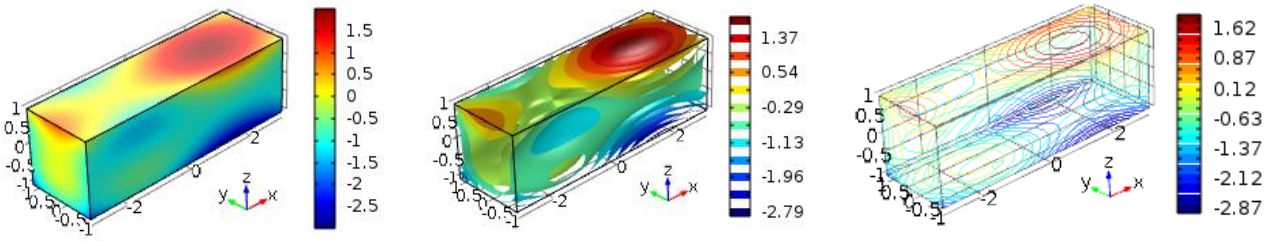


В гілку **Results** додайте підгілку **3D Plot Group** і в ній створіть вузол типу **Slice**. В ПВ гілки **3D Plot Group 4** в списку **Data Set** виберіть `Grid 3D 3`. У вікні властивостей вузла **Slice 1** в поле **Expression** розділу **`Expression`** введіть вираз `f(x,y,z)`. В розділі **`Plane Data`** в полі **Planes** введіть кількість перерізів/шарів, наприклад, `7`. Натисніть кнопку Plot. В результаті буде побудовано наступне зображення.



Колір точок, розташованих на зрізах, вказує значення функції. Шкала кольорів, яка представляє значення функції, показана праворуч від графіка.

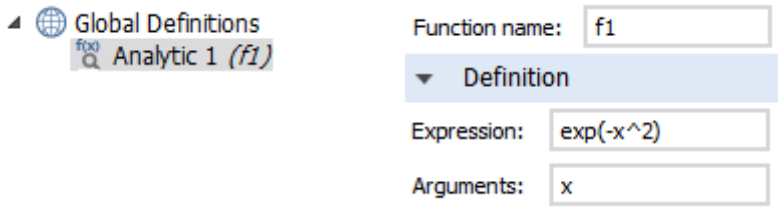
Ту ж функцію 3 змінних можна зобразити по-іншому. Наприклад, створіть ще графічні групи і вузли в них: **3D Plot Group 5 - Surface**, **3D Plot Group 6 - Isosurface**, **3D Plot Group 7 - Contour**. На наступному рисунку показані зображення функції `f(x,y,z)` побудовані на цих графіках (зліва направо **Surface**, **Isosurface** і **Contour**). Вікна властивостей цих вузлів мають поле **Expression** для введення виразу `f(x,y,z)`, а ПВ двох останніх вузлів мають ще поле **Total Levels** для завдання на графіку кількості поверхонь/ліній рівня.



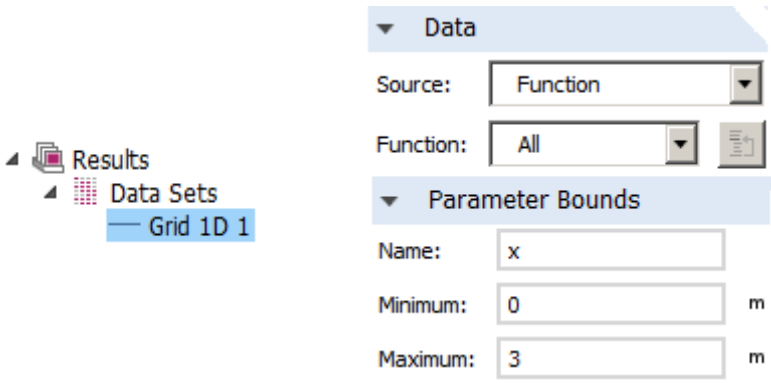
■

Розв’язок і будь-які функції можна інтегрувати.

Приклад 2. Побудуємо графік функції $f(x)=\int_0^x e^{-\tau^2}d\tau$. Для цього створимо порожній проект. У ньому створимо аналітичну функцію $f_1(x)=\exp\left(-x^2\right)$.



Далі створимо одновимірне джерело даних. Для цього у випадному меню гілки **Results - Data Sets** обираємо команду More Data Sets->Grid 1D і заповнюємо ПВ створеного вузла **Grid 1D** так, як показано нижче .



Створюємо групу графіків **1D Plot Group** і в ній створюємо вузол типу **Line Graph**. В ПВ гілки **1D Plot Group 1** в списку **Data Set** вибираємо `Grid 1D 1`. У вікні властивостей вузла **Line Graph 1** в розділі **`y-Axis Data`** в полі **Expression** вводимо вираз `integrate(f1(v),v,0,x)`.

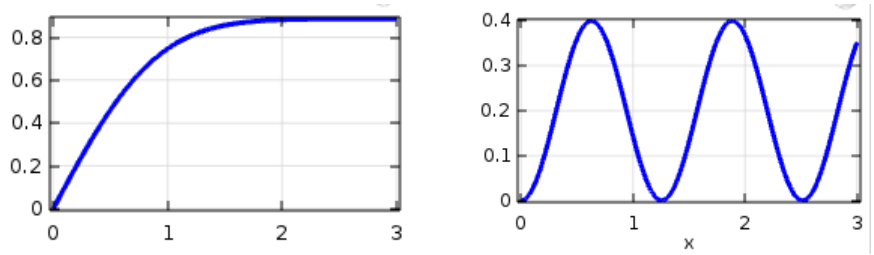
Оператор `integrate` має наступний синтаксис

`integrate(expr(v),v,v1,v2)`,

де `expr(v)` являє підінтегральний вираз, `v` – ім'я змінної інтегрування, `v1` і `v2` – нижня і верхня межа інтегрування, які не зобов'язані бути константами.

В розділі **`x-Axis Data`** в списку **Parameter** вибираємо `Expression`, і в поле **Expression** вводимо `x`. В розділі **`Coloring and Style`** задаємо стиль, колір і товщину лінії. Натискаємо кнопку Plot і отримуємо графік, показаний на наступному рисунку зліва.

Інтегрувати можна будь-який вираз, а не тільки створену заздалегідь функцію. Наприклад, створіть ще одну графічну гілку **1D Plot Group 2** з вузлом **Line Graph 1**. В ПВ гілки **1D Plot Group 2** в списку **Data set** виберіть `Grid 1D 1`. В ПВ його підвузла **Line Graph 1** в розділі **`y - Axis Data`** в полі **Expression** введіть вираз `integrate(sin(v*5),v,0,x)`, а в розділі **`x - Axis Data`** в полі **Expression** введіть `x`. Натискання на кнопку Plot побудує графік функції $\int_0^x \sin(5v)\,dv$, який показано на наступному рисунку праворуч.



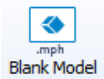
Інтегрування виразів, які залежать від часу, виконується оператором `timeint`.

Крім використання операторів `integrate` і `timeint` існують інші способи обчислення інтегралів. Ви познайомитеся з ними пізніше.

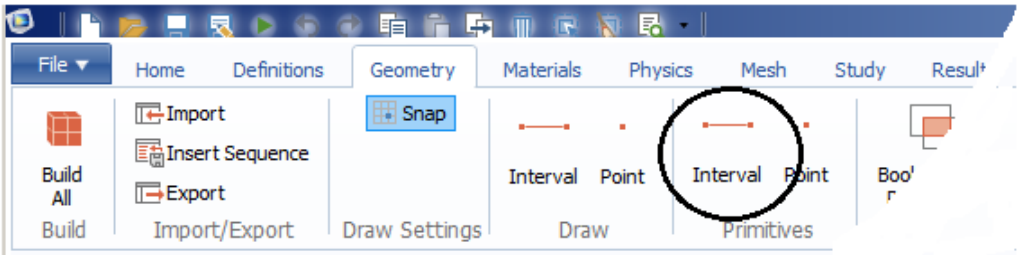
■

Інший спосіб задання множини точок, на якій будується графік функції, полягає в тому, щоб використовувати скінченно-елементну сітку, створену при моделюванні задачі, або створену спеціально.

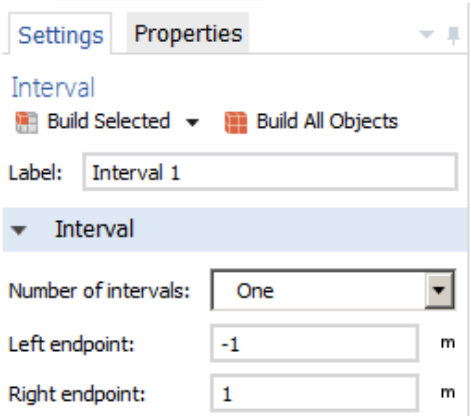
Приклад 3. Створимо порожню модель. Виберіть пункт меню File->New і у відкрившомуся вікні виберіть створення порожньої моделі (`Blank Model`),

тобто натисніть кнопку 

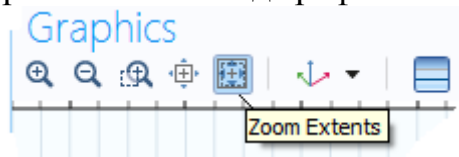
. Клацніть правою кнопкою миші по найвищому вузлу дерева моделі (**Untitled.mph**) і в випадному меню виберіть команду Add Component->1D. Цим ми створюємо одновимірну модель. З'являється вузол **Component 1** і кілька його підвузлів, з яких нас цікавлять вузли **Geometry 1** і **Mesh 1**. Виділіть вузол **Geometry 1** і на стрічці панелей відкрийте закладку `Geometry`, на якій клацніть по кнопці Interval в розділі **`Primitives`**.



В гілку **Geometry 1** додається вузол **Interval 1**. В його ПВ задайте координати лівої і правої кінцевих точок відрізка, наприклад, `-1` і `1`. Щоб намалювати відрізок, клацніть по кнопці Build All Objects вгорі ПВ вузла **Interval 1**.

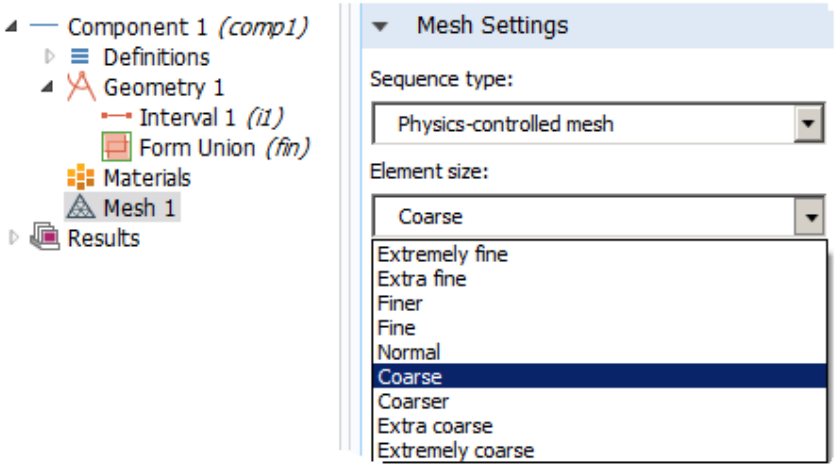


Якщо зображення відрізка вийшло за межі графічного вікна, то натисніть на кнопку Zoom Extents, яка розташована над графічним вікном.

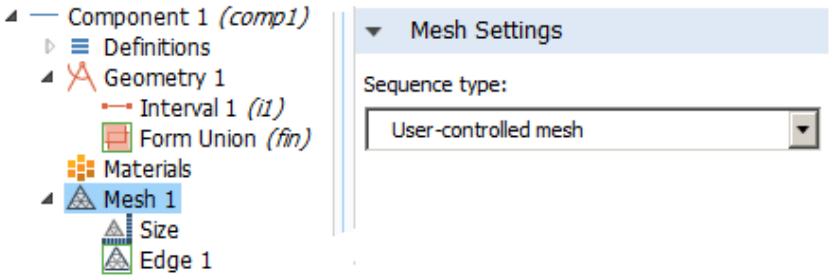


Тепер побудуйте скінченно-елементну сітку на цьому відрізку. Для цього виділіть вузол **Mesh 1** і на стрічці панелей, на закладці `Mesh` натисніть кнопку Build Mesh. У графічному вікні зображення відрізка набуде наступного вигляду.

Кількість точок розбиття визначається довжиною скінченних елементів. Їх зумовлені розміри вибираються в списку **Element Size** вікна властивостей вузла **Mesh 1**.



Якщо в списку **Sequence type** цієї ПВ вибрати `User-controlled mesh`, то в дереві моделі в вузлі **Mesh 1** з'являться два підвузли **Edge 1** і **Size**. В ПВ останнього можна явно задати довжину скінченного елемента.



В списку **Sequence type** залишимо значення `Physics-controlled mesh`, а в списку **Element Size** виберемо значення `Normal`.

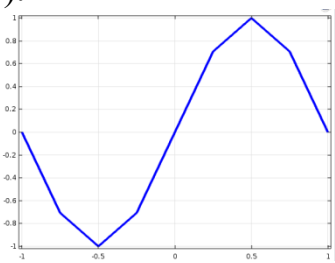
Оскільки вузла **Study** в нашій моделі немає, то виконувати обчислення за допомогою команди Compute непотрібно, і можна відразу перейти до роботи з гілкою результатів **Results**.

В гілку даних **Data Sets** дерева моделі додамо вузол, який буде представляти джерело даних, що складається з нашої скінченно-елементної сітки. Клацніть правою кнопкою по вузлу **Data Sets** і в контекстному меню оберіть Mesh. В ПВ вузла **Mesh 1**, що з'явився в дереві моделі, в списку **Mesh** виберіть `Mesh 1`. Якби в нашій моделі було створено кілька сіток скінченних елементів, то можна було б вибрати будь-яку з них. Перейменуйте вузол **Mesh 1**, надрукувавши в полі **Label** текст `DataMesh`.

Тепер в гілку **Results** додайте графічну групу **1D Plot Group** з вузлом типу **Line Graph**. У списку **Data set** ПВ вузла **1D Plot Group 1** виберіть ім'я `DataMesh`, а в списку **Data set** ПВ вузла **Line Graph 1** виберіть `From parent`. В ПВ вузла **Line Graph 1** в списку **Selection** виберіть `All domains` або, як альтернативу, виберіть `Manual` і додайте індикатор відрізка, клацнувши в графічному вікні по відрізку лівою кнопкою миші.

Далі чинимо так само, як при побудові одновимірних графіків в прикладі 1. У вікні властивостей вузла **Line Graph 1** в розділі **`y-Axis Data`** в полі **Expression** вводимо вираз, наприклад `sin(pi*x)`. В розділі **`x-Axis Data`** в списку **Parameter** вибираємо `Expression`, і в поле **Expression** вводимо `x`. В розділі **`Coloring and Style`** задаємо стиль, колір і товщину графіка, і

натискаємо на кнопку Plot. В графічному вікні з'являється графік функції, побудований по вузлах скінченно-елементної сітки. Щоб продемонструвати те, що графік будується по вузлах, ми використали «грубу» сітку (у вас графік повинен вийти більш гладким).



Побудова графіків поверхонь виконується аналогічно:

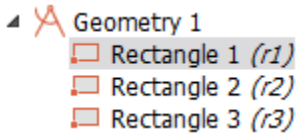
- створюється модель розмірності 2D;
- у вузлі **Geometry** створюється двовимірна область;
- в області будується скінченно-елементна (СкЕ) сітка;
- в гілці **Data Sets** створюється **Mesh** вузол, пов'язаний з СкЕ сіткою;
- створюється група двовимірних графіків, які використовують вузол **Data Sets–Mesh** як джерело даних;
- додається вузол конкретного графічного типу, в поле **Expression** якого вводиться бажаний вираз.

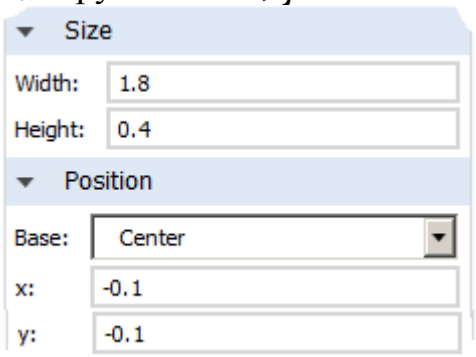
Замість кроків 2 і 3 (створення двовимірної області і побудови в ній СкЕ сітки) інколи використовується двовимірне джерело даних **Grid 2D**.

Приклад 4. Побудуємо графік функції двох змінних над областю у вигляді «скоби» .

Створіть порожню модель. Виберіть пункт меню File–>New і у відкрившомуся вікні виберіть створення `Blank Model` моделі.

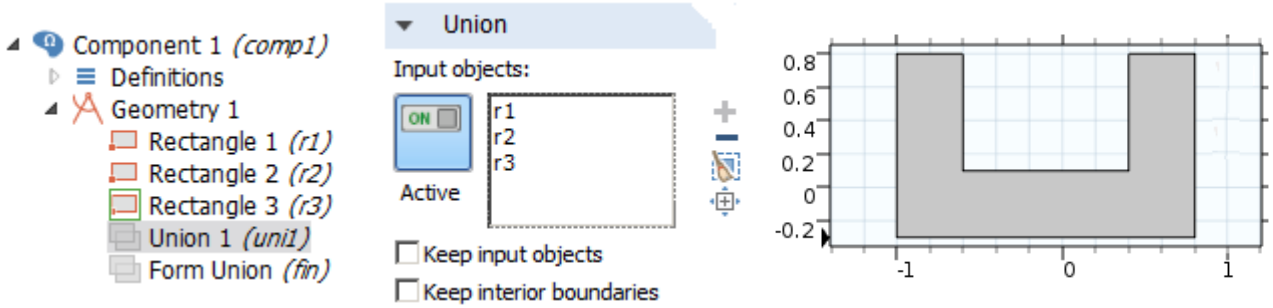
Клацніть правою кнопкою миші по найвищому вузлу дерева моделі (**Untitled.mph**) і в випадному меню виберіть команду Add Component–>2D. З'явиться вузол **Component 1** і кілька його підвузлів, з яких нас цікавлять вузли **Geometry 1** і **Mesh 1**. Виділіть вузол **Geometry 1**, на стрічці панелей відкрийте закладку **Geometry** і на ній клацніть по кнопці Rectangle. У випадному списку виберіть Rectangle (Center) і в графічній області мишею намалюйте прямокутник приблизно такого положення і розміру, як ви бажаєте. В ПВ вузла **Rectangle 1** підправте (якщо треба) параметри прямокутника: задайте розміри **Width** = `1.8`, **Height** = `0.4` і положення центру **x** = `-0.1`, **y** = `0.1`.





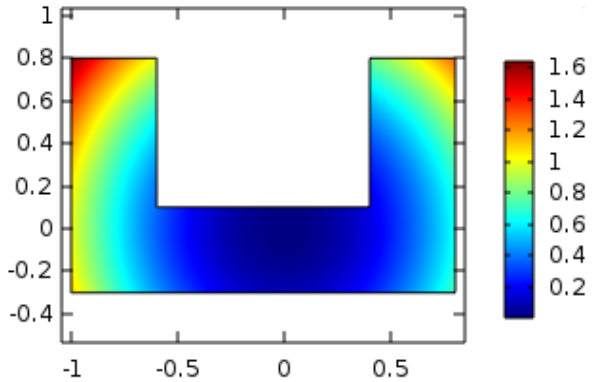
Аналогічно додайте ще два прямокутника. В ПВ вузла **Rectangle 2** введіть **Width** = `0.4`, **Height** = `0.8`, **x** = `-0.8`, **y** = `0.4`. В ПВ вузла **Rectangle 3** введіть **Width** = `0.4`, **Height** = `0.8`, **x** = `0.6`, **y** = `0.4`.

На наступному кроці об'єднайте прямокутники в єдину область, додавши вузол типу **Union**. Для цього клацніть правою кнопкою миші по вузлу **Geometry 1** і в контекстному меню виберіть команду Booleans and Partitions–Union. З'явиться вузол **Union 1**. Активізуйте його і в графічній області по черзі клацніть мишею по кожному з трьох прямокутників. Зніміть перемикач **Keep interior boundaries** і вгорі ПВ натисніть кнопку Build All Objects.

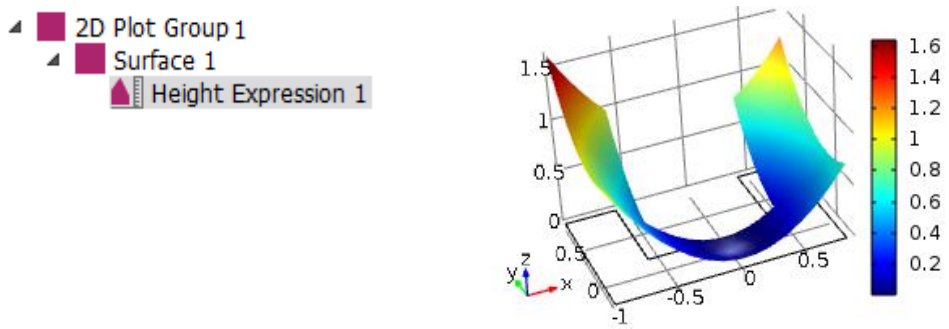


В графічній зоні з'явиться зображення області в формі скоби. На закладці `Mesh` натисніть кнопку Build Mesh, і в області з'явиться скінченно-елементна сітка. В гілку даних **Data Sets** дерева моделі додайте джерело даних, яке буде представляти сітку. Для цього клацніть правою кнопкою миші по вузлу **Data Sets** і в контекстному меню оберіть Mesh. В ПВ вузла **Mesh 1** в списку **Mesh** виберіть `Mesh 1`.

В гілку **Results** додайте графічну групу **2D Plot Group** з підвузлом типу **Surface**. В списку **Data set** ПВ гілки **2D Plot Group 1** виберіть `Mesh 1`, а в списку **Data set** ПВ вузла **Surface 1** виберіть `From parent`. Далі, в ПВ вузла **Surface 1** в розділі **`Expression`** в поле **Expression** введіть вираз, графік якого бажаєте побудувати, наприклад, `x^2+y^2`. Натисніть на кнопку Plot і отримайте наступне зображення.



Щоб побачити графік поверхні в вузол **Surface 1** додайте підвузол типу **Height Expression**.



■

Інколи в полях **Expression** потрібно використовувати «кускові» вирази. В Comsol Multiphysics їх можна створювати в стилі Matlab. Якщо логічний вираз з'являється в аналітичній формулі, то він інтерпретується як +1, якщо умова істинна, або як 0 – якщо умова хибна. Наприклад, вираз $\begin{cases} 0 \ , x \le 0 \\ x^2, x > 0 \end{cases}$ можна записати у вигляді $(x > 0) * x^2$. Також це можна записати як $\text{if}(x > 0, x^2, 0)$. Якщо логічний вираз в першому аргументі оператора `if(...)` істинний, то обчислюється вираз в другому аргументі, інакше обчислюється третій аргумент.

Приклад 5. Побудуємо графік поверхні півсфери, яка задана виразом

$$z = \begin{cases} \sqrt{1 - x^2 - y^2} \ , x^2 + y^2 < 1 \\ \quad 0 \qquad , x^2 + y^2 \ge 1 \end{cases}.$$

Звичайне використання формули $z = \sqrt{1 - x^2 - y^2}$ буде приводити до помилки, оскільки для деяких (x, y) підкореневий вираз буде від'ємним. В COMSOL Multiphysics наш вираз можна записати у вигляді

```
sqrt((x^2+y^2<1)*(1-x^2-y^2)),
```

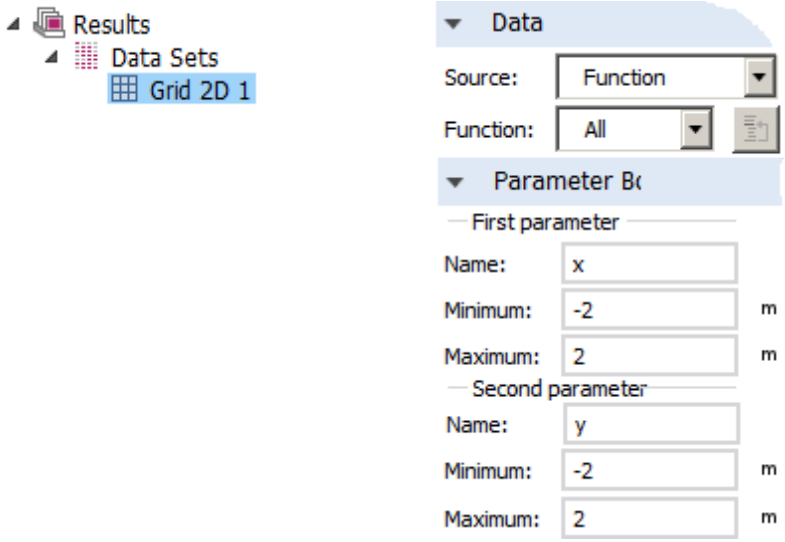
або у вигляді

```
if(x^2+y^2<1,sqrt(1-x^2-y^2),0).
```

Створіть порожню модель. Виберіть пункт меню File->New і у відкрившомуся вікні виберіть створення порожньої моделі (`Blank Model`).

Створіть глобальну функцію $f(x, y) = 1$. Це функція «заглушка», вона не буде використовуватися, але для нормального функціювання джерел даних `Grid` в файлі моделі має бути присутня принаймні одна глобальна функція. Клацніть правою кнопкою миші по вузлу дерева моделі **Global Definitions** і виберіть команду Functions->Analytic. В ПВ створеного вузла в поле **Function name** введіть ім'я функції, наприклад, `f`. В полі **Arguments** через кому перелічіть імена аргументів функції `x`, `y`. В поле **Expression** введіть 1.

В гілку **Results - Data Sets** додайте джерело даних **Grid 2D 1**. Для цього клацніть правою кнопкою миші по вузлу **Data Sets** і в контекстному меню виберіть команду More Data Sets - Grid 2D. В ПВ з'явившегося вузла **Grid 2D 1** в якості джерела даних в списку **Source** оберіть `Function`, в списку **Function** виберіть `All`. Потім задайте діапазони, яким будуть належати змінні `x` і `y`.

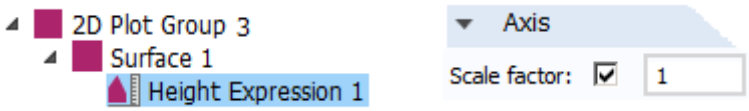


В гілку **Results** додайте вузол **2D Plot Group 1** з підвузлом **Surface 1**. У списку **Data set** ПВ вузла **2D Plot Group 1** оберіть `Grid 2D 1`, а в списку **Data set** ПВ вузла **Surface 1** виберіть `From parent`. В ПВ вузла **Surface 1** в розділі **`Expression`** в поле **Expression** введіть вираз

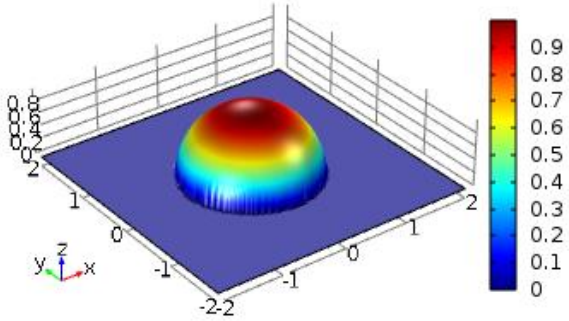
`sqrt((x^2+y^2<1)*(1-x^2-y^2)),`

або вираз `if(x^2+y^2<1,sqrt(1-x^2-y^2),0))`.

Вузлу **Surface 1** додайте підвузол типу **Height Expression**, в ПВ якого в поле **Scale factor** введіть одиницю, попередньо встановивши відповідний прапорець.



Натисніть кнопку Plot і отримайте наступне зображення.



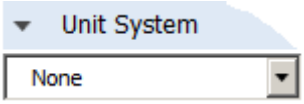
▪

Особливість системи Comsol Multiphysics полягає в тому, що графічні вузли, які використовуються при побудові графіків, повинні бути прив'язані до певного джерела даних (`Data set`). Їх можна створити штучно так, як ми це робили в попередніх прикладах, або розв’язати яку-небудь задачу і її розв’язок використовувати в якості джерела даних для графіків.

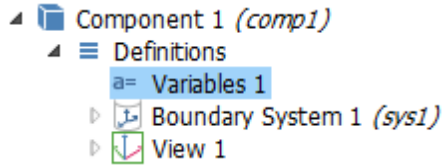
Приклад 6. Побудуємо графік тривимірної кривої, наприклад, гвинтової лінії (спіралі) з рівнянням $x = \cos t$, $y = \sin t$, $z = t/8$. Для цього створимо тривимірну динамічну модель зі змінним `u,v,w`, які будуть залежати від змінної часу `t` по зазначеним формулам. Прив'язуючись до цих змінних, ми побудуємо криву.

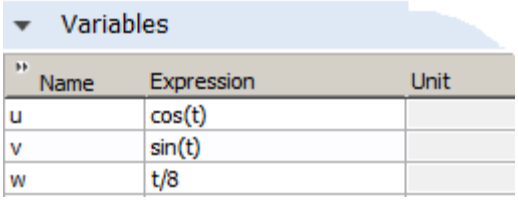
Виберіть пункт меню File -> New і у відкрившомуся вікні виберіть створення порожньої моделі (`Blank Model`).

В ПВ кореневого вузла **Untitled.mph (root)** в списку **`Unit system`** відключіть використання систем одиниць, вибравши зі списку значення `None`.

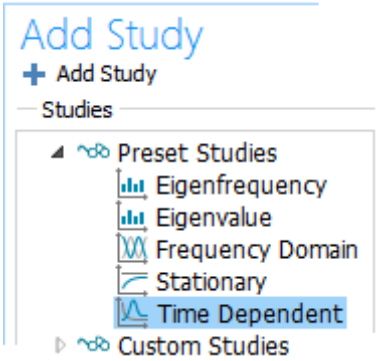


Клацніть правою кнопкою миші по найвищому вузлу дерева моделі (**Untitled.mph (root)**) і в випадному меню виберіть команду Add Component-> 3D. З'явиться вузол **Component 1** і кілька його підвузлів, з яких нас цікавить вузол **Definitions**. Клацніть по ньому правою кнопкою миші і з випадного меню виберіть команду Variables. В ПВ створеного вузла **Variables 1** знаходиться таблиця, призначена для задання змінних. Заповніть її так, як показано нижче.



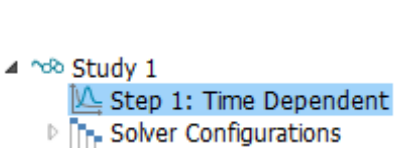


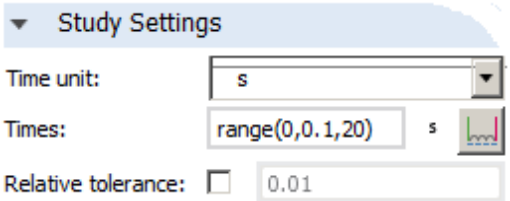
Клацніть правою кнопкою миші по кореневому (найвищому) вузлу дерева моделі (**Untitled.mph (root)**) і в випадному меню виберіть команду Add Study. Система відкриє вікно вибору типу обчислення (`Add Study`).



Виконайте подвійне клацання по рядку `Time Dependent`, який означає, що змінні моделі можуть залежати від часу. Закрийте вікно хрестиком в правому верхньому куті. В дереві моделі з'явиться вузол **Study 1**.

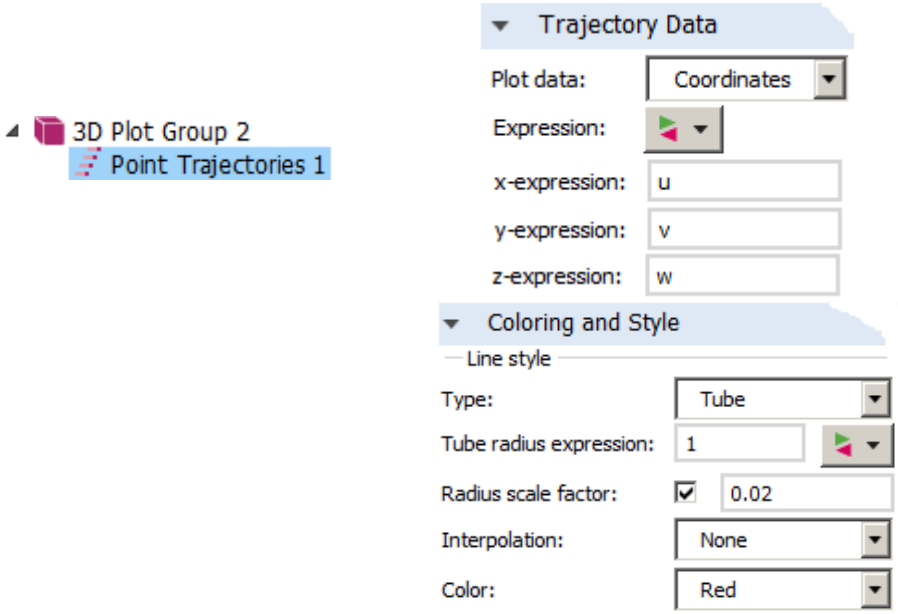
В ПВ вузла **Study 1 - Step 1: Time Dependent** в полі **Times** задайте діапазон зміни параметра `t`. Функція `range(0,0.1,20)` представляє значення параметра `t`, що змінюються від `0` до `20` з кроком `0.1`. Нижче в полі **Relative Tolerance** можна задати відносну похибку.



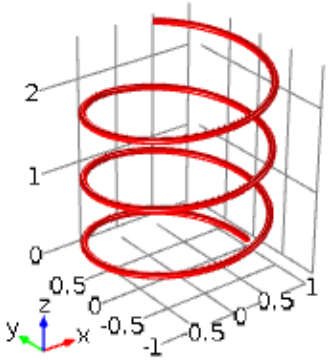


На закладці `Home` натисніть кнопку = Compute . Обчислення моделі буде виконано і в гілці **Results - Data Sets** з'явиться вузол джерела даних **Study 1/Solution 1.** Щоб його використати для побудови кривої, в гілку **Results** додайте вузол тривимірної графічної групи. Для цього клацніть правою

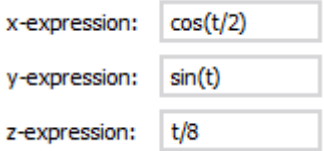
кнопкою миші по вузлу **Results** і в випадному меню оберіть команду 3D Plot Group. Зверніть увагу, що в ПВ створеного вузла **3D Plot Group 1** в списку **Data set** автоматично обрано `Study 1/Solution 1`, тобто нова графічна група прив'язана до джерела даних, що збігається з розв'язком моделі. Перевірте, що в полі **Time(s)** стоїть число `20` – максимальний момент часу із заданого нами діапазону. Клацнувши по вузлу **3D Plot Group 1** правою кнопкою миші, виберіть команду More Plots - Point Trajectories. З'явиться підвузол **Point Trajectories 1**. В його ПВ в поля **x-expression**, **y-expression**, **z-expression** введіть `u,v,w`. Також в цій ПВ в розділі **`Coloring and Style`** задайте параметри зображення кривої так, як показано нижче (показані тільки деякі розділи вікна властивостей).



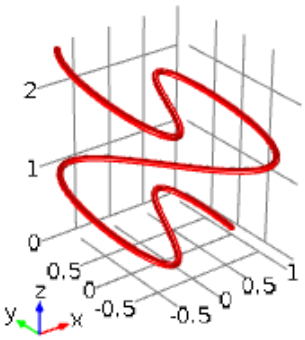
Клацання по кнопці Plot вгорі ПВ вузла **Point Trajectories 1** будує криву.



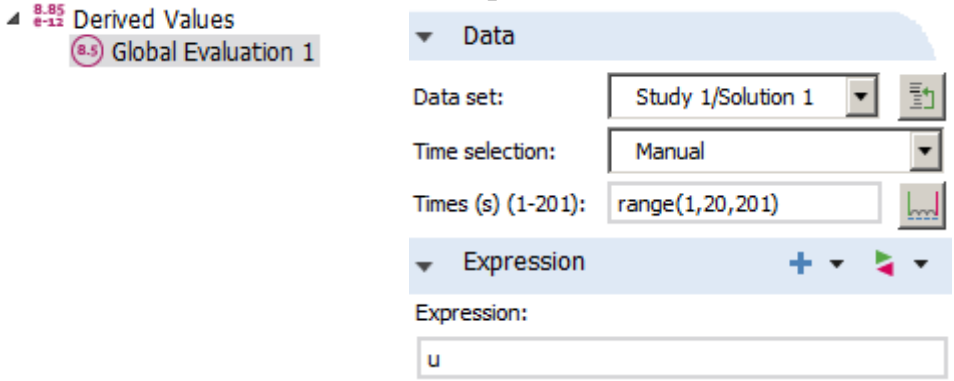
Для побудови кривих необов'язково створювати змінні моделі. Можна безпосередньо в полях виразів графічних вузлів вводити формули. Наприклад, додамо ще один вузол **3D Plot Group 2**. В його ПВ в списку **Data Set** знову виберіть `Study 1/Solution 1`. Додайте в **3D Plot Group 2** підвузол **Point Trajectories 1**. В його ПВ в розділі **`Trajectory Data`** введіть такі вирази



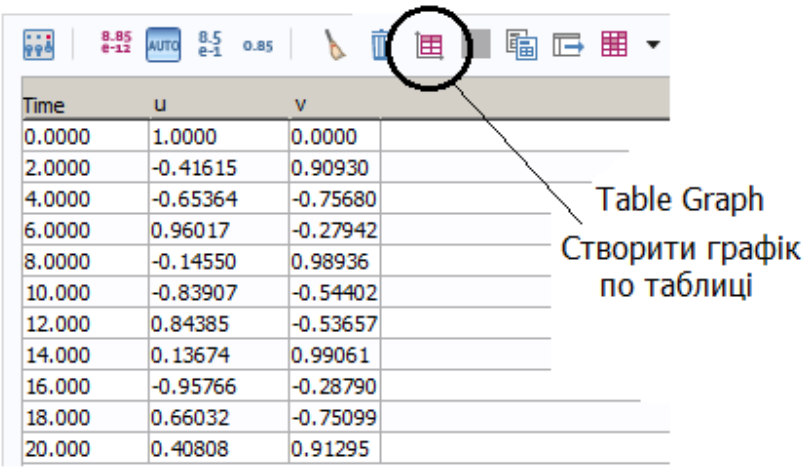
Клацання по кнопці Plot вгорі ПВ вузла **Point Trajectories 1** побудує просторову криву, не пов'язану зі змінними `u`, `v`, `w`. Прив'язка існує тільки до діапазону зміни параметра `t`, який заданий нами раніше у вигляді `range(0,0.1,20)`.



Припустимо тепер, що ви бажаєте надрукувати таблицю значень функцій `u` і `v`. Для цього призначені підвузли гілки **Derived Values**. Клацніть по вузлу **Derived Values** правою кнопкою миші і в випадному меню оберіть Global Evaluation. В ПВ створеного вузла **Derived Values - Global Evaluation 1** в писку **Time selection** виберіть `Manual`, і в полі **Times (s) (1-201)** задайте список номерів моментів часу (не самі моменти), які бажаєте використати, наприклад, `range(1,20,201)`. Це означає, що вибирається кожен двадцятий момент часу.

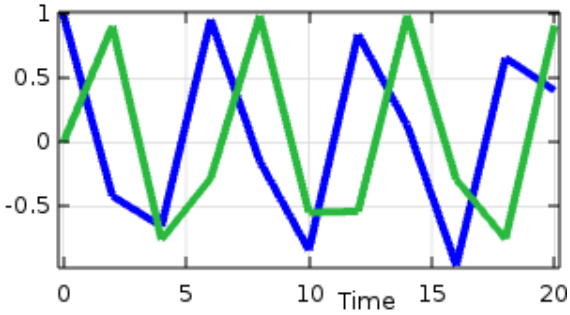


Кнопкою Evaluate, розташованою вгорі ПВ, побудуйте таблицю, яка містить два стовпці: моменти часу і значення функції `u`. Щоб додати в таблицю значення другої змінної, потрібно створити схожий вузол для змінної `v`. Клацніть по вузлу **Derived Values** правою кнопкою миші і в випадному меню оберіть Global Evaluation. В ПВ створеного підвузла **Derived Values - Global Evaluation 2** в полі **Expression** введіть `v`, і в кнопці-списку Evaluate оберіть ім’я вже створеної таблиці **Table 1 - Global Evaluation 1**. Діапазон моментів часу можна не задавати (він буде взятий з таблиці).



Time	u	v
0.0000	1.0000	0.0000
2.0000	-0.41615	0.90930
4.0000	-0.65364	-0.75680
6.0000	0.96017	-0.27942
8.0000	-0.14550	0.98936
10.000	-0.83907	-0.54402
12.000	0.84385	-0.53657
14.000	0.13674	0.99061
16.000	-0.95766	-0.28790
18.000	0.66032	-0.75099
20.000	0.40808	0.91295

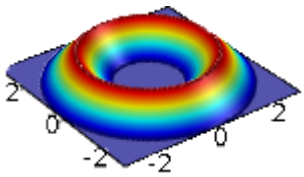
По табличним значенням можна будувати графіки. Кнопка Table Graph, яка розташована над таблицею, створить одновимірну графічну групу і побудує графіки кривих за значеннями в стовпцях цієї таблиці. При цьому перший стовпець буде використаний в якості абсцис вершин ламаних.



Зауваження. Не називайте змінні та шукані функції іменами, що збігаються з назвами просторових координат. Наприклад, в поточному прикладі в таблиці змінних **Variables** не варто використовувати літери `x`, `y`, `z` в якості імен змінних (ми застосували імена `u,v,w`). Інакше деякі з побудов не працюватимуть. Повідомлень про помилки може не бути, але графіки немалюватимуться.

Вправа 1. Використовуючи логічні операції в арифметичному виразі, побудуйте в області $-3 \le x \le 3$, $-3 \le y \le 3$ графік функції

$$f(x,y)=\begin{cases} 0 \quad , & x^2+y^2 \ge 9 \\ 1-\left|\sqrt{x^2+y^2}-2\right| \ , & 1<x^2+y^2<9 \\ 0 \quad , & x^2+y^2<1 \end{cases}$$



В математичних виразах Comsol функція $\sqrt{x}$ позначається як `sqrt(x)`, а $|x|$ як `abs(x)`. Логічні операції **І** та **АБО** позначаються `&&` та `||`. Наприклад, подвійна нерівність $1<x<2$ записується як `1<x && x<2`.

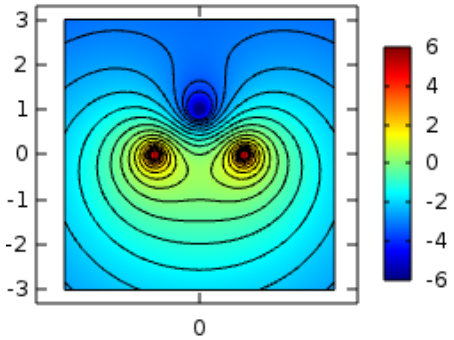
Вправа 2. Графічно зобразити двовимірне електростатичне поле.

Електростатичний потенціал прямої перпендикулярної до площині `z=0`, яка проходить через точку (x_0, y_0), обчислюється за формулою

$$\varphi(\mathbf{r},\mathbf{p}_0) = 2q_0 \ln \frac{1}{\sqrt{(x-x_0)^2+(y-y_0)^2}}.$$

де q_0 – лінійна густина заряду уздовж прямої. Потенціал поля `n` прямих внаслідок суперпозиції силових полів визначається формулою $\varphi = \sum_{i=1}^{n} \varphi(\mathbf{r},\mathbf{p_i})$.

За допомогою двовимірного суцільного та лінійного контурного графіків побудувати поле трьох точкових зарядів $q_1 = 1, q_2 = 1, q_3 = -1$, розташованих в точках $(1,0),(-1,0),(0,1)$.



Для виразу $\varphi(\mathbf{r},\mathbf{p}_0)$ створіть функцію. Натуральний логарифм $\ln x$ в Comsol позначається як `log(x)`.

Щоб отримати наведений вище рисунок, при побудові суцільного контурного графіка обмежте зверху діапазон значень кольорів, а для контурного графіка підберіть значення ліній рівня потенціалу.

1.4. Термінологія, позначення і скорочення.

В цьому параграфі описуються домовленості, які будуть застосовуватися на протязі всього посібника для скороченого запису команд та інструкцій.

В посібнику часто використовуються наступні скорочення:

ПВ , ВВ	панель властивостей, вікно властивостей
ДМ	дерево моделей
СкЕ	скінченні елементи, скінченно-елементний
МСкЕ	метод скінченних елементів
ДР	диференціальне рівняння
ЗДР	звичайне диференціальне рівняння
ДРЧП	диференціальне рівняння в частинних похідних
ГВ	графічне вікно
ДД, НД	джерело даних, набір даних (Data Sets)

В COMSOL Multiphysics команди можуть виконуватися вибором відповідного елемента з головної панелі інструментів (стрічки інструментальних панелей), з випадного меню, що відкривається після клацання правою RMB (Right Mouse Button) кнопкою миші по вузлу ДМ, або клацанням лівою LMB (Left Mouse Button) кнопкою миші по кнопці в ПВ.

При запису назви вузла часто будемо вказувати ім'я батьківської гілки, наприклад, **Data Sets** (якщо потрібно), а потім через тире - ім'я вузла **Grid 1D 1**. Для цих імен буде використовуватися шрифт **Times New Roman жирний**. Таким чином, запис **Data Sets - Grid 1D 1** репрезентує ім'я вузла **Grid 1D 1**, який розташований в гілці **Data Sets** дерева моделі.

Для запису команд ми будемо використовувати різні скорочені позначення. Значок **>** (або **–>**) відокремлює одну команду від іншої. Назва активного вузла ДМ може стояти перед командою і відділятися від неї двокрапкою. Наприклад, запис

Study 1 : RMB > Update Solution

означає, що слід виконати клацання правою кнопкою миші (RMB) по вузлу **Study 1** і в випадному меню вибрати команду Update Solution.

Часто назва активного вузла і команди розносяться на різні рядки, наприклад,

Study 1 :

RMB > Update Solution

Запис LMB > Plot без імені вузла означає клацання лівою (LMB) кнопкою миші по кнопці Plot в ПВ активного вузла. Наприклад, якщо активним є вузол **1D Plot Group 1 - Line Graph 1**, то запис

LMB > Plot

є скороченням інструкції

1D Plot Group 1 – Line Graph 1: LMB > Plot

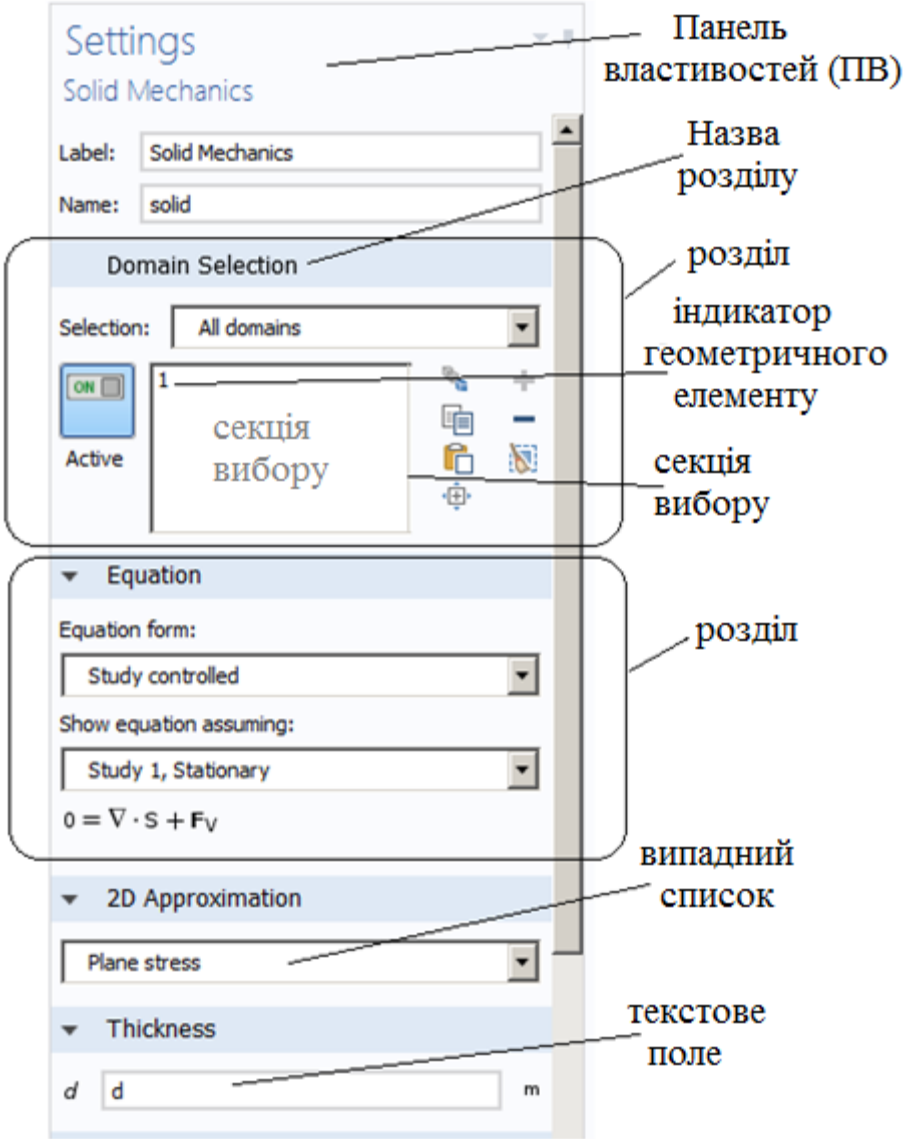
В тексті посібника ім'я вузла і назва команди набираються різними шрифтами. Якщо з контексту ясно, де вузол розташований, то назва батьківського вузла не вказується.

Команда клацання лівою кнопкою миші по кнопці, розташованій над відкритим вікном (в тому числі над ПВ), записується схожим рядком.

Graphics (графічне вікно) : LMB > Play

Тут команда Play являє назву кнопки, яка розташована над графічним вікном.

На наступному рисунку наведено пояснення деяких термінів, які використовуються в посібнику.



Для полегшення виконання прикладів, інструкції в тексті посібника будуть виділятися темно-червоним (або коричневим) кольором, а пояснювальний текст буде набиратися звичайним чорним. Деякі інструкції ми будемо записувати словами. Тоді відповідний текст теж буде виділятися коричневим кольором. Отже, чорне читай, коричневе – виконуй!

Основний потік дій в ході моделювання задається порядком наступності команд. Команди, які йдуть поспіль і не потребують аргументів, будемо записувати коротко. Наприклад, послідовність команд при створенні нової моделі може бути представлена рядком

File –> New –> Blank Model

або, наприклад, довшим рядком

File –> New –> Model Wizard –> 2D –> Mathematics – Classical PDEs – Poisson's Equation(poeq) –> Stationary

Тут ми перелічуємо елементи інтерфейсу, які вибираються на кожному кроці роботи конструктора (`Model Wizard`) при створенні моделі задачі. Стрілки –> відокремлюють назви послідовно вибираних елементів інтерфейсу (фактично команд), а тире розділяє назви вузлів і підвузлів дерева, в якому виконується вибір елемента.

Якщо при виконанні послідовності команд на якомусь кроці потрібно додатково ввести значення або змінити вміст поля, то в послідовність інструкцій після знака + (плюс) вставлятиметься рядок виду **Назва поля** = `значення`. Наприклад, якщо потрібно змінити ім'я шуканої змінної, то послідовність інструкцій записуватиметься наступним чином:

File –> New –> Model Wizard –> 2D –> Mathematics – Classical PDEs – Poisson's Equation(poeq) + Variables = T **–>** Stationary

В цьому запису мається на увазі, що одночасно з вибором інтерфейсу Poisson's Equation (poeq) треба замінити вміст поля **Variables**, в якому буде введено ідентифікатор шуканої змінної T.

При розв'язанні системи рівнянь потрібно вказати кількість невідомих функцій, наприклад, 2, і ідентифікатори двох шуканих функцій. Це буде записуватися схожим чином:

File –> New –> Model Wizard –> 2D –>
Mathematics – Classical PDEs – Poisson's Equation(poeq) +
Number of Dependent Variables = 2 +
Variables = [u1,u2] **–>** Stationary

Застосування двох плюсів означає виконання додаткового вибору в двох полях поточного вікна. Т.ч., в останньому запису мається на увазі, що після вибору Poisson's Equation (poeq) в тому ж вікні буде виконано два додаткових введення **Number of Dependent Variables** = `2` і **Variables** = `[u1,u2]`.

Послідовність команд, які використовуються при створенні нового елементу моделі (тобто нового вузла ДМ), будемо записувати наступним рядком:

`Ім'я панелі` –> **`Назва групи команд`** **–>** Команда

При цьому імена панелей, груп команд (або кнопок-списків) і самі команди будемо розділяти стрілками, і виділяти різним шрифтом. Наприклад, команда

`Results` **–> `Data Set`** - More Data Sets **–>** Grid 1D

означає, що на панелі `Results` в розділі **`Data Set`** слід розкрити кнопку-список More Data Sets і в випадному списку обрати команду Grid 1D.

Якщо перед виконанням команди треба активізувати конкретний вузол, то це може бути записано наступним рядком:

Geometry 1 `:>` `Geometry` **–> `Draw`** **–>** Circle

Значок **`':>'`** відокремлює назву активного вузла від імені команди, яка обирається на головній інструментальній стрічці (в даному випадку на панелі `Geometry`).

Пам'ятайте, що доступ до команд можна отримати різними способами. Один з них полягає у використанні головної інструментальної панелі (стрічки панелей). Інший полягає у виборі команд з контекстного меню, яке відкривається при натисканні правою кнопкою миші по відповідному вузлу дерева моделі. Ми завжди будемо описувати тільки один зі способів, але ви можете використовувати будь-який, зручний для вас.

Наші дії при заповненні панелей властивостей вузлів теж будемо записувати коротко. Наприклад, введення параметрів в ПВ вузла **Data Sets - Grid 1D 1** будемо записувати в такий спосіб

Data Sets – Grid 1D 1:

Data – **Source** = Function

Data – **Function** = All

Parameter Bounds – **Name** = x

Parameter Bounds – **Minimum** = -1

Тут перший рядок репрезентує ім'я активного вузла дерева моделі. Кожний наступний рядок представляє в ПВ цього вузла ім'я заповнюваного поля і значення, що вводиться або вибирається зі списку. Ліворуч від знака рівності стоїть ім'я поля (або списку), яке складається з імені розділу, в якому це поле знаходиться, і самого імені поля. Ім'я розділу відділяється від імені поля знаком тире. Для назви розділу буде використовуватися шрифт **Courier New жирний**, а для імені поля - шрифт **Constantia жирний**. Для значень, які стоять праворуч від знака рівності і вводяться або вибираються зі списку, буде використовуватися шрифт Courier New. Рядки команд для полів, вміст яких не змінюється (використовуються значення за замовчуванням), ми записувати не будемо.

Щоб багаторазово не друкувати назву повторюваних розділів в кожному рядку, ці назви інколи опускаються. Наприклад, наведені вище інструкції ми можемо записувати в такий спосіб

Data Sets – Grid 1D 1:

Data – **Source** = Function

– **Function** = All

Parameter Bounds – **Name** = x

– **Minimum** = -1

Коли налаштування задаються за допомогою перемикачів або радіокнопок, то щоб вказати значення таких елементів ми будемо присвоювати їх іменам логічні константи True або False. Наприклад,

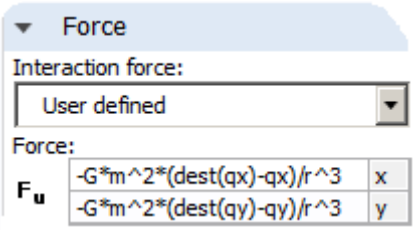
Scale – **Scale factor** = True (*включити*)

Інколи в розділах зустрічаються поля без міток. Для їх позначення будемо використовувати кутові дужки **< >**. Наприклад, запис

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

означає, що в ПВ кореневого вузла ДМ в розділі **Unit System** в списку без мітки слід вибрати значення None.

Іноді мітка відноситься до декількох полів, наприклад, коли вона позначає вектор.



Для розрізнення таких полів ми будемо використовувати мітки в кутових дужках. Наприклад, для скороченого запису вводу даних, показаних на попередньому рисунку, ми будемо використовувати такі рядки

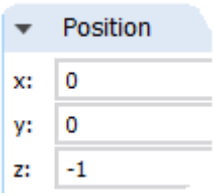
`Force` – **Force – Fu - <x>** = `-G*m^2*(dest(qx)-qx)/r^3`

`Force` – **Force – Fu - <y>** = `-G*m^2*(dest(qy)-qy)/r^3`

Задання значень в три схожі поля одного розділу будемо записувати одним рядком. Наприклад, запис

`Position` – **x, y, z** = `0,0,-1` (*три поля*)

означає, що треба виконати наступні введення



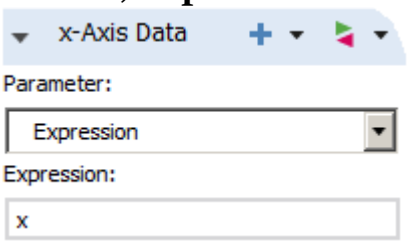
Інструкції введення/вибору значень в різнойменні поруч розташовані поля одного розділу ми також інколи будемо записувати одним рядком. Наприклад, замість двох рядків

`x-Axis Data` – **Parameter** = `Expression`

`x-Axis Data` – **Expression** = `x`

будемо писати один

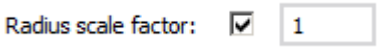
`x-Axis Data` – **Parameter, Expression** = `Expression,x` (*два поля*)



Праворуч від значень ми іноді будемо додавати коментар – текст в дужках. В останній інструкції коментар пояснює факт введення значень в два поля, а не в одно.

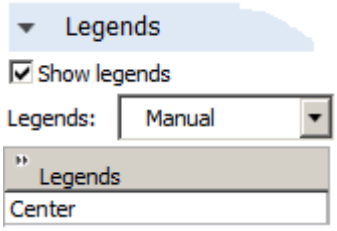
Якщо навпроти поля треба встановити прапорець і ввести значення, то обидві дії будемо записувати одним рядком, наприклад,

`Coloring and Style` – **Radius scale factor** = `True,1`



Коли в схожій інструкції друге значення повинно бути рядком, то його в нашому посібнику будемо записувати в лапках (щоб не виникало непорозуміння), хоча вводити лапки в текстове поле ми не будемо. Наприклад,

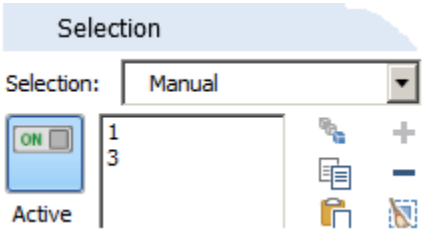
`Legends` – **Show legends, Legends** = `True, 'Center'`



Схожі скорочення будемо використовувати в секціях вибору. Наприклад, інструкція

`Selection` – **Selection,** <> = `Manual;1,3`

позначає, що в розділі **`Selection`** у випадному списку **Selection** обрано значення `Manual`, а нижче в секції вибору – графічні примітиви з номерами/індикаторами `1` і `3`. Тут в якості розділювача ми будемо використовувати крапку з комою та коми.



Коли довжина «скороченої» інструкцій велика, то будемо використовувати запис з перенесенням частини назви команди на наступний рядок.

Results - Data Sets - Grid 3D 2 :

`Paramemter Bounds` –

– **First parameter** – **Minimum, Maximum** = `-1,1` (*два поля*)

– **Second parameter** – **Minimum, Maximum** = `-1,1` (*два поля*)

– **Third parameter** – **Minimum, Maximum** = `-1,1` (*два поля*)

Останнє означає, що групи полів **First parameter, Second parameter, Third parameter** розташовані в спільному розділі **`Paramemter Bounds`**.

Для введення значень в таблиці, які зустрічаються в панелях властивостей, будемо використовувати схожі позначення. Наприклад, введення значень в перший рядок таблиці глобальних змінних **Global Definitions - Variables 2** будемо записувати наступним чином

Global Definitions - Variables 2:

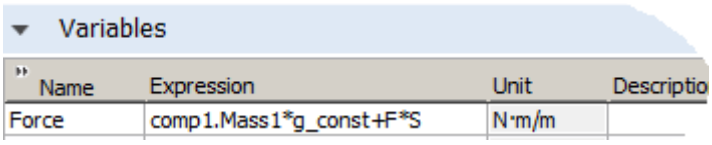
`Variables` – **Name** = `Force`

– **Expression** = `comp1.Mass1*g_const+F*S`

або коротше

`Variables` – **Name, Expression** = `Force, comp1.Mass1*g_const+F*S`

Тут **Name** і **Expression** назви стовпців таблиці, яка знаходиться в розділі **`Variables`**.



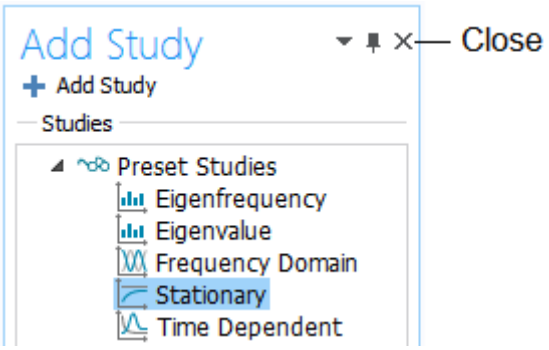
Повторення схожого запису буде означати заповнення наступного рядка таблиці.

Інколи після команди відкривається вікно, в якому треба зробити вибір і негайно його закрити. Наші дії стисло будуть записуватися, наприклад, таким рядком:

Untitled.mph (root): RMB > Add Study = `Add Study ->`

Preset Studies - Stationary ->X

Це означає, що після клацання правою кнопкою миші по вузлу **Untitled.mph (root)** і вибору в випадному меню команди Add Study відкриється вікно `Add Study`, в дереві якого треба обрати елемент `Stationary`.



Вибір виконується в стандартний спосіб (в нашому випадку подвійним клацанням миші по рядку `Stationary` або однократним клацанням по кнопці +Add Study вгорі вікна). Потім вікно `Add Study` закривається хрестиком (`X`) в правому верхньому куті.

Знак '=' (дорівнює) в останній інструкції є ознакою того, що відкривається вікно, назва якого зазначена після рівності.

Після побудови деяких вузлів ми не будемо вказувати їх назву, і будемо відразу заповнювати поля. Це означає, що подальші інструкції стосуються налаштувань створеного вузла. До того ж, одна з таких інструкцій, може змінити поточну назву вузла в дереві моделі. Наприклад, група інструкцій, що створює глобальну аналітичну функцію, може виглядати наступним чином:

Global Definitions : RMB > Functions – Analytic

Function name = `xs`

Definition – **Expression** = `1-abs(t)/2+abs(t-2)/2`

– **Arguments** = `t`

Тут немає вказівки на назву вузла, але вона нам і непотрібна, бо нас цікавить ім'я функції, а воно `xs` тут вказано.

В тих випадках, коли скорочений запис не відповідатиме реальним етапам моделювання, будемо додавати пояснювальний текст. Але кольорова схема тексту посібника зберігатиметься: «чорне читай, коричневе – виконуй!»

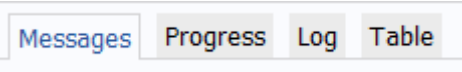
Поступово ми перейдемо до запропонованого стилю опису процесу створення моделей.

1.5. Початкові відомості про моделювання в COMSOL Multiphysics.

1.5.1. Структура моделей COMSOL Multiphysics.

Моделювання в COMSOL Multiphysics починається з вибору розмірності простору і фізичної моделі (стаціонарної або динамічної), тобто зі створення шаблону задачі. Цей шаблон будується в дереві моделі, яке розташовано в панелі конструктора моделей (`Model Builder`). Спочатку модель має ім'я `Untitled`, але пізніше (при збереженні в файл) ви надасте їй осмислене ім'я. Зазвичай праворуч від конструктора моделей знаходиться вікно властивостей активного вузла дерева. Ще правіше розташовано графічне вікно, яке використовується для креслення геометрії моделей або для відображення графіків. Нижче графічного вікна розташована зона з ярликами, що

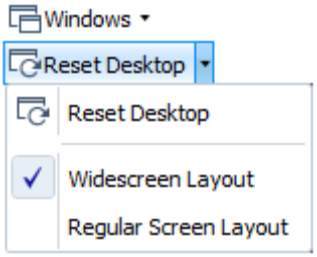
представляють вікна повідомлень (Messages), процесу виконання обчислень (Progress), вікно журналу (Log) і вікно таблиць (Table).



Вікно Messages зазвичай відображає повідомлення про помилки. Вікно Progress активно під час обчислень і відображає поточну інформацію про хід виконання завдання. Вікно журналу Log відображає службову інформацію про результати обчислень. При активізації ярлика Table в цій області з'являються таблиці, які зазвичай містять значення розрахункових величин.

Вікно конструктора моделей (Model Builder), вікно властивостей (Settings), графічне вікно, а також вікна Messages, Progress, Log і Table можна розгорнути на все вікно програми. Для цього потрібно виконати подвійне клацання по корінцю вікна. Повторне подвійне клацання по корінцю поверне вікно до початкового розміру.

Протягом роботи ваш екран може приймати різні види. Щоб повернутися до стандартного способу розташування вікон можна на стрічці панелей відкрити закладку Home, розкрити кнопку-список **Reset Desktop** і вибрати один з можливих варіантів: Widescreen Layout (вікна Model Builder, Settings і Graphics розмістяться на екрані зліва направо), Regular Screen Layout (перші два вікна розмістяться одне над одним зліва екрану, а графічне вікно посяде весь простір праворуч від них) або Reset Desktop, щоб відновити колишній вигляд.

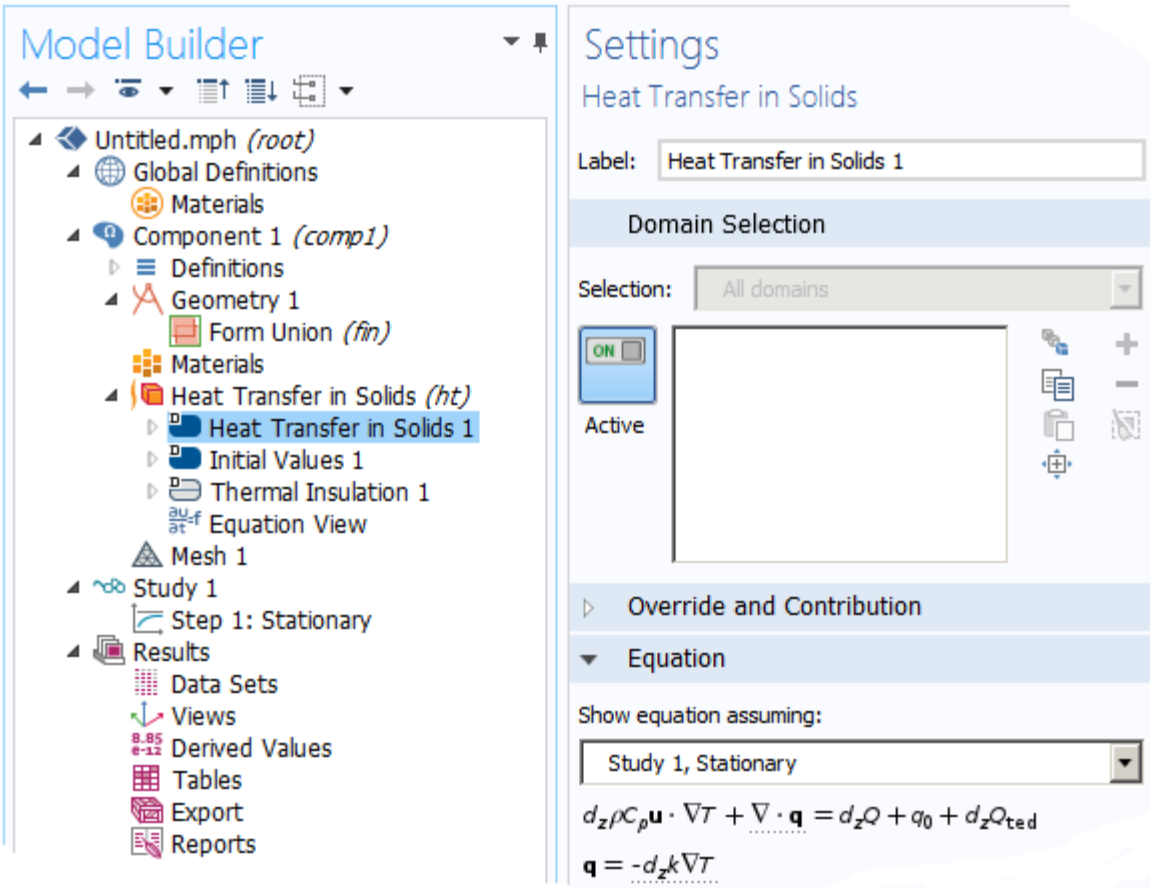


В цьому параграфі ми детально обговоримо послідовність створення та розв’язання двовимірної задачі теплопровідності. Ймовірно, при першому читанні вам не все буде зрозуміло. Познайомтесь з параграфом і поверніться до нього пізніше, коли ми розв’яжемо більше задач, і у вас з'явиться досвід моделювання в COMSOL Multiphysics.

Створіть шаблон моделі двовимірної стаціонарної задачі теплопровідності (heat transfer in solids) наступною послідовністю команд.

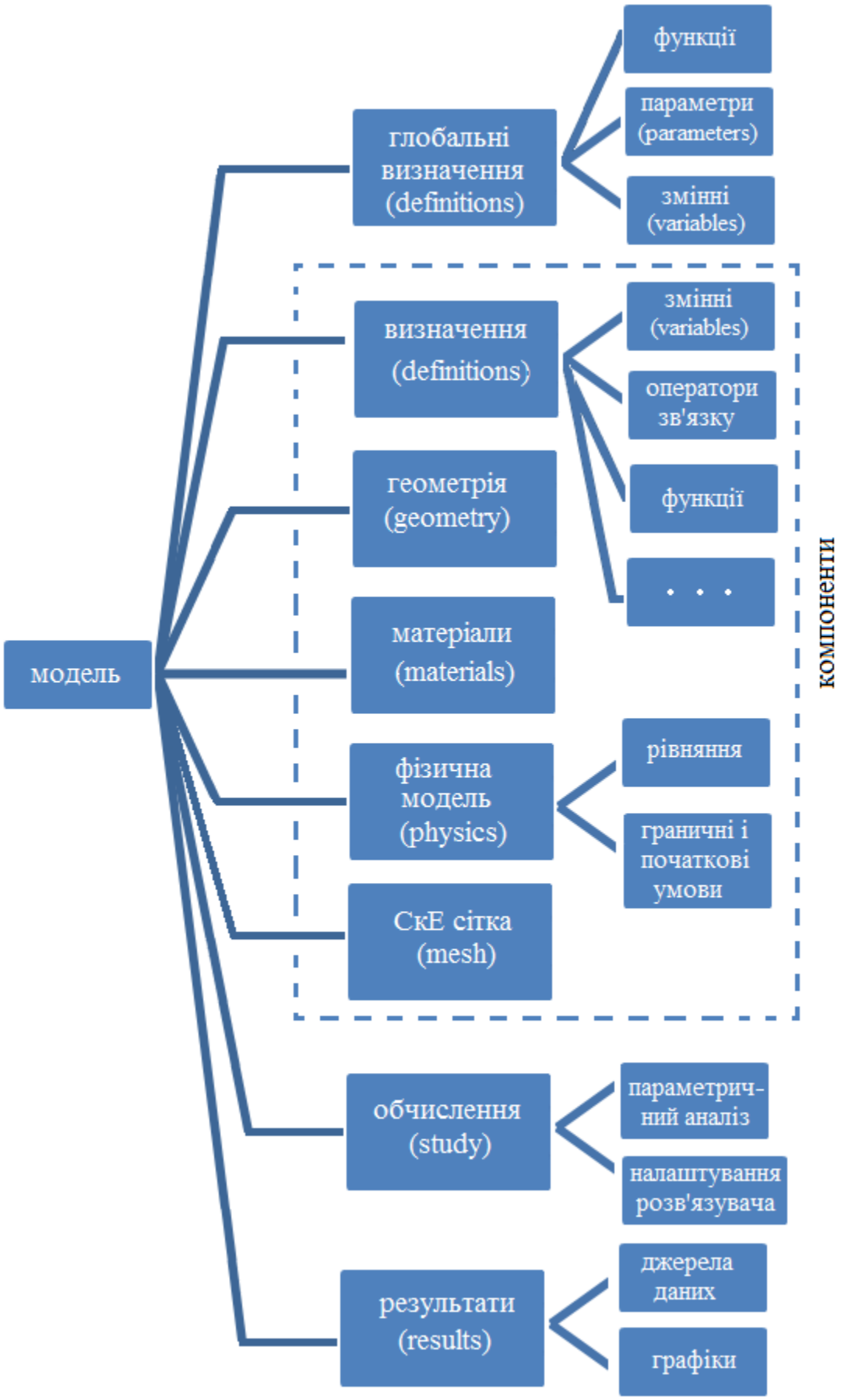
File –> New –> Model Wizard –> 2D –>
Heat Transfer - Heat Transfer in Solids (ht) –> Stationary

Дерево моделі набуде вигляду, показанному на наступному рисунку зліва.



Праворуч від вікна `Model Builder` буде відкрито вікно `Settings` (**в**ікно **в**ластивостей – ВВ, інакше **п**анель **в**ластивостей – ПВ) активного вузла. Воно призначено для налаштування параметрів відповідного елемента моделі. Елементи дерева моделі можуть мати підвузли, у яких можуть бути свої підвузли і т.д. Основними елементами моделі є гілки: **Global Definitions** (визначення), **Component** (компонент), **Geometry** (геометрія), **Materials** (матеріали), тут **Heat Transfer in Solids** (теплопровідність в твердих тілах), **Mesh** (скінченно-елементна сітка), **Study** (обчислення), **Results** (результати обчислень). Замість **Heat Transfer in Solids** може бути обрана будь-яка фізична модель, що доступна в системі. Ці основні елементи завжди присутні в дереві повноцінної моделі, хоча іноді ми будемо використовувати і неповні моделі. Значення деяких елементів зрозуміло з їх назви, сенс інших ми коротко пояснимо нижче.

Головні елементи будь-якої моделі, створеної в Comsol Multiphysics, і їх взаємозв’язки можна описати наступною схемою.



Гілки **Definitions** (визначення) бувають глобальними (для всієї моделі) і локальними для кожного компонента.

Під компонентом розуміється гілка дерева моделі, яка використовується для об'єднання вузлів фізичних процесів, що відбуваються в спільній просторовій області. Наприклад, в тілі може протікати струм, в наслідок цього виділятися тепло та відбуватися деформування. Разом ці процеси можна назвати «мультифізикою». Мультифізика – це жаргон. Під цим терміном ми розуміємо будь-яку замкнену систему диференціальних рівнянь з кількома шуканими величинами різної фізичної сутності та розмірності, що розв’язується як одне ціле в спільній просторовій області.

Зустрічаються задачі, коли потрібно зв'язувати моделі кількох обчислювальних областей. В цьому випадку в COMSOL файлі присутні кілька компонентів (вузлів **Component**), і для їх зв’язку, окрім глобальних змінних, використовуються спеціальні оператори (coupling operators). В нашому посібнику для таких задач ми теж будемо використовувати термін «мультифізика» та інколи словосполучення «міждисциплінарні» задачі.

Отже колекція вузлів декількох фізичних моделей, що використовують спільну геометричну область, створювану в гілці **Geometry**, в термінології Comsol Multiphysics називається компонентом (component). Окрім гілки **Geometry** та гілок фізичних моделей в цю колекцію включаються вузли **Mesh** та **Materials**. На наведеній вище схемі елементи одного компонента виділені пунктиром, і таких компонентів може бути декілька.

Гілки **Definitions** зазвичай містять вузли з таблицями параметрів, змінних, а також вузли операторів і функцій. Параметри – це числові константи, імена яких можна використовувати в декількох місцях моделі. Змінні – це назви величин, які можуть залежати від положення спостережуваної точки. Змінними є просторові координати (наприклад, x, y, z) і шукані величини (наприклад, T - температура). В таблиці змінних користувач може створювати власні змінні з інших, використовуючи припустимі операції (наприклад, диференціювання). Оператори – це математичні конструкції, які зазвичай обробляють величини (шукані функції), досліджувані в моделі. Вузли функцій містять «алгоритми» їх обчислення (аналітичні вирази, таблиці для завдання кускових функцій і т.д.). Кожній функції надається ім'я, і в будь-якому виразі, в якому вона потрібна, ви будете використовувати це ім'я.

Елементи глобальної (зовнішньої) гілки визначень **Global Definitions** можна використовувати в будь-яких місцях моделі. Елементи гілки **Definitions**, визначеній в компоненті, призначені для використання в його межах, але, в разі потреби, можуть використовуватися поза компонентом за допомогою точкової нотації.

Гілка **Geometry** містить інструкції креслення розрахункової області компонента. Вони представляються вузлами ДМ. Безпосередньо в ПВ вузла **Geometry** задаються одиниці вимірювання довжин та кутів.

Використовуючи скорочені позначення, описані раніше, запишемо послідовність команд створення геометрії моделі.

`Geometry` **`-> Primitives`** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

`Size` – **Width, Height** = `3,1` (*два поля*)

Position – **x, y** = 0,0 (*два поля*)

Geometry -> **Primitives** – Circle

Component 1 - Geometry 1 - Circle 1:

Size and Shape – **Radius:** = 0.25

Position – **x, y** = 1,0.5 (*два поля*)

Geometry -> **Primitives** – Circle

Component 1 - Geometry 1 - Circle 2:

Size and Shape – **Radius:** = 0.25

Position – **x, y** = 2,0.5 (*два поля*)

За допомогою операції логічного віднімання видалимо з області прямокутника область правого кола.

Geometry -> **Booleans and Partitions** – Difference

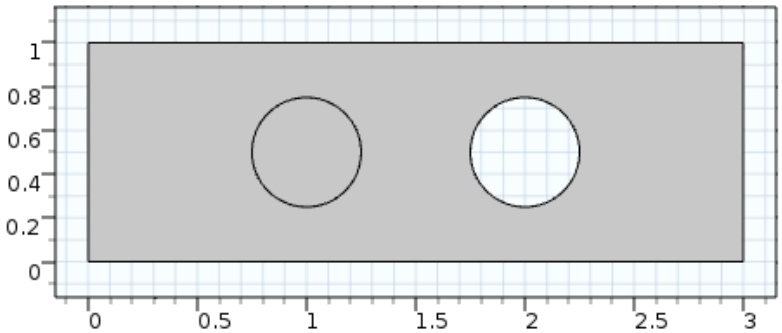
Component 1 - Geometry 1 - Difference 1:

Difference – **Object to add** = r1 (*прямокутник*)

– **Object to subtract** – **Active** = True

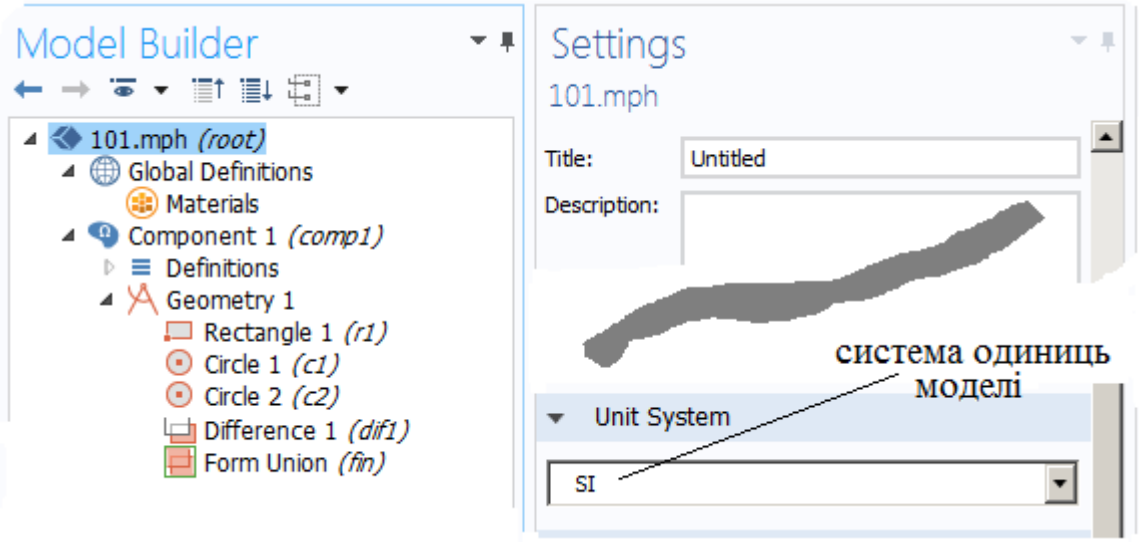
– **Object to subtract** = c2 (*праве коло*)

LMB > Build All Objects

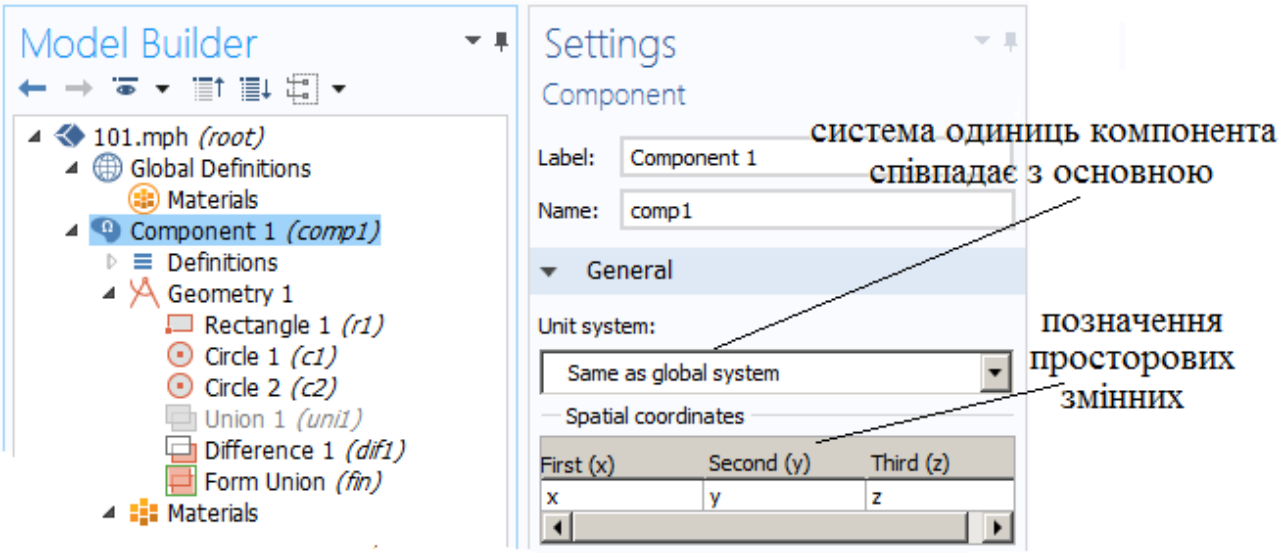


Гілка **Materials** дозволяє вибирати речовини з вбудованою бази, щоб використовувати їх фізичні параметри, а також дозволяє «створювати» власні матеріали.

Всі значення параметрів бажано задавати в системі одиниць, яка використовується в моделі. Дізнатися (вибрати) основну систему одиниць можна в ПВ вузла самого верхнього рівня (**root**) в розділі **Unit System**.



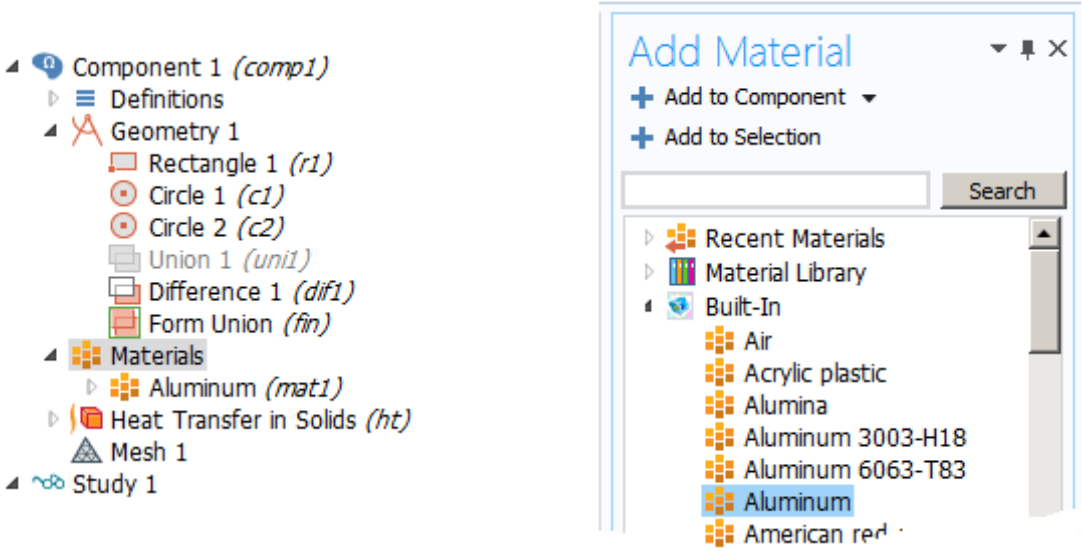
Кожен компонент може мати власну систему одиниць, назву якої можна дізнатися і змінити в ПВ вузла **Component**.



Створіть в моделі вузол матеріалу. Для цього виконайте команду

`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 – Aluminum`.



Клацніть по кнопці +Add to Component, яка розташована вгорі цього вікна, потім закрийте вікно «хрестиком» в правому верхньому кутку. В ДМ з'явиться вузол **Materials - Aluminum (mat1)** і його ПВ з таблицею фізичних параметрів. Галочки навпроти деяких властивостей вказують, що їх значення використовуються в поточній фізичної моделі.

Material Contents

Property	Name	Value	Unit	Property
Heat capacity at constant pressu...	Cp	900[J/(k...	J/(kg·K)	Basic
Thermal conductivity	k	238[W/(...	W/(m·K)	Basic
Density	rho	2700[kg/...	kg/m³	Basic
Relative permeability	mur	1	1	Basic
Electrical conductivity	sigma	3.774e7[...	S/m	Basic
Relative permittivity	epsilonr	1	1	Basic

Тепер клацніть по вузлу «фізики», в нашому випадку по вузлу **Heat Transfer in Solids (ht)**. В його ПВ ви побачите рівняння, які описують вибране фізичне явище.

Equation

Equation form:

Study controlled

Show equation assuming:

Study 1, Stationary

$d_z \rho C_p \mathbf{u} \cdot \nabla T + \nabla \cdot \mathbf{q} = d_z Q + q_0 + d_z Q_{ted}$

$\mathbf{q} = -d_z k \nabla T$

Physical Model

Thickness:

d_z 1[m] m

Інтерфейс **Heat Transfer in Solids** використовується для розв'язання рівняння теплопровідності

$$d_z \rho C_P \frac{\partial T}{\partial t} + d_z \rho C_P \mathbf{u} \cdot \nabla T + \nabla \cdot \mathbf{q} = d_z Q,$$

де T - шукана функція температури, $\mathbf{q} = -d_z k \nabla T$ – тепловий потік, k - коефіцієнт теплопровідності, $\nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right]$ – оператор градієнта (в нашому випадку двовимірний). В рівняння входять матеріальні коефіцієнти: густина ρ, теплоємність C_p, і коефіцієнт теплопровідності k (скалярний або тензорний). В рівняння входять також вектор швидкості **u** речовини і джерела тепла Q різної природи. Для двовимірних задач використовується умовна товщина $d_z = 1[m]$, яку не часто доводиться міняти.

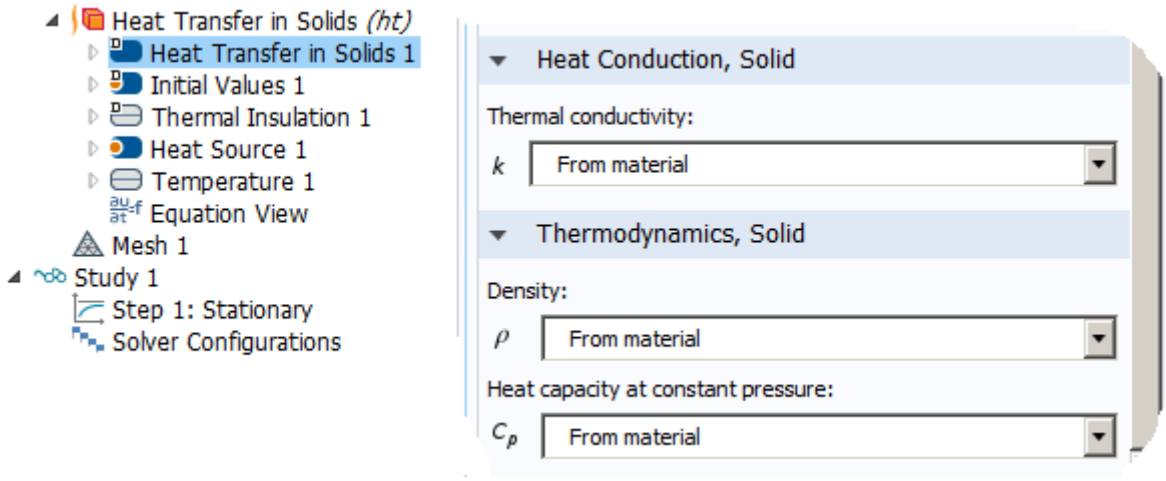
Перший член $d_z \rho C_\rho \frac{\partial T}{\partial t}$ в статичних задачах зникає. Для поля швидкостей **u** і джерел тепла слід створювати власні вузли ДМ. Якщо таких вузлів немає, то відповідні параметри вважаються рівними нулю. В нашому прикладі ми створимо одне джерело тепла потужністю Q_0, яке буде розташовано в області лівого кола. Тому рівняння насправді матиме вигляд $-d_z k(\nabla \cdot \nabla T) = d_z Q_0$, і параметр d_z не буде впливати на розв'язок.

Наведемо позначення, які використовуються під час запису рівнянь в вікнах «фізики».

$\nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right]$	оператор градієнта
$\nabla u = \mathrm{grad}\, u$	градієнт скалярної функції u
$\nabla \cdot \boldsymbol{u} = \mathrm{div}\, \boldsymbol{u}$	дивергенція вектор-функції $\boldsymbol{u}$
$\nabla \times \boldsymbol{u} = \mathrm{rot}\, \boldsymbol{u}$	ротор вектор-функції $\boldsymbol{u}$

Тут же в ПВ вузла **Heat Transfer in Solids (ht)** в розділі **`Domain Selection`** вказується індикатор (номер або номери) області, в якій буде розв'язуватися задача. Якщо геометрія складається з декількох підобластей, то можна вибрати всі або тільки частину з них. В ПВ вузла фізики внизу є розділ **`Discretization`**, в якому вибирається гладкість базисних функцій МСЕ, використовуваних при апроксимації розв'язку. Ви можете вибрати `linear` (лінійну), `quadratic` (квадратичну) або більш високу ступінь. Для більшості «фізик» за замовчуванням використовується квадратична апроксимація. У розділі **`Dependent Variables`** задаються імена змінних шуканих величин. Зазвичай для задач теплопровідності це `T`, а для математичних задач - `u`. Ви можете вказати інше ім'я.

Коефіцієнти рівняння задаються в ПВ вузла **Heat Transfer in Solids - Heat Transfer in Solids 1**. Якщо в цій панелі нічого не змінювати, то вони автоматично будуть взяті з параметрів матеріалу.



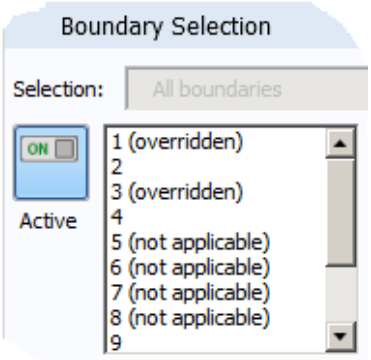
Граничні значення за замовчуванням задаються у вузлі **Zero Flux** і, як правило, зображають нульовий потік (тепла, речовини і т.д.) через границю. В фізичних інтерфейсах цьому вузлу зазвичай дається інша назва. В нашому прикладі автоматично створений вузол **Thermal Insulation 1** (теплоізоляція) задає нульовий тепловий потік на всіх ділянках границі. В більшості випадків вам буде потрібно це змінити. Для цього необхідно додати нові вузли, що представляють інші типи граничних умов, вказати в їх секціях вибору індикатори ділянок границі, на яких вони діють, і задати відповідні значення.

Задамо на лівій і верхній стороні прямокутника сталу температуру 293.15 [K]. Для цього виконайте команду

`Physics` **`-> Boundariy -`** Boundaries **->**Temperature

Активуйте створений вузол **Temperature 1**, натисніть кнопку Active в розділі **Boundary Selection** (якщо вона ще не активна) і в графічному вікні клацніть по лівій і верхній сторонам прямокутника. Їх індикатори (номери) 1 і 3 додадуться в секцію вибору **Selection**. В розділі **Temperature** в полі $\mathbf{T_o}$ залиште значення 293.15 [K].

Якщо зараз відкрити ПВ вузла **Thermal Insulation 1**, то ви побачите, що для ділянок границі з номерами 1 і 3 гранична умова «теплоізоляція» перевантажена (overridden).



Вузол **InitialValues** створюється автоматично і містить початкові (зазвичай нульові) наближення для нелінійного розв'зувача стаціонарних задач, або початкові умови для динамічних задач. В статичних задачах вузол слід використовувати, коли початкову умову не можна задавати нульовою, наприклад, коли температура з'являється в рівнянні у вигляді 1/T. В такому випадку для початку ітераційного процесу слід використати інше початкове значення. В нашому прикладі за замовчуванням початкове значення задано температурою 293.15 [K]. Залиште його без змін.

Рівняння теплопровідності може мати члени, які моделюють внутрішні джерела тепла. У нас в правій частині рівняння стоїть член $d_z \cdot Q$, де Q являє об'ємну густину внутрішніх джерел тепла. Щоб їх врахувати, потрібно створити відповідний вузол ДМ і задати в його ПВ параметри джерела тепла. В нашому прикладі зоною, в якій буде діяти внутрішнє джерело тепла, є ліве коло (для цього воно і створювалося).

Побудуйте джерело тепла.

Physics **-> Domain - Domains ->** Heat Source

В ПВ створеного вузла **Heat Transfer in Solids (ht) - Heat Source 1** в секцію вибору помістіть індикатор лівого кола. Для цього в графічному вікні підведіть покажчик миші до кола. Якщо воно не змінить колір, то крутніть колесо миші і, коли коло змінить колір, клацніть по ньому. Номер 2 (індикатор кола) з'явиться в секції вибору. Потім введіть інші параметри джерела тепла:

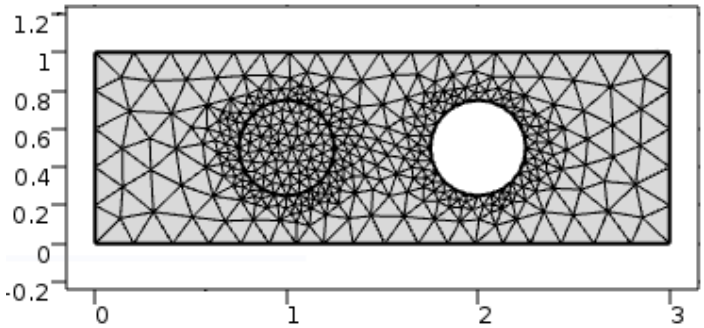
Heat Source – **General source** = True (*включити радіокнопку*)

Heat Source – $\mathbf{Q_o}$, < > = User defined, 100000

Тут $\mathbf{Q_0}$ являє потужність джерела тепла (кількість енергії, що виділяється в одиниці об'єму за одиницю часу). Сумарна потужність джерела в нашому випадку дорівнює

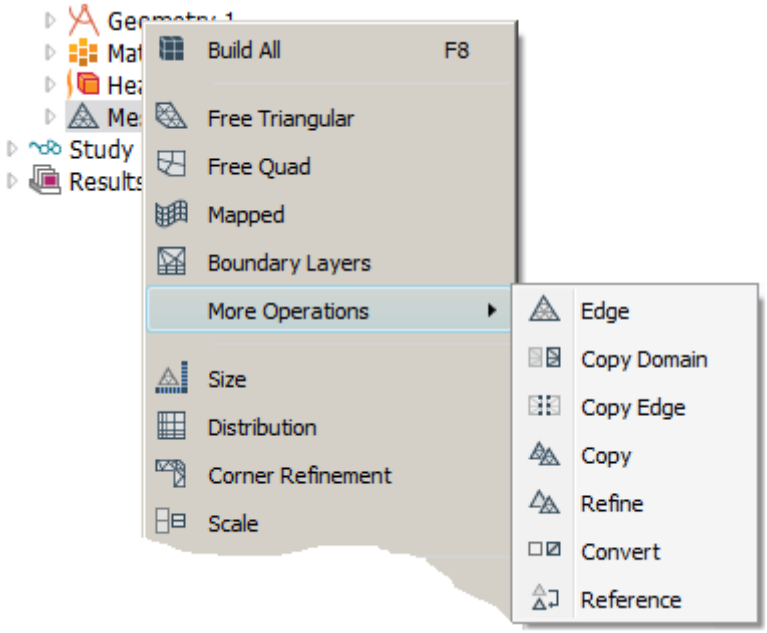
$$Q_\Sigma = Q_0 \cdot \pi r^2 = 100000[W/m^3] \cdot \pi \cdot (0.25[m])^2 = 19634.95[W/m].$$

Важливим елементом моделі, що істотно впливає на точність обчислень, є вузол **Mesh** (скінченно-елементна сітка). Є можливість вибирати обумовлену сітку або проектувати власну. За замовчуванням вибирається обумовлена (`Physics-controlled mesh`) сітка розміром `Normal`. Якщо активувати вузол **Mesh 1** і в його ПВ клікнути по кнопці Build All, що розташована вгорі цієї панелі, то буде згенерована СкЕ сітка, яка показана на наступному рисунку.



В «серйозних» задачах починайте обчислення з сітки `Normal` і поступово робіть її більш дрібною доки розв'язок не перестане змінюватися.

Якщо натиснути правою кнопкою миші по вузлу **Mesh**, стануть доступними інструкції Free Triangular, Free Quad(rilateral), Mapped і Boundary Layers.



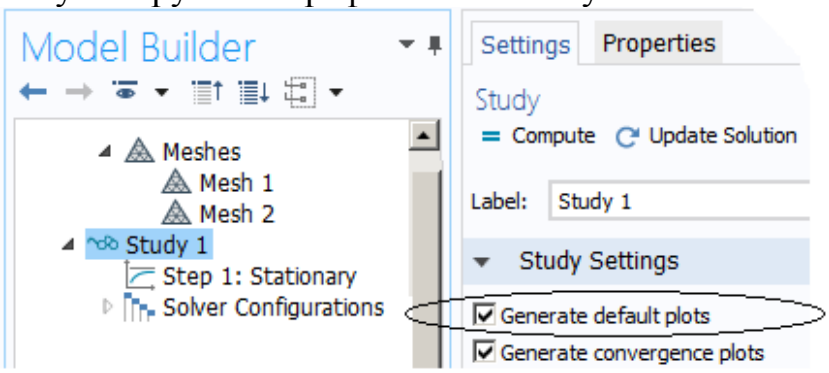
Перші дві покривають область (двовимірну) трикутною або чотирикутною сіткою. `Mapped` сітка покриває область чотирикутними елементами, які «дотримують» форму границі. Виберіть один з методів покриття, і виконайте команду Build All. Інструкція Refine дозволяє згустити існуючу сітку у всій розрахунковій зоні або тільки в деякій її підобласті. Для цього в ПВ вузла **Mesh** потрібно перейти в режим `User-controlled mesh`, а потім викликати команду Refine з контекстного меню або з закладки `Mesh` стрічки панелей. Залишимо в нашому прикладі «нормальну» сітку.

Обчислювання моделі виконується за командою Compute. Її можна викликати декількома способами: клацанням правою кнопкою миші по вузлу **Study 1** і вибором в контекстному меню пункту Compute, функціональною

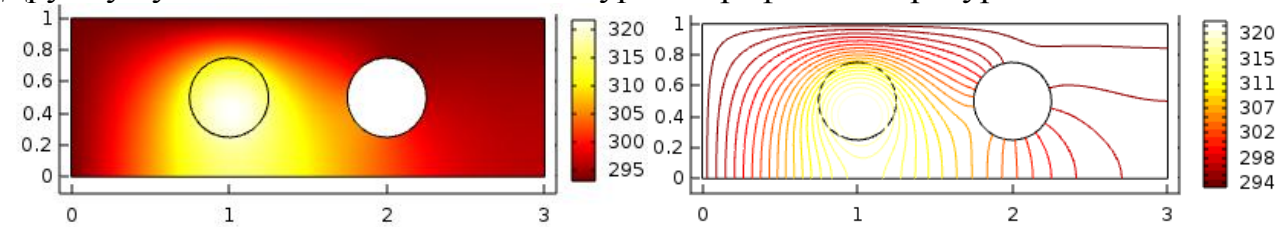
клавішею F8, або з панелі інструментів викликом інструкції Home – **Study** – Compute.

Здійсніть обчислення клавішею F8. У правому нижньому кутку головного вікна ви побачите індикатор виконання.

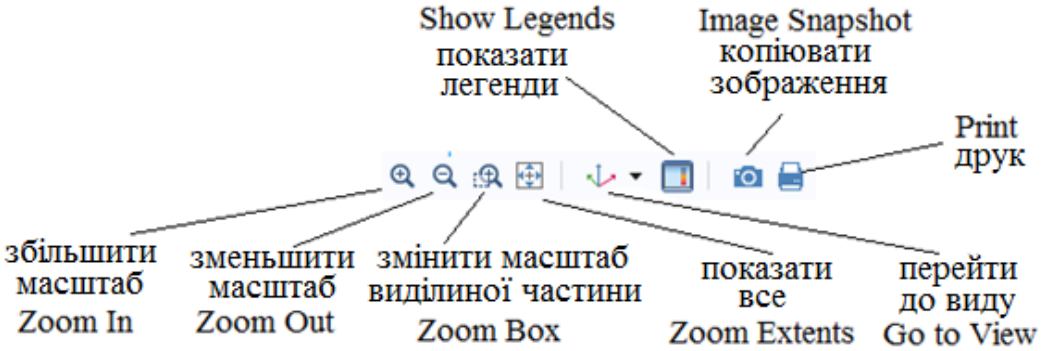
Якщо з якої-небудь причини виникне помилка, то з'являється вікно повідомлень. Крім цього, інформація про помилку записується в ПВ вузла **Study 1 - Solver Configurations - Solution 1 - Stationary Solver 1 - Information - Error 1**. Нагадаємо, що тут тире розділяє назви вузлів і підвузлів ДМ. Якщо помилок немає, то зазвичай автоматично створюються вузли з графіками розв'язків. В ПВ кожного вузла **Study** є прапорець **Generate default plots**, що встановлює ознаку генерування графіків за замовчуванням.



В нашому прикладі буде створено дві графічні групи (типу 2D Plot Group): **Temperature (ht)** і **Isothermal Contours (ht)**. Перша буде містити вузол **Surface 1**, що представляє графік температури з функціональним забарвленням. Друга у вузлі **Contour 1** містить контурний графік температури.



Над кожним графічним вікном розташована панель управління графіками. Її іконки дозволяють збільшити або зменшити зображення (Zoom In, Zoom Out, Zoom Box, Zoom Extents), зберегти його в файл чи буфер обміну (Image Snapshot), або надрукувати.

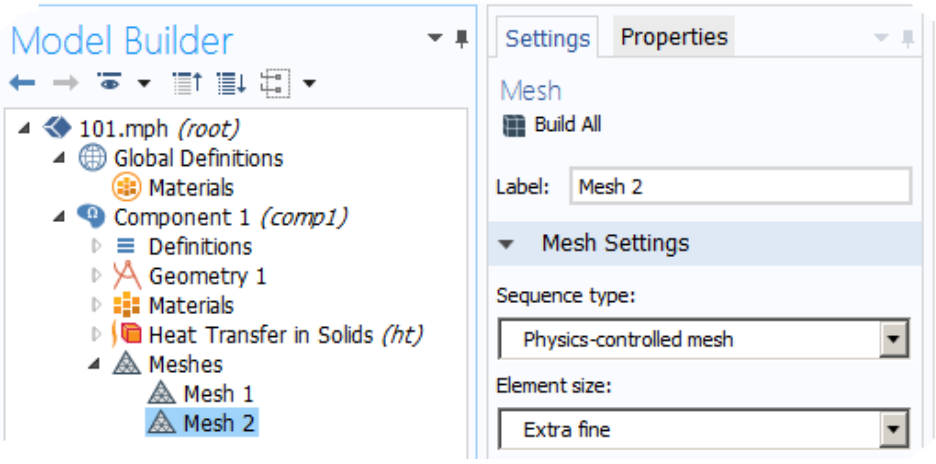


Двовимірні зображення можна змінювати за допомогою кнопок миші:

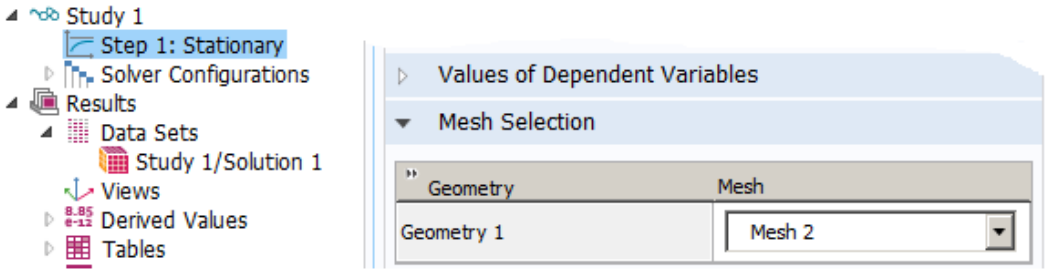
- натискання правої кнопки миші в графічній області і переміщення миші зсовує зображення;
- натискання середньої кнопки миші в графічній області і переміщення миші масштабує зображення;

Зображення можна повернути до стандартного виду, клацнувши по іконці Go to View, і вибравши в списку один із запропонованих «видів».

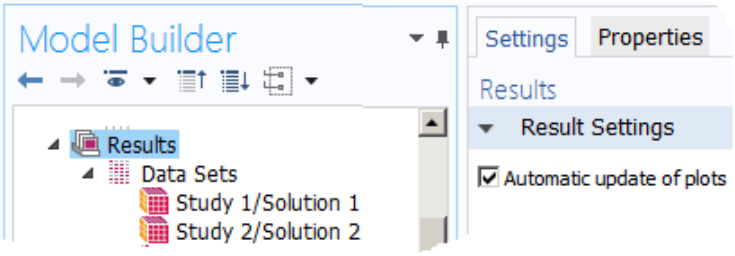
В компоненті можна мати кілька СкЕ сіток і виконувати обчислення для кожної з них окремо. Обидва попередніх графіка побудовані з використанням «нормальної» СкЕ сітки. Ви можете повернутися назад і створити другу більш дрібну сітку. Наприклад, клацніть правою кнопкою миші по вузлу **Component 1** і в контекстному меню виберіть команду Add Mesh. В гілці **Meshes** буде створений вузол **Mesh 2**, в ПВ якого ви можете задати Extra fine розмір нової сітки.



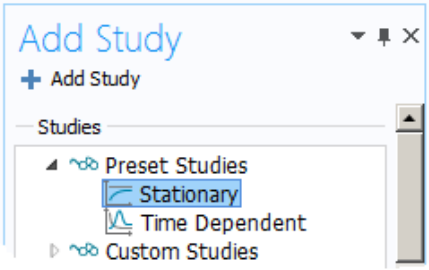
Після цього відкрийте ПВ вузла **Study 1 - Step 1: Stationary**, і в розділі **Mesh Selection** виберіть другу сітку Mesh 2. Повторіть обчислення моделі, натиснувши на клавішу F8.



Результати розміщуються в наборі (джерелі) даних **Study 1/Solution 1**. Разом з цим будуть оновлені графіки **Temperature (ht)** і **Isothermal Contours (ht)**, які використовують ці дані. Автоматичне оновлення графіків задається установкою прапорця **Automatic update of plots** в ПВ вузла **Results**.



Вибирати різні сітки по черзі не завжди зручно. Якщо ви хочете, щоб обидва результати існували одночасно, то в моделі можна створити другий вузол **Study**. Для цього виділіть кореневий вузол моделі **root** (самий верхній вузол ДМ), клацніть по ньому правою кнопкою миші і в контекстному меню виберіть команду Add Study. Система відкриє вікно вибору типу дослідження (Add Study). В нашому випадку це може бути стаціонарне або динамічне обчислення.

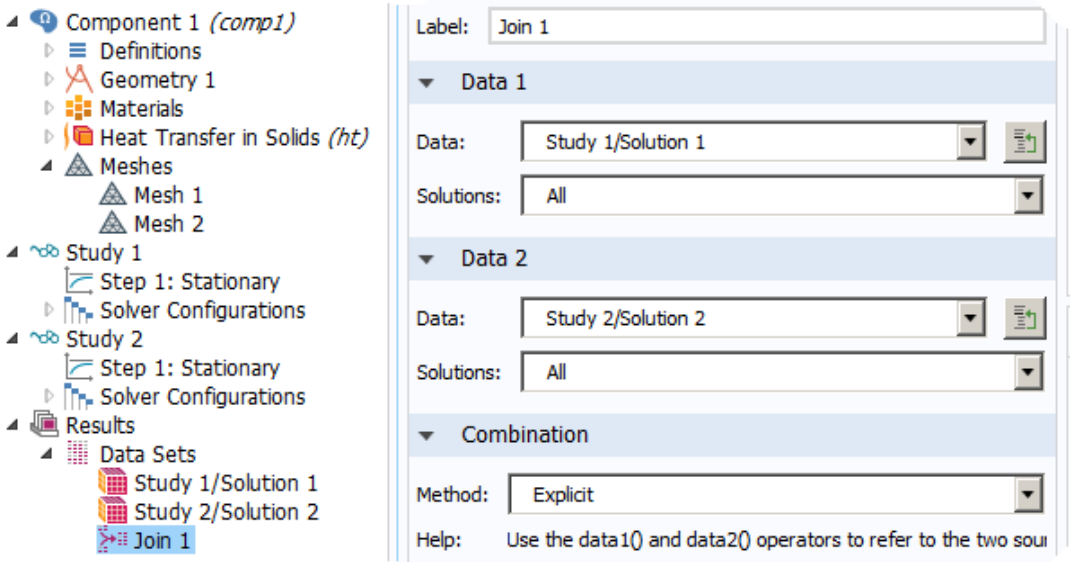


Виконайте подвійне клацання миші по рядку Stationary і закрийте вікно хрестиком в правому верхньому куті. В дереві моделі з'явиться вузол **Study 2**. Активуйте його підвузол **Step 1: Stationary**, і в ПВ в розділі **Mesh Selection** виберіть Mesh 2. Виконайте обчислення Study 2, наприклад, клавішею F8 (коли активний вузол Study 2). Потім поверніться у вузлі **Study 1** до першої сітки Mesh 1 і переобчисліть цю версію задачі (клавішею F8, коли активний вузол Study 1). Тепер в гілці даних **Results - Data Sets** будуть присутні два набори даних **Study 1/Solution 1** і **Study 2/Solution 2**, які представляють два розв'язки побудовані на різних сітках. Зверніть увагу, що кожне дослідження **Study** будує по два графіка **Temperature (ht)** і **Isothermal Contours (ht)**. При цьому в назви графіків, побудованих після другого обчислення, будуть додані одиниці.

Нашою метою є побудова графіка різниці двох розв'язків. Для цього в гілці **Data Sets** створіть ще один набір даних, клацнувши правою кнопкою миші по вузлу **Data Sets**, і в контекстному меню вибравши команду Join. Тобто виконайте інструкцію

Data Sets: RMB > Join

Вузли типу Join дозволяють комбінувати розв'язки.



В ПВ створеного вузла **Join 1** в розділі **Data 1** виберіть перший розв'язок, а в розділі **Data 2** - другий (попередній рисунок праворуч). У розділі **Combination** в списку **Method** виберіть Explicit. Це означає, що в полях виразів, в яких будуть використовуватися ці дані, до першого набору даних можна звертатися за допомогою оператора data1(змінна), наприклад, data1(T). Аналогічно, звернення до другого набору даних буде виконуватися з використанням оператора data2(змінна). Таким чином, щоб створити графік різниці двох розв'язків потрібно побудувати графік виразу data1(T) – data2(T). Виконаємо це за допомогою наступних інструкцій.

Results **-> Plot Group ->**2D Plot Group

Results - 2D Plot Group 5:

2D Plot Group 5-> Surface

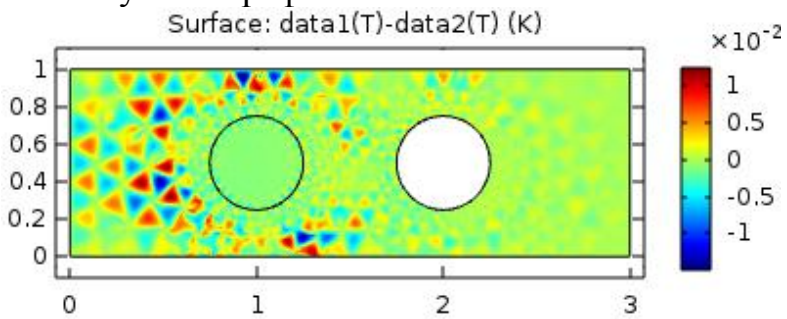
2D Plot Group 5 – Surface 1:

Data – **Data set** = Join 1

Expression – **Expression** = data1(T)- data2(T)

LMB > Plot

Буде побудований наступний графік.



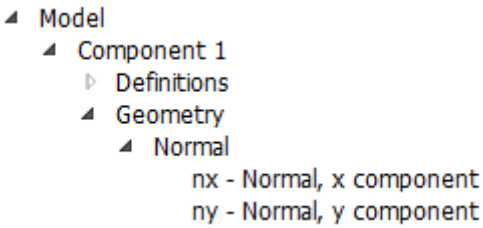
Зі шкали кольорів, розташованій праворуч, видно, що максимум різниці розв'язків приблизно дорівнює 0.015[K]. При цьому слід очікувати, що другий розв'язок точніший за перший, оскільки побудований на більш дрібній сітці.

Зауваження. Графік тієї ж різниці розв'язків можна було б побудувати дещо по-іншому. Для цього можна було б створити такий же вузол Join, але в списку **Combination** – **Method** можна вибрати значення Difference (Різниця). Використання виразу T в вузлі **Results - 2D Plot Group 5 - Surface 1**, який «спирається» на цей новий набір даних **Join 2**, означало б обчислення різниці температур першого і другого розв'язків.

При обчислюванні фізичних величин, таких як тепловий потік, потрібно записувати математичні вирази, в які входять імена різноманітних змінних. Деякі імена ми вже використовували. Наприклад в нашій задачі T являє температуру. Такі імена створюються разом з вибором прикладного режиму, в нашому випадку Heat Transfer in Solids. Імена Tx і Ty використовуються для позначення частинних похідних скалярного поля температури. Але в прикладних режимах створюється багато інших змінних, які можна використовувати в виразах.

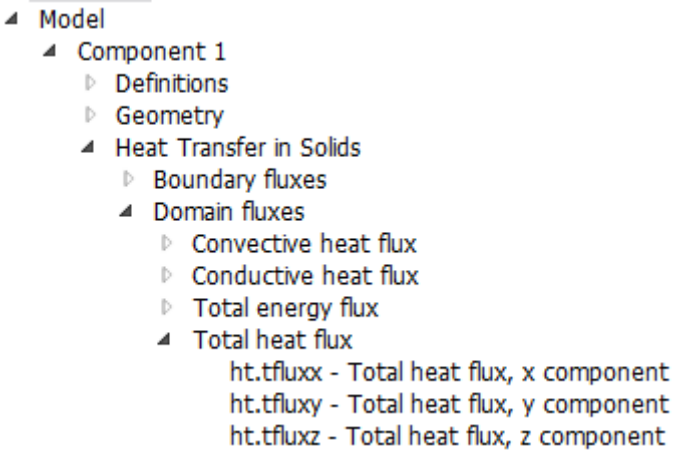
Деякі змінні COMSOL Multiphysics доступні для використання незалежно від діючого прикладного режиму. Серед них є змінні, які пов'язані з

геометричними властивостями об'єктів моделі. Наприклад в двовимірних задачах ім'я `s` являє параметр вздовж кривої, а імена `tx,ty,nx,ny` представляють координати одиничних векторів дотичної і нормалі до кривої. Коли задається вираз на кривій, наприклад гранична умава, то такі імена можна використовувати у відповідному полі **Expression**. Допустимі імена знаходяться в списку готових величин. Він відкривається кнопкою Replace Expression, яка розташована праворуч над кожним полем **Expression**. Наприклад, імена змінних, що представляють координати вектора нормалі кривої, знаходяться у списку/дереві готових величин в гілці `Model – Component 1 – Geometry –Normal`.



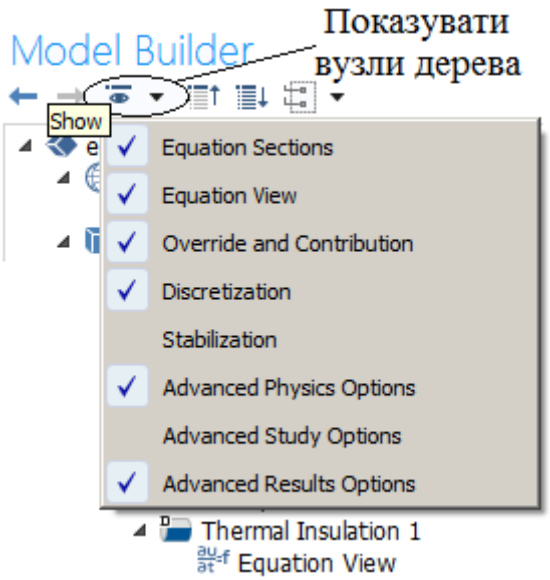
Припустимо, що ми бажаємо обчислити сумарний тепловий потік через межу джерела тепла (лівого кола). Для цього потрібно обчислити інтеграл $\oint_C \mathbf{q}\cdot\mathbf{n}\,ds$, де C – коло (крива інтегрування), $\boldsymbol{q}$ – вектор густини теплового потоку (тепловий потік віднесений до одиниці довжини), $\boldsymbol{n}$ – вектор нормалі до кривої.

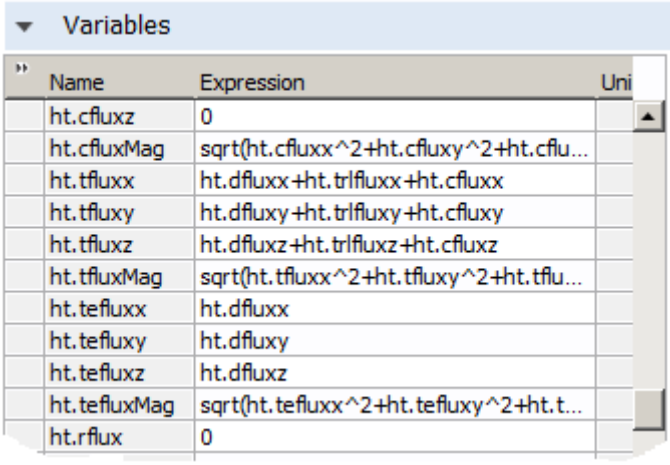
Позначення для координат вектора $\boldsymbol{q}$ можна знайти в тому ж списку/дереві вікна готових величин в гілці `Model – Component 1 – Heat Transfer in Solids – Domain fluxes – Total heat flux`.



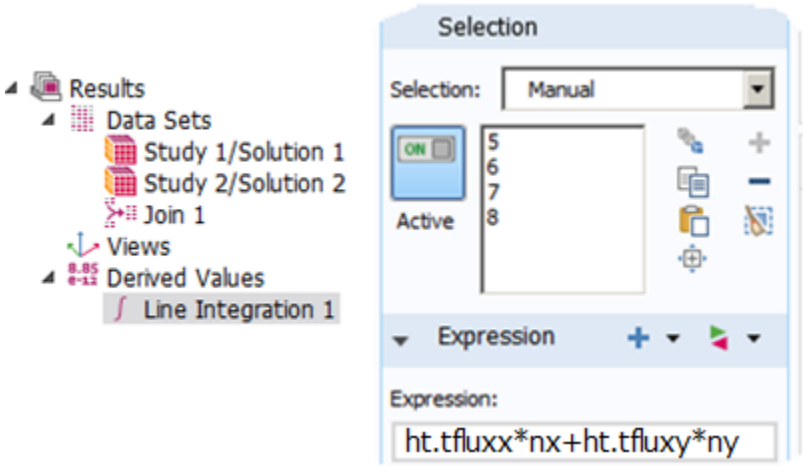
Отже імена `ht.tfluxx` і `ht.tfluxy` представляють координати вектора теплового потоку.

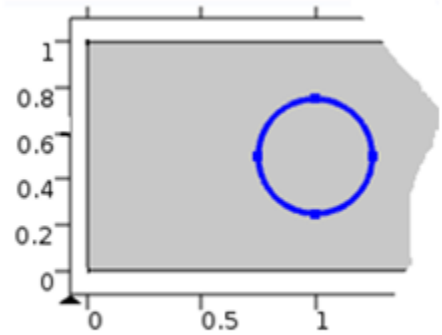
Інший спосіб знайти імена змінних, які визначені в обраному фізичному інтерфейсі, полягає в перегляді таблиці змінних в ПВ вузла **Heat Transfer in Solids 1 - Equation View**. У вузлах типу `Equation View` приводяться таблиці існуючих змінних моделі (наступний рисунок праворуч). Але такі вузли відображаються в ДМ лише тоді, коли в списку **Show**, який розташований вгорі панелі `Model Builder`, встановлено прапорець **Equation View** (наступний рисунок ліворуч).





Обчислимо сумарний тепловий потік через межу джерела тепла (ліве коло), тобто обчислимо інтеграл $\oint_C \boldsymbol{q}\cdot\boldsymbol{n}\,ds$. Для цього клацніть правою кнопкою миші по вузлу **Results - Derived Values** і в контекстному меню виберіть команду Integration - Line Integration. В ПВ створеного вузла **Line Integration 1** в секцію вибору (клацаючи мишею в графічному вікні) помістіть індикатори дуг лівого кола 5, 6, 7, 8. Це означає, що інтегрування буде виконуватися по цим кривим (тобто по всій окружності). Щоб інтеграл обчислювався від величини $\boldsymbol{q}\cdot\boldsymbol{n}$, де $\boldsymbol{q}=(q_x,q_y)$ вектор теплового потоку, а $\boldsymbol{n}=(n_x,n_y)$ одиничний вектор нормалі до кривої, в поле **Expression** ПВ вузла **Line Integration 1** слід ввести вираз $q_x n_x + q_y n_y$. В позначеннях COMSOL Multiphysics він може буде записаний у вигляді `ht.tfluxx*nx + ht.tfluxy*ny`.





Клацніть вгорі панелі **Line Integration 1** по кнопці Evaluate, і в інформаційній зоні вікна програми з'явиться таблиця зі значенням `19583`. Тепер ще раз обчисліть інтеграл, але по лівій і верхній сторонах квадрата. Для цього видаліть з секції вибору номери дуг кола (повторно в графічній області клацніть по ним), і додайте індикатори `1` і `3` відповідних сторін прямокутника. Розкрийте кнопку - список Evaluate і виберіть `Table 1`. В тій же таблиці з'явиться значення інтеграла `19611`. Тепер інтеграл від того ж самого виразу

обчислювався уздовж лівої і верхньої сторін квадрата. Оскільки повинен дотримуватися тепловий баланс, а на інших ділянках границі заданий нульовий тепловий потік, то обидва інтеграли повинні були б збігтися. Однак в силу наближеності обчислень вони дещо відрізняються. Повторіть два останніх обчислення, але в якості джерела даних для `Line Integration` оберіть **Study 2/Solution 2**. Помістіть результати в ту ж саму таблицю.

5,6,7,8 Study 1 (W/m)	1,3 Study 1 (W/m)	5,6,7,8 Study 2 (W/m)	1,3 Study 2 (W/m)
19583	19611	19583	19629

До речі, ми знаємо точну відповідь, бо сумарний тепловий потік через межу джерела тепла повинен дорівнювати сумарній потужності джерела, тобто величині $Q_{\Sigma} = 19634.95[W/m]$, яку ми обчислили раніше.

Загальну потужності джерела тепла ми можемо також обчислити як подвійний інтеграл по області. Дійсно, потужність внутрішніх джерел тепла ми задали при введенні даних у ПВ вузла **Heat Source 1**. Тому інтегрування цієї величини по області лівого кола дасть потрібну сумарну потужність.

Results –> Derived Values:

RMB > Integration - Surface Integration

Results -> Derived Values – Surface Integration 2:

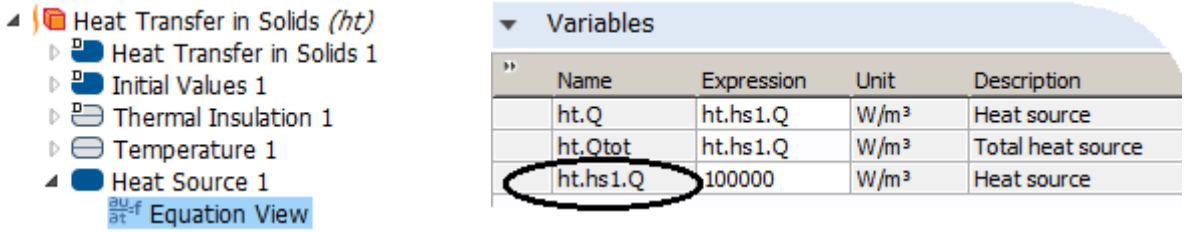
`Data` – **Data set** = `Study 1/Solution 1`

`Selection` – **Selection** = `Manual`

У розділі **`Selection`** натисніть кнопку Active (якщо вона ще не активна) і в графічній області клацніть по лівому колу, додавши індикатор 2 в секцію вибору. Це означає, що інтегрування буде виконуватися по цій області (тобто по всьому колу). Потім введіть:

`Expression` – **Expression** = `ht.hs1.Q`

Ім'я змінної `ht.hs1.Q`, яка представляє потужність джерела тепла, можна знайти у вікні **Equation View** вузла **Heat Source 1**.



Тепер надрукуємо значення інтеграла, виконавши команду

LMB > Evaluate –Table 1

В таблиці **Table 1** з'явиться значення `19635`.

Взагалі, прикладні інтерфейси містять багато додаткових обчислюваних величин. Звісно, що загальна потужність джерел тепла теж обчислюється. Перевіримо це.

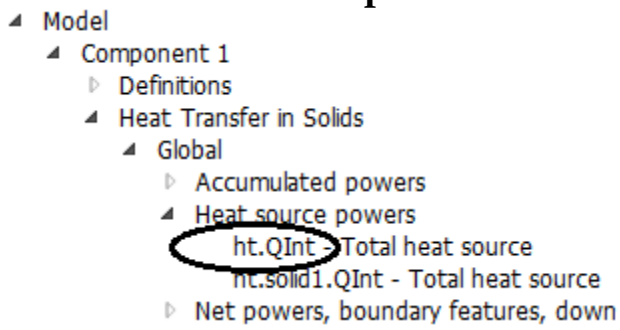
Для того, щоб друкувати значення, які не залежить від просторових координат, в гілці **Derived Values** створюються вузли типу `Global Evaluation`.

Results –> Derived Values: RMB > Global Evaluation

Results -> Derived Values – Global Evaluation 1:

Expression – **Expression** = `ht.QInt`

Ім'я змінної `ht.QInt`, яка представляє загальну потужність джерела тепла, ми знаходимо у вікні готових величин поля **Expression**.



Надрукуємо це значення в першій таблиці.

LMB > Evaluate –Table 1

ht.tfluxx*nx +	(W/m) ht.tfluxx*nx +	ht.tfluxx*nx	ht.tfluxx*nx +	Heat source (W/m)	Total heat source (W)
19583	19611	19583	19629	19635	19635

Єдина відмінність від попередніх значень полягає в одиницях вимірювання, яка для змінної `ht.QInt` дорівнює `W` (вона вже помножена на 1 м умовної товщини конструкції).

Припустимо тепер, що ви бажаєте побудувати графік температури уздовж більшої осі прямокутника. Тоді в модель в гілку **Results** потрібно додати одновимірну графічну групу і лінійний графік. Але спершу потрібно створити джерело даних, яке зберігає інформацію про розв'язок уздовж бажаної лінії. Отже виконайте інструкції

`Results` -> Data Set - Cut Line 2D

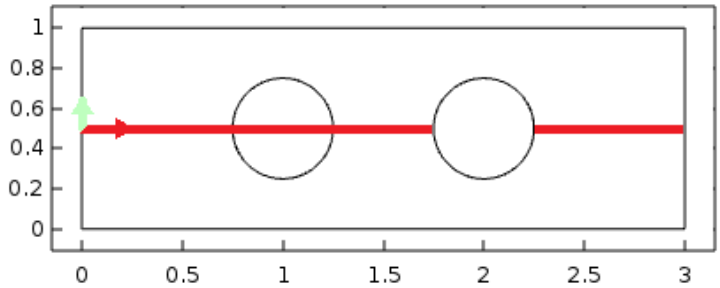
В ПВ створеного вузла **Cut Line 2D 1** введіть:

Data – **Data set** = `Study 1/Solution 1`

Line Data – **Point 1** – **x, y** = `0,0.5` (*два поля*)

Line Data – **Point 2** – **x, y** = `3,0.5` (*два поля*)

Побудуйте лінію командою Plot.



Тепер побудуйте на цій лінії графік. Створіть групу одновимірних графіків командою

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

В ПВ створеного вузла **1D Plot Group 6** введіть:

Data – **Data set** = `Cut Line 2D 1`

Зверніть увагу, що на стрічці панелей з'явилася закладка `1D Plot Group 6`. З неї виберіть команду

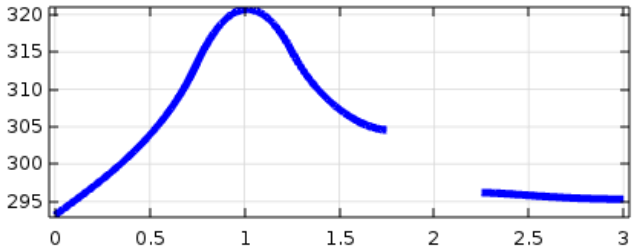
`1D Plot Group 6`-> Line Graph

В ПВ створеного вузла **Line Graph 1** введіть:

y-Axis Data – **Expression** = T

x-Axis Data – **Parameter, Expression** = Expression,x (*два поля*)

Побудуйте графік температури командою Plot.



Додайте сюди графік температури вздовж нижньої сторони прямокутника. Для цього в поточній графічній групі створіть ще один лінійний графік.

1D Plot Group 6-> Line Graph

В ПВ створеного вузла **1D Plot Group 6 - Line Graph 2** введіть:

Data – **Data set** = Study 1/Solution 1

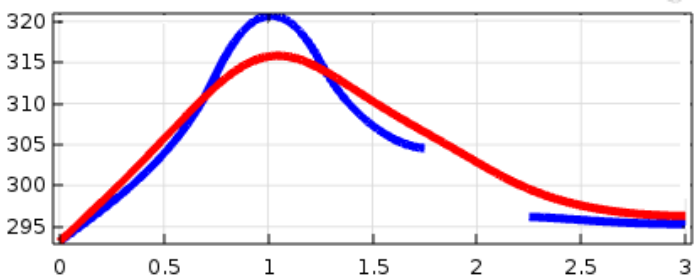
Selection – **Selection** = Manual

У розділі **Selection** натисніть кнопку Active і в графічній області клацніть по нижній стороні прямокутника, додавши індикатор 2 в секцію вибору. Потім виконайте інструкції:

y-Axis Data – **Expression** = T

x-Axis Data – **Parameter, Expression** = Expression,x (*два поля*)

LMB > Plot



Перед побудовою графіка вздовж лінії, якої немає в початковій геометрії, ми будували окреме джерело даних (Cut Line 2D 1). Для цього ми задали його положення в геометрії компонента. Але крім цього таке джерело містить всі дані з розв'язку Study 1/Solution 1, які відносяться до його геометрії. Після створення джерело даних можна використовувати для побудови графіків. Для цього в графічних вузлах в полях **Data - Data set** вказується ім'я цього джерела/набору даних. Для ліній, які вже присутні в геометрії компонента, створювати нове джерело даних немає потреби. Їх індикатори можна відразу розміщати (клацаючи по ним мишею) в секції вибору ПВ графічного вузла.

Якщо вам потрібно виконати інтегрування по лінії, якої немає в геометрії, то таким же чином ви повинні створити набір даних типу Cut Line 2D, а потім використати його в якості джерела даних для інтегрування. Якщо крива інтегрування присутня у вхідній геометрії, то створювати джерело даних не потрібно. Таку криву можна відразу вибирати в якості джерела даних в ПВ вузла інтегрування.

Схожий підхід використовується в 3D випадку. Якщо розв'язок визначений в тривимірній області і треба проінтегрувати його по поверхні, якої

немає в області, то слід створити джерело даних типу Cut Plane 3D і використати його для обчислення поверхневого інтеграла. Те саме джерело можна використовувати для побудови графіка функції на поверхні.

Аналогічно використовуються «точкові» набори даних Cut Point, які крім свого місця розташування містять інформацію про розв'язок задачі в точці (або точках). Значення розрахункових величин в точках, які вже є в геометрії, можна обчислювати відразу, вказавши точку в секції вибору вузла **Point Evaluation**. Якщо потрібних точок в геометрії немає, то слід побудувати джерело даних **Cut Point**, задавши в його ПВ положення точок і вказати розв'язок (якщо в моделі їх декілька) з якого будуть вибиратися дані.

Створіть точковий набір даних, який показаний на наступному рисунку. Для цього виконайте інструкції:

`Results` -> Data Set - Cut Point 2D

Data Sets - Cut Point 2D 1:

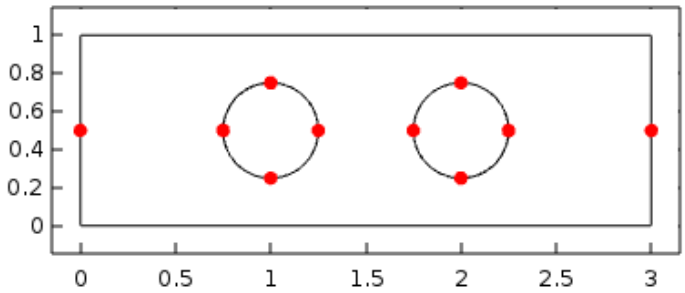
`Data` – **Data set** = `Study 1/Solution 1`

`Point Data` –**Entry method** = `Coordanates`

`Point Data` – **x:** = `0,0.75,1,1,1.25,1.75,2,2,2.25,3`

`Point Data` – **y:** =`0.5,0.5,0.75,0.25,0.5,0.5,0.75,0.25,0.5,0.5`

Тут в полях **Point Data-x** і **Point Data-y** вводяться абсциси і ординати точок набору даних **Cut Point 2D 1**. Побудуйте зображення точок командою Plot.



Тепер в точках, представлених в наборі даних `Cut Point`, можна обчислювати всі величини, які є в розв'язку **Study 1/Solution 1**.

У вузлі **Derived Values** створіть вузол типу `Point Evaluation`, який призначений для обчислення значень в точках. Для цього виконайте команду

Derived Values: RMB > Point Evaluation

Derived Values - Point Evaluation 1:

`Data` – **Data set** = `Cut Point 2D 1`

`Expression` – **Expression** = `T`

LMB > Evaluate – New Table

В гілці **Tables** з'явиться вузол нової таблиці **Table 2**, а сама таблиця буде відкрита в інформаційній зоні під графічним вікном. Вона буде містити значення температури в обраних точках. Якщо відкрити ПВ таблиці **Results – Tables – Table 2**, то в розділі **`Column Headers`** ви побачите таблицю заголовків, яку можна редагувати. Поміняйте заголовки так, як показано на наступному рисунку, залишивши тільки координати точок.

Column Headers

Column	Header
1	(0, 0.5)
2	(0.75, 0.5)
3	(1, 0.75)
4	(1, 0.25)
5	(1.25, 0.5)
6	(1.75, 0.5)
7	(2, 0.75)
8	(2, 0.25)
9	(2.25, 0.5)
10	(3, 0.5)

В результаті таблиця набуде наступного вигляду.

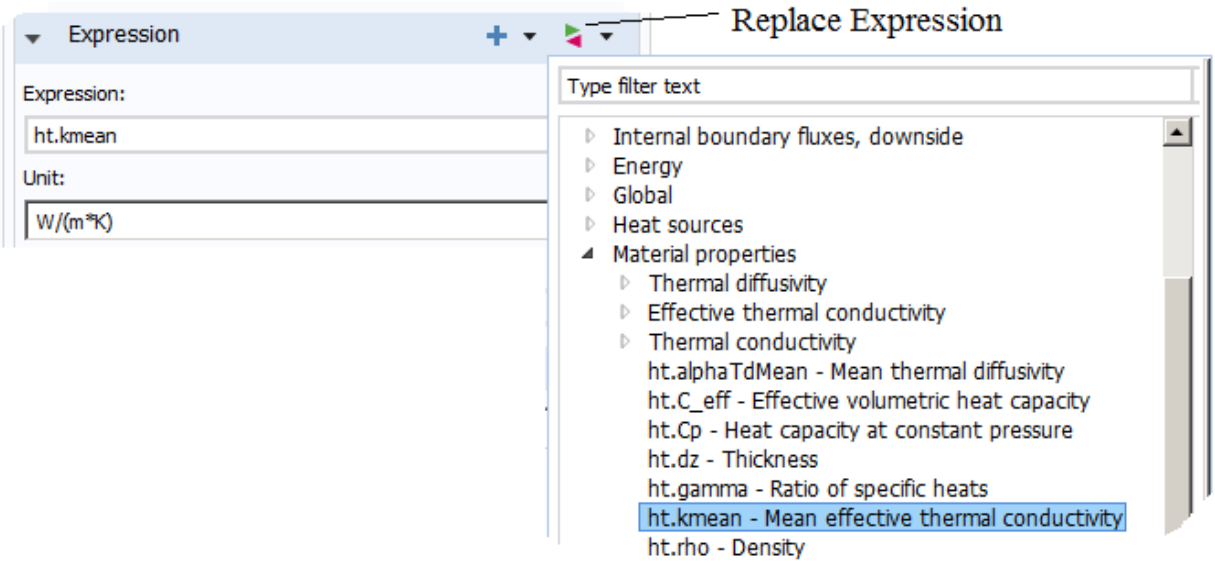
Messages Progress Log Table 2

(0, 0.5)	(0.75, 0.5)	(1, 0.75)	(1, 0.25)	(1.25, 0.5)	(1.75, 0.5)	(2, 0.75)	(2, 0.25)	(2.25, 0.5)	(3, 0.5)
293.15	313.48	308.84	319.13	315.06	304.57	294.63	302.78	296.16	295.29

Вузли `Point Evaluation` можна використовувати не тільки для друку значень розв'язку, але і для визначення значень фізичних параметрів. Для цього в геометрії моделі можна вибрати точку і вивести в ній значення будь-якого параметра. В нашому прикладі ви можете виконати наступну інструкцію

Results - Derived Values: RMB > Point Evaluation

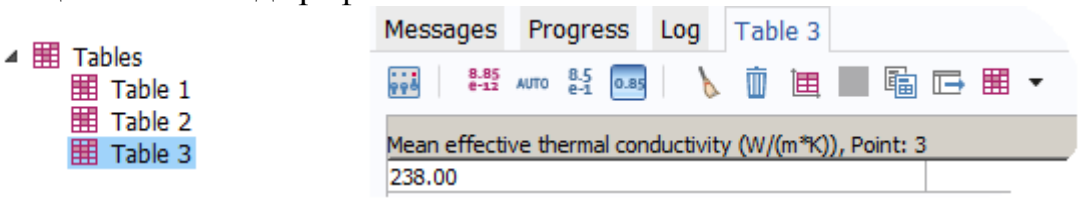
В ПВ створеного вузла **Point Evaluation 2** в розділі **`Selection`** виберіть точку області і клацанням миші по ній додайте її номер (наприклад, 3 - крайня ліва точка лівого кола) в секцію вибору. Оскільки точка вже існує в геометрії, то створювати додаткове джерело даних не потрібно. Справа над текстовим полем **Expressions** розкрийте кнопку Replace Expression. У відкрившомуся вікні, розкрийте дерево змінних готових до вставки, знайдіть і розкрийте гілку `Material properties`, і подвійним клацанням виберіть бажаний параметр матеріалу (наприклад, середнє значення коефіцієнта теплопровідності).



В полі **Expression** з'явиться вираз `ht.kmean`, значення якого можна відобразити в таблиці, якщо виконати команду

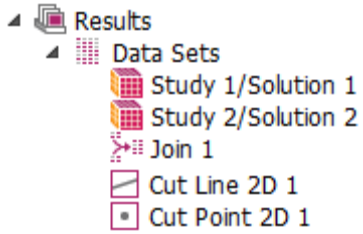
LMB > Evaluate – New Table

В гілці **Tables** з'явиться вузол нової таблиці `Table 3`, а сама вона буде відкрита в інформаційній зоні під графічним вікном.



Для ізотропного матеріалу коефіцієнт теплопровідності `k` зберігається у вигляді діагонального тензора з однаковими компонентами `kxx`, `kyy` і `kzz`, до яких у виразах можна звертатися за іменами `ht.kxx`, `ht.kyy` і т.д. Тут `ht` ім'я вузла **Heat Transfer in Solids (ht)**. У будь-якого вузла є назва (Label) і ім'я (Name). Тут `Heat Transfer in Solids` - назва (позначка) вузла, а `ht` - ім'я. Система автоматично надає імена деяким вузлам. В панелях властивостей таких вузлів в текстовому полі **Name** ці імена можна міняти. В інтерфейсі `Heat Transfer in Solids` скалярний коефіцієнт теплопровідності `k` обчислюється як середнє значення елементів `kxx`, `kyy` і `kzz`, і позначається через `ht.kmean`. Саме його значення відображено в таблиці на попередньому рисунку.

Як ви бачили, при обробці результатів обчислень важливу роль відіграють набори даних, що створюються в гілці **Data Sets**. Після обчислення моделі тут завжди присутній набір, який являє «повний» розв'язок (`Study 1/Solution 1`). Вузли типу `Study X/Solution X`, де `X` - цифра, як правило створюються солверами (розв'язувачами). Вони містять інформацію про компонент і його геометрію, про шукані величини, а також значення змінних, створених у вузлі **Variables** компонента.



Допоміжні набори/джерела даних створюються користувачем безпосередньо, або неявно через інші елементи ДМ. Наприклад, зонди (probes), якщо ви побудуєте їх в гілці **Definitions** компонента, самі створюють джерела даних.

Крім гілки **Data Sets** з наборами даних, в гілці **Results** є вузол **Derived Values** (залежні величини). Він призначений для створення інструкцій обчислення залежних від розв'язку величин: мінімальних і максимальних значень, інтегралів, середніх значень, а також для обчислення окремих величин в точках.

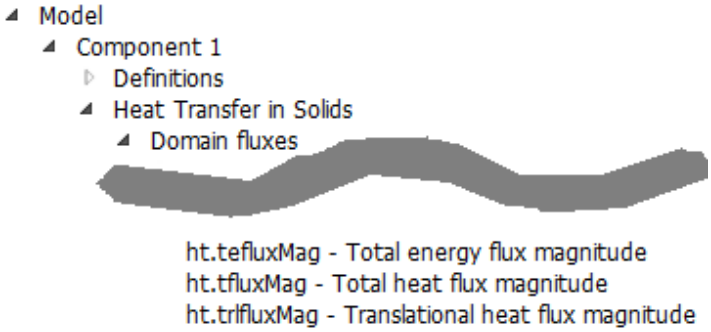
Знайдемо максимальну величину теплового потоку. Спочатку виконайте команду

Results - Derived Values:

RMB > Maximum – Surface Maximum

В ПВ створеного вузла **Surface Maximum 1** в якості набору даних за замовчуванням використовується розв'язок **Study 1/Solution 1**. В секції вибору слід вказати індикатори 1 і 2 (вся геометрія), які визначають область де буде

обчислюватися максимум. Вираз, максимум якого буде обчислюватися, вказується в полі **Expression**. Справа над текстовим полем **Expressions** розкрийте кнопку Replace Expression. У відкрившомуся вікні розкрийте дерево готових до вставки змінних, знайдіть і розкрийте гілку `Domain fluxes`, і подвійним клацанням виберіть бажану змінну `ht.tfluxMag`.



В полі **Expression** з'явиться вираз `ht.tfluxMag`, значення якого відобразиться в таблиці, якщо клікнути по кнопці Evaluate, розташованій вгорі ПВ вузла **Surface Maximum 1**.

Total heat flux magnitude (W/m^2)
18218

Тепер в ПВ вузла **Surface Maximum 1** поміняйте джерело даних, вибравши другий розв'язок **Study 2/Solution 2**. Знову знайдіть максимум. Для цього виконайте інструкції

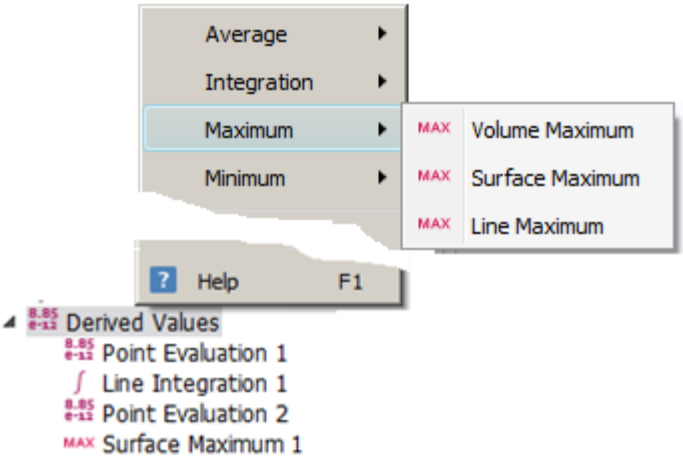
`Data` – **Data set** = `Study 2/Solution 2`

LMB > Evaluate – Table 4

У попередню таблицю додасться друге значення максимуму теплового потоку, але обчислене за розв'язком, побудованим на дрібнішій сітці.

Total heat flux magnitude (W/m^2)	Total heat flux magnitude (W/m^2)
18218	18056

Максимум і мінімум можна обчислювати на лініях, поверхнях і за об'ємом. Аналогічно обчислюються середні значення і інтеграли (по лініях, поверхнях і об'ємах). Це видно з контекстного меню, якщо натиснути правою кнопкою миші по вузлу **Derived Values** і розкрити підменю.



В гілці **Derived Values** знаходяться також вузли **Point Evaluation**, які використовуються для відображення значень змінних або виразів в конкретній

точці (на відміну від максимуму, який обчислювався по області). Можна виводити значення в декількох точках одночасно і поміщати результати в одну таблицю. Вузли типу `Point Evaluation` ми створювали раніше.

Збережіть файл моделі, давши йому якесь осмислене ім'я, наприклад, Heat2D_InHoledRect.mph.

■

Файли моделей зберігаються у `*.mph` файлах командами File – Save або File – Save As.

Під час сеансу система зберігає історію роботи з моделлю і веде повний облік змін, які в ній чиняться. Це збільшує розмір файла. Щоб його стиснути існує команда File - Compact History (ущільнення історії), яка видаляє всі перевизначені зміни і скорочує розмір моделі. Перед остаточним збереженням моделі в файл рекомендується виконати цю команду.

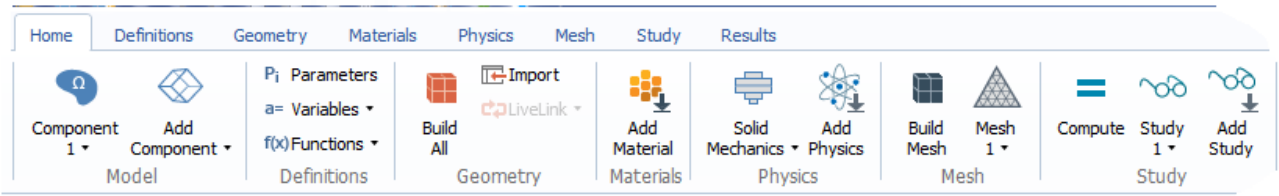
Підіб'ємо невеликий підсумок. Основним інструментом розв’язання задач в COMSOL Multiphysics є конструктор моделей (`Model Wizard`). Базовими об'єктами, які використовуються в конструкторі моделей, виступають вузли:

- **Global Definition** – призначений для визначення глобальних параметрів, змінних і функцій, які можна використовувати в будь-якому місці моделі;
- **Component** – призначений для створення основної моделі, включаючи геометрію конструкції, завдання матеріалів і розбиття на скінченні елементи; в гілці **Geometry** задається геометрична форма об'єкта, яка являє розрахункову область при чисельному моделюванні;
- **Study** – призначений для обчислювання задачі, вибору і налаштуванню параметрів розв’язувача (програми, яка реалізує чисельний алгоритм);
- **Results** – призначений для зображення та аналізу результатів, зокрема для побудови графіків розв’язків.

Фактично, вузли є інструкціями створення моделі. Послідовність вузлів утворює дерево моделі, а управління моделлю трансформується в роботу з деревом моделі. В наступному параграфі ми докладніше розглянемо деякі інструменти керування вузлами цього дерева.

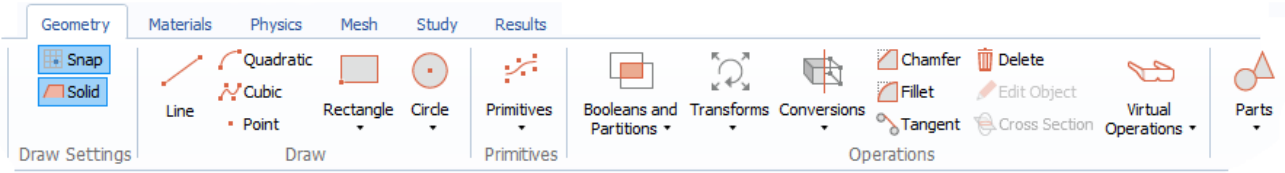
1.5.2. Керування деревом моделі

Головним інструментом управління деревом моделі COMSOL Multiphysics, починаючи з п'ятої версії, є стрічковий інтерфейс. Вважається, що стрічки дають більш наочний і простий спосіб управління програмами в порівнянні з вкладеними випадними меню. Філософія стрічки полягає в групуванні команд і відображенні тоді, коли їх можна використовувати. Команди групуються по вкладках, які в COMSOL Multiphysics впорядковані зліва направо відповідно до головної послідовності інструкцій створення моделей.



Інструментальна панель Home містить кнопки для загальних команд, таких як внесення змін до дерева моделі або виконання обчислення. Наприклад, якщо ви захочете змінити параметр, який управляє геометричним розміром, додати нову фізику, побудувати СкЕ сітку, виконати обчислення, додати новий графік - ви робите це з вкладки Home. Решта панелей інструментальної стрічки впорядковані зліва направо згідно потоку операцій, який заданий послідовністю розташування гілок в ДМ: Визначення, Геометрія, Матеріали, Фізика, Сітка, Дослідження та Результати.

Стрічка панелей реалізує три типи групування команд. Перший – стандартні закладки, які завжди відображаються в верхній частині програми. Вони відносяться до всіх основних етапів процесу моделювання і впорядковані відповідно до його основних етапів. На кожній такій закладці можна побачити всі команди, доступні для цього етапу. Наприклад, закладка Geometry містить значки команд, які можна використовувати при створенні геометрії моделі.

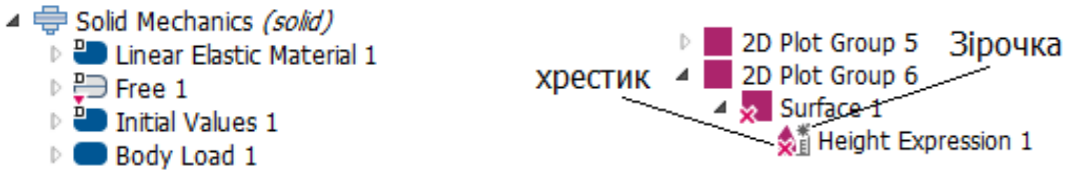


Другий тип - це контекстні закладки, які з'являються, коли ви працюєте зі специфічним об'єктом. Прикладом цього є закладки графічних груп, які з'являються на час роботи з графіками.

Третім типом логічного групування є модальні закладки, які з'являються, коли ви виконуєте дуже специфічні завдання, яких немає на стандартних закладках стрічки. Прикладом модальної закладки є Work Plane. Коли ви працюєте з «робочими перерізами» інші закладки недоступні, якщо їх команди не можуть керувати відповідними площинами. Наприклад, ви не зможете створити СкЕ сітку на робочій площині і тому закладка Mesh недосяжна.

Відмітною рисою кожного кроку роботи з моделлю є кольорова схема. Наприклад, всі значки панелі Geometry є червоно-помаранчевими, всі піктограми панелі Materials – жовто-оранжеві, всі значки панелі Physics – сині, а піктограми панелі Results – червоні. До того ж закладка Results має різні значки для різних розмірностей.

На піктограмах вузлів ДМ ви іноді будете бачити маленькі позначки. Маленький червоний трикутник на піктограмі вузла попереджає, що він (вузол) був переписаний; вузли іноді змінюють (переписують) вміст вікон властивостей інших вузлів, які в ДМ розташовані вище.

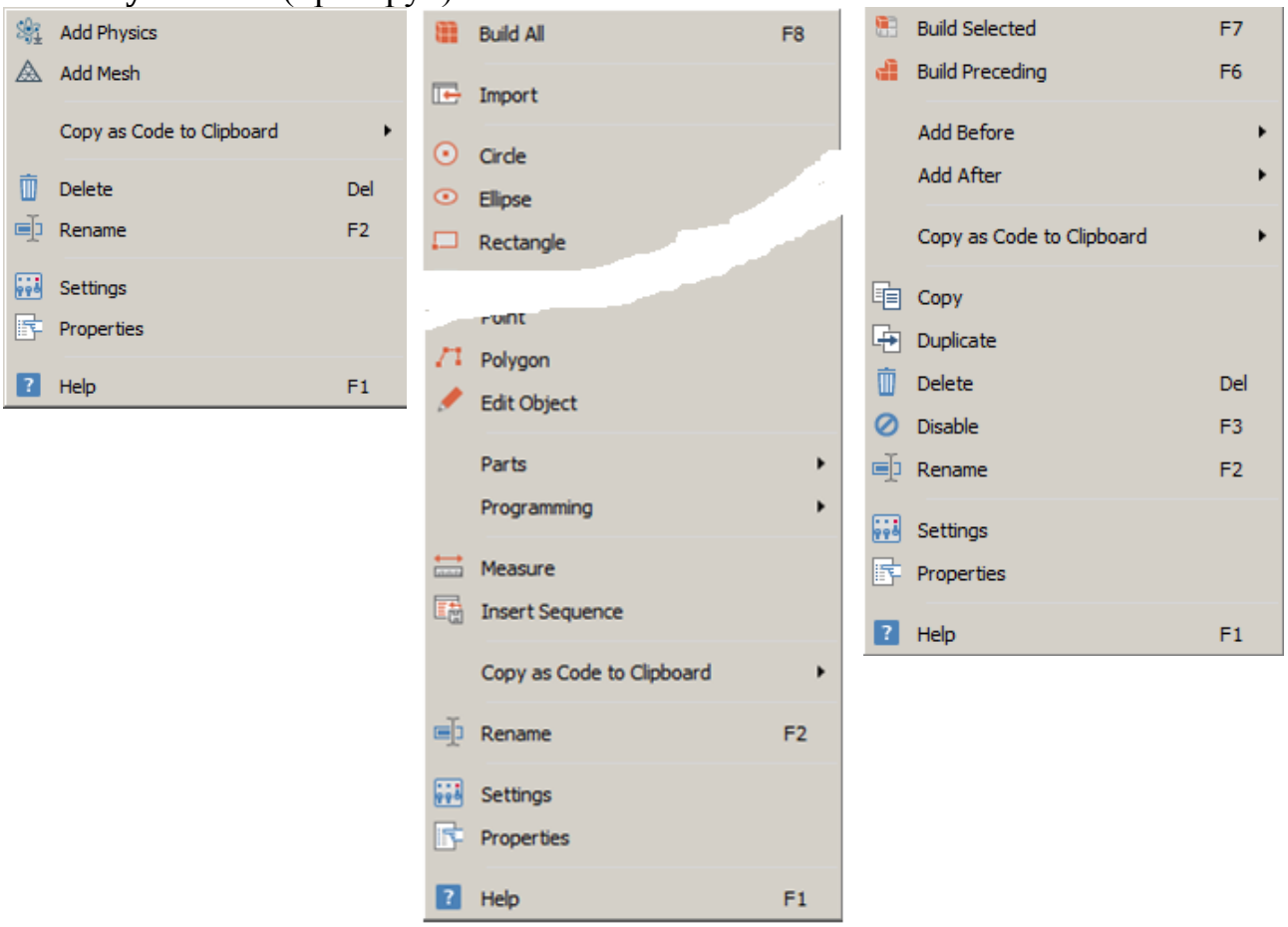


Буква D вгорі значка вузла означає, що вузол був створений системою автоматично з параметрами за замовчуванням. Зірочка в верхньому правому

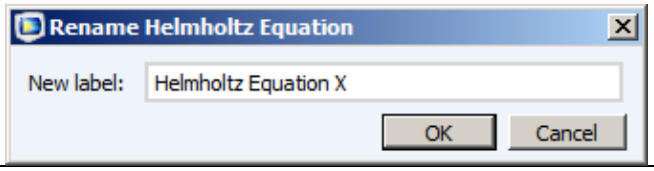
куті значка вказує, що вузол редагується. Червоний хрестик на піктограмі вузла означає, що є помилка в ПВ цього вузла.

При моделюванні відмінність між деревом моделі і стрічкою панелей полягає в тому, що ДМ показує нам те, що вже зроблено, а стрічка підказує, що може йти далі.

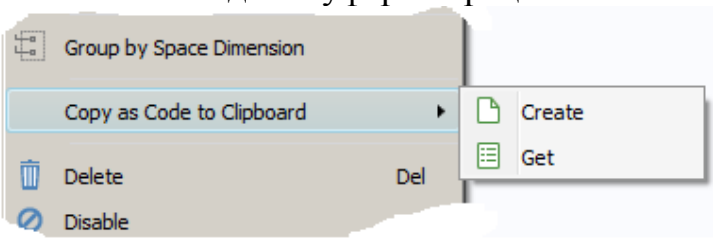
Майже кожну команду можна виконати двома способами (або навіть більшим): вибором іконки команди на стрічці або клацанням правої кнопки миші по вузлу ДМ і вибором команди з контекстного меню. Ці методи еквівалентні і доповнюють один одного. Розглянемо, як можна конструювати модель і управляти його деревом, використовуючи контекстне меню. Воно з'являється при натисканні правою кнопкою миші по вузлу дерева. Його зміст залежить від вузла, для якого воно викликано. На наступному малюнку показані контекстні меню гілки двовимірного компонента **Component** (ліворуч), його підгілки **Geometry** (в середині) і вузла графічного примітиву **Geometry - Circle** (праворуч).



Ці меню сильно відрізняються і докладний опис команд, які можуть бути викликані з різних меню, фактично становить зміст нашого посібника. Однак є пункти, які зустрічаються майже всюди. Про них ми поговоримо в цьому параграфі.

Команда	Опис
Help	Відкриває контекстну довідку по поточному вузлу.
Properties	Відкриває службову закладку `Properties` вікна властивостей активного вузла.
Settings	Відкриває вікно властивостей вибраного вузла, тобто ту ПВ, в якій ми зазвичай міняємо налаштування і вводимо дані.
Rename	Відкриває вікно перейменування поточного вузла дерева моделі. 
Copy	Копіює вузол в буфер обміну. Після цього командою `Past назва_узла` можна вставити копію вузла в дерево моделі.
Duplicate (дублювання)	Вставляє в дерево моделі копію активного вузла. Дублікат вузла створюється нижче поточного в тій же гілці. Дублювання допустимо для вузлів багатьох типів, але не для всіх. Воно є зручним інструментом, оскільки виконує дії копіювання і вставки за один крок.
Delete	Видаляє поточний вузол з дерева моделі.
Disable	Деактивує поточний вузол, виключаючи його з послідовності інструкцій роботи з моделлю. Піктограма неактивного вузла в ДМ стає тьмяною.
Enable	Включає поточний вузол в послідовність інструкцій, якщо до цього він був деактивований. Тьмяна піктограма неактивного вузла повертається в початковий стан яскравості.
Move Up Move Down	Переміщують поточний вузол в ДМ на одну позицію вгору або вниз.

Команда Copy as code to clipboard використовується при програмуванні додатків. COMSOL Multiphysics дозволяє створювати застосування, що використовують його об'єктну модель. Команди підменю Copy as code to clipboard записують текст команди в буфер операційної системи.



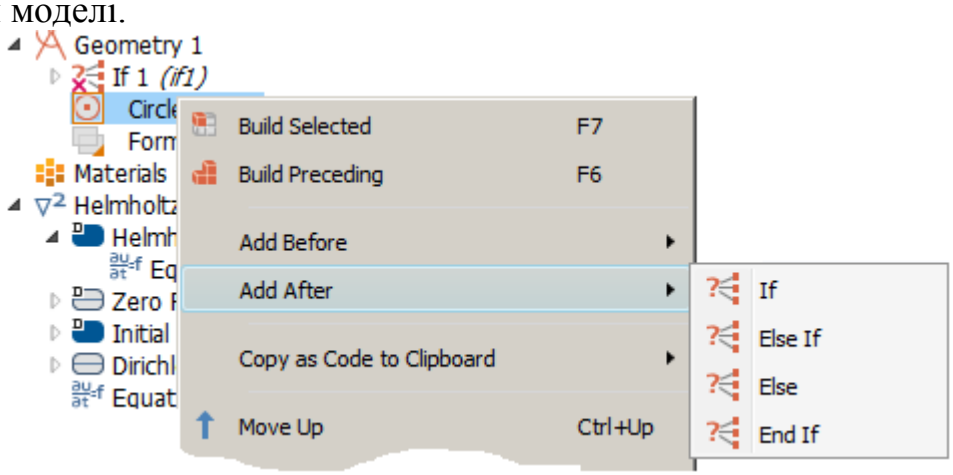
З буфера ви можете вставити текст коду в програму. Ось приклад коду, створеного командою Copy as code to clipboard - Create для вузла **Helmholtz Equation (hzeq)**, який вставлено з буфера в текстовий документ

```
model.physics().create("hzeq","HelmholtzEquation", "geom1");
```

Оскільки ми не маємо наміру описувати процес створення додатків, що використовують COMSOL API, то нам ця можливість не буде потрібна.

Команди Export та Import дозволяють експортувати/імпортувати дані в файли. Експортувати можна геометрію, анімацію, графіки, СкЕ сітку, таблиці, рівняння і деякі інші об'єкти. Імпортування виконується найчастіше з текстових файлів, які раніше створювалися для запам'ятовування даних. Все що експортувалося може бути імпортоване. Імпортувати можна також геометрію, створену в програмах сторонніх розробників, або створену в іншому файлі COMSOL Multiphysics.

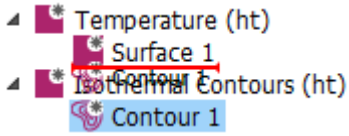
У вузлів, що представляють геометричні елементи, є підміню Add Before і Add After. Вони містять команди створення «умовних» послідовностей графічних вузлів/інструкцій. Це елементи, які включаються в послідовність геометричних вузлів, тільки при виконанні деяких умов, що накладаються на параметри моделі.



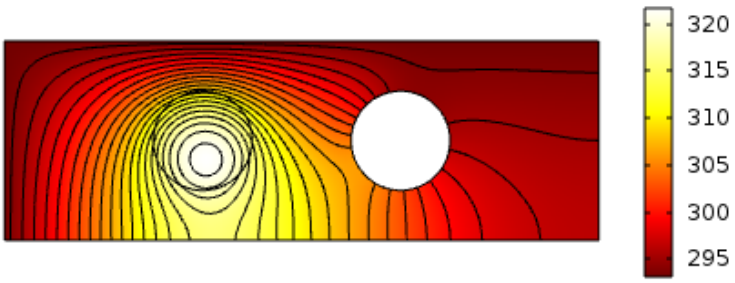
У розділі, що розповідає про створення геометрії, ми докладніше розглянемо приклади застосування таких умовних вузлів/інструкцій.

Всі операції в дереві моделі виконуються по черзі – зверху вниз. Фактично, вузли ДМ є інструкціями створення моделі, які повинні іти в певному логічному порядку. Таким чином, розташування вузлів в ДМ визначає порядок побудови і обчислення моделі. Але позиція вузлів в дереві моделі не фіксована. Багато з них можна переміщати. Однак таке пересування не повинно порушувати «розумну» послідовність. Наприклад, не можна поставити вузол граничних умов вище вузла «фізики», а вузол **Mesh** раніше вузла **Geometry**. Але місцезнаходження деяких вузлів не настільки важливо. Наприклад, не має значення порядок вузлів в гілці **Global Definitions**.

Графічні вузли можна переміщати в межах групи графіків або навіть переміщати в іншу графічну групу. Щоб, наприклад, перемістити графічний вузол, захопіть його лівою кнопкою миші (натисніть кнопку і не відпускайте) і перетягніть вниз або вгору в межах поточної графічної гілки або в іншу графічну групу того ж типу (наприклад, з **2D Plot Group** в таку ж саму). В процесі пересування позиція вузла в дереві позначається відрізком.



В прикладі попереднього параграфа (файл Heat2D_InHoledRect.mph) були створені вузли **Temperature (ht) - Surface 1** з функціональним забарвленням розрахункової області, і вузол **Isothermal Contours (ht) - Contour 1** з контурними лініями того ж температурного поля. Перенесемо вузол **Contour 1** в графічну групу **Temperature (ht)**. Для цього захопіть його лівою кнопкою миші і перетягніть вгору в графічну групу **Temperature (ht)** (обидві групи відносяться до типу 2D Plot Group). В переміщеному вузлі **Contour 1** внесіть зміни в колір контурних ліній (зробіть їх чорними) і відключіть виведення шкали кольорів (легенди). В результаті рисунок в графічній гілці **Temperature (ht)** набуде вигляду



Порожню графічну гілку **Isothermal Contours (ht)** можна видалити. Для цього клацніть правою кнопкою миші по цьому вузлу і в випадному меню виберіть команду Delete.

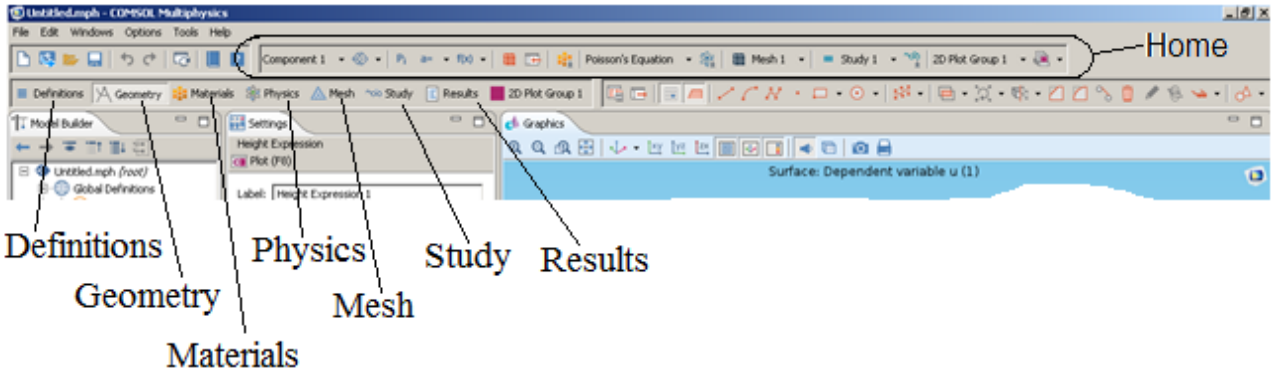
Для керування деревом моделі і швидкого виконання деяких команд можна використовувати гарячі клавіші, а також поєднання клавіш клавіатури і миші. Наведемо деякі з них.

- `F1` – виклик контекстної довідки;
- `Ctrl-F1` – виклик довідкової документації;
- `F3` – відключає/деактивує виділений вузол дерева моделі;
- `F4` – включає вузол, який до цього був деактивований;
- `F7` – виконує завершальну команду поточного редагування вузла: один крок обчислювання (Compute Selected Step), будує елемент геометрії (Build Selected), будує ділянку СкЕ сітки (Build Selected);
- `F8` – повністю перебудовує геометрію (Build All), будує СкЕ сітку (Build All), виконує повне обчислення моделі (Compute), будує графік (Plot);
- `Shift+F10` – викликає контекстне меню;
- `Del` – видаляє поточний вузол ДМ;
- `CTRL - A` – виділяє всі об'єкти в графічному вікні;
- коли активна стрічка інструментальних панелей, колесо миші переміщує фокус між закладками стрічки.

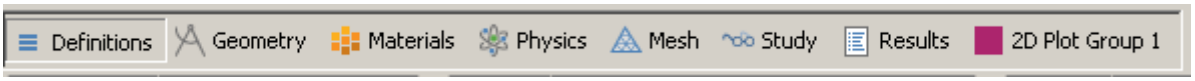
Зауваження про панелі інструментів Windows XP. Якщо вашою операційною системою є Windows 7 або новіша, то подальший текст цього параграфа ви можете пропустити.

У Windows XP відсутня стрічка інструментальних панелей. Тому зовнішній вигляд головного вікна COMSOL Multiphysics відрізняється від стандартного, який ми спостерігаємо в Windows 7 або Windows 10. Замість

головної інструментальної панелі з'являється рядок меню і панелі інструментів, звичайні для додатків Windows XP. Дерево моделі, вікна властивостей, вікно повідомлень і графічне вікно практично не відрізняються. На наступному рисунку показана верхня частина головного вікна COMSOL Multiphysics, запущеного під управлінням Windows XP.



В правій частині першого рядка панелі інструментів розташована панель `Home`. Другий рядок панелі інструментів містить закладки, ідентичні іншим основним закладкам головної інструментальної панелі. При виділенні ярлика закладки, в цьому ж рядку праворуч відкривається відповідна панель інструментів. На наступному малюнку показана ліва половина панелі інструментів з виділеної закладкою `Definitions`.

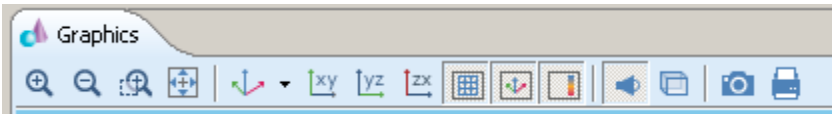


Права половина рядка містить кнопки команд, що відповідають цій закладці.

Закладки `Geometry`, `Materials`, `Physics`, `Mesh`, `Study` і `Results` також відображають панелі зі своїми кнопками команд.

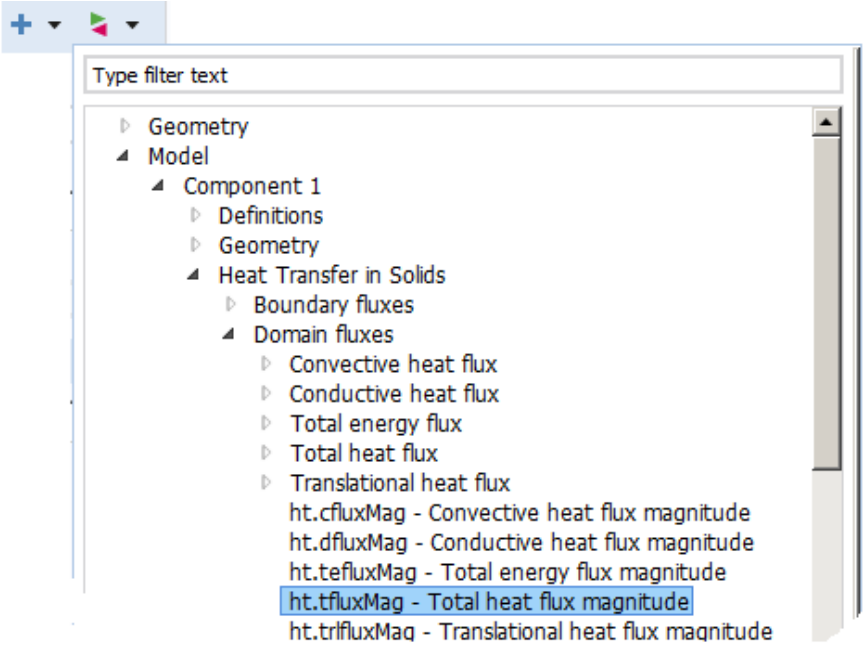
В Windows 7 (10) на стрічці можуть з'являтися контекстні закладки, що містять команди, залежні від виділеного в дереві моделі об'єкта. Аналогічно, в Windows XP можуть з'являтися контекстні панелі.

Панель команд, призначена для управління графічним вікном і яка розташована над ним, практично не відрізняється від аналогічної в Windows 7(10).

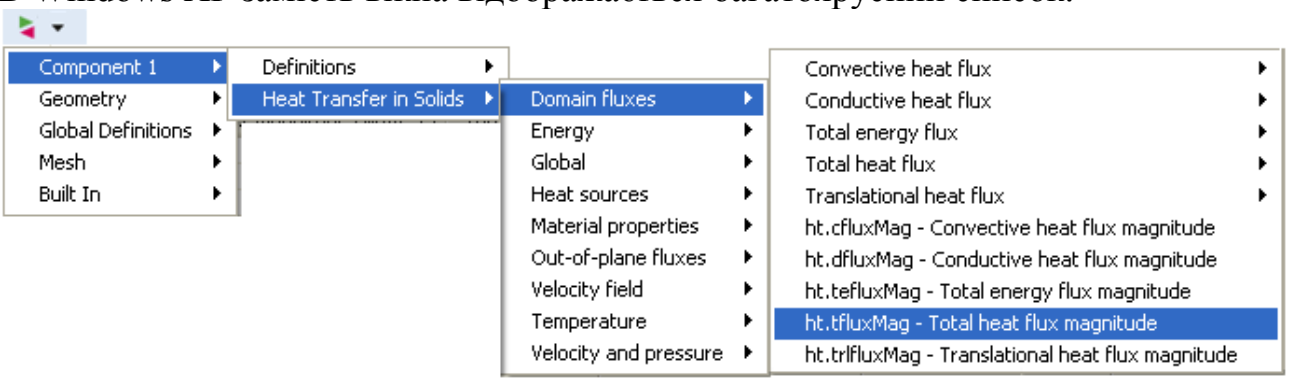


Невелика відмінність спостерігається при відображенні вікна готових величин для вставки в поля **Expression** (наступний рисунок).

Windows 7



В Windows XP замість вікна відображається багатоярусний список.



Зовнішній вигляд інших вікон мало відрізняється від стандартного.

Контекстні меню, які відкриваються клацанням правої кнопкою миші по вузлу дерева моделі, нічим не відрізняються від тих, які відкриваються в Windows 7.

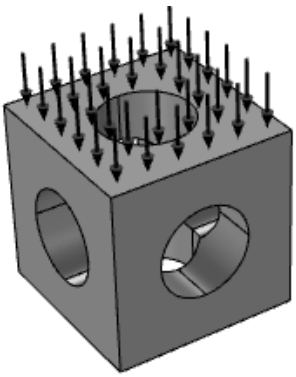
В нашому посібнику ми завжди будемо приводити зовнішній вигляд панелей і вікон системи COMSOL Multiphysics для Windows 7(10) . Якщо ви працюєте в Windows XP, то вікна будуть мати ту ж саму функціональність, але можуть мати трохи інший зовнішній вигляд.

1.5.3. Налаштування графіки.

Для керування зображеннями над кожним графічним вікном розташована панель управління. Її іконки дозволяють збільшувати або зменшувати зображення, зберігати його в файл чи буфер обміну (Image Snapshot), або друкувати.

Для більш ефективного управління графікою існують вузли, налаштування в яких впливають на те, що ми бачимо в графічних вікнах. Призначення таких вузлів та способи їх використання ми пояснимо, розглядаючи задачу деформування тривимірного тіла, яке показано на наступному рисунку. Воно становить бетонний куб $-1 \le x, y, z \le 1$ з циліндричними отворами радіусів

0.5 м, розташованими вздовж координатних осей. Основа куба закріплена. Зверху діє сила $F = 10000\, н/м^2$, спрямована вертикально вниз і рівномірно розподілена по поверхні грані.



Для розв'язання задач теорії пружності в COMSOL Multiphysics призначений прикладний інтерфейс Solid Mechanics. Використовуючи його, вам непотрібно записувати систему ДРЧП (програма зробить це сама). Треба лише побудувати геометрію тіла, вибрати матеріал, задати зовнішню навантагу та вказати обмеження/закріплення. В результаті система обчислює переміщення, напруження і деформації, а також багато інших параметрів.

Створіть модель послідовністю команд:

File->New->Model Wizard->3D->

Structural Mechanics - Solid Mechanics (solid)-> Stationary

Зверніть увагу, що шуканими є функції з іменами `u,v,w`, які були зазначені у вікні вибору фізичної моделі `Select Physics`.

В Comsol Multiphysics в виразах замість числових значень можна застосовувати параметри. Імена та значення параметрів задаються в таблиці параметрів ПВ вузла **Parameters**. В нашому прикладі ми створимо параметри, які представляють довжину сторони куба (a), радіус циліндричного отвору (r), величину зовнішнього тиску на верхню грань (F) і площу грані ($S = a^2 - \pi r^2$).

Задання параметрів моделі. Виконайте команду

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Parameters 1** в колонці **Name** вводьте імена параметрів, а в колонці **Expression** – значення. Поля в стовпці **Value** будуть заповнюватися автоматично. В колонці **Description** вводяться коментарі. Заповніть таблицю параметрів так, як показано нижче.

Parameters

Name	Expression	Value	Description
a	2[m]	2 m	Сторона
r	0.5[m]	0.5 m	Радіус
F	10000[N/m^2]	10000 N/m²	Тиск
S	a^2-pi*r^2	3.2146 m²	Площа грані

Конструювання геометрії. При активному вузлі **Geometry 1** виконайте інструкції:

`Geometry` **->** **`Pimitives`** **->** Block

Geometry 1 – Block 1 (blk1):

`Size` – **Width, Depth, Height** = `a,a,a` (*три поля*)

Position – **Base** = Center

LMB > Build Selected

Створіть три циліндри наступними інструкціями.

Geometry -> **Primitives** -> Cylinder

Geometry 1 – Cylinder 1(cyl1):

Size and Shape – **Radius** = r

– **Height** = a

Position – **x, y, z** = 0,0,-a/2 (*три поля*)

Axis – **Axis type** = z-axis

Geometry -> **Primitives** -> Cylinder

Geometry 1 – Cylinder 2(cyl2):

Size and Shape – **Radius** = r

– **Height** = a

Position – **x, y, z** = 0,-a/2,0 (*три поля*)

Axis – **Axis type** = Cartesian

– **x, y, z** = 0,1,0 (*три поля*)

Geometry -> **Primitives** -> Cylinder

Geometry 1 – Cylinder 3(cyl3):

Size and Shape – **Radius** = r

– **Height** = a

Position – **x, y, z** = -a/2,0,0 (*три поля*)

Axis – **Axis type** = Cartesian

Axis – **x, y, z** = 1,0,0 (*три поля*)

LMB > Build All Objects

За допомогою операції логічного віднімання видалимо циліндри з області куба.

Geometry -> **Booleans and Partitions** -> Difference

Geometry 1 – Difference 1 (dif1):

Difference – **Object to add** = blk1 (*куб*)

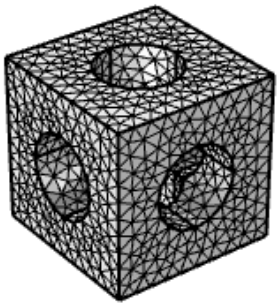
– **Object to subtract** = cyl1,cyl2,cyl3 (*циліндри*)

LMB > Build All Objects

В результаті буде побудований куб з циліндричними отворами, який показано на початку цього параграфа (без стрілок навантаги).

Побудуйте скінченно - елементну сітку з налаштуваннями за замовчуванням.

Component 1 – Meah 1: LMB > Build All

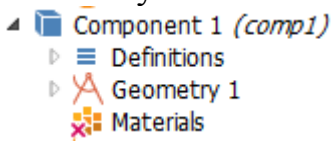


Вибір матеріалу. Виконайте команду

Materials -> Add Material

Праворуч від графічного вікна відкриється вікно Add Material. У вікні Add Material в дереві доступних матеріалів виберіть Built-In – Concrete (бетон). 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)** і його ПВ з таблицею фізичних параметрів. Галочки навпроти деяких властивостей вказують, що їх значення використовуються в поточній фізичній моделі.

Зауваження. Якщо в ПВ вузла «фізики» вказати вибір матеріальних параметрів From material, і спробувати обчислити модель без зазначення матеріалу, то в лівому нижньому куті піктограми вузла **Materials** з'явиться червоний хрестик (×), який позначатиме помилку.



Задайте навантаги і граничні умови, виконавши наступні інструкції.

Physics -> **Boundaries** – Boundary Load

Boundary Load 1:

Boundary Selection – **Selection** = 4 (*верхня грань куба*)

Force – $\mathbf{F_A}$ = 0,0,-F (*три поля*)

Оскільки зовнішній тиск F на верхню грань спрямований вниз (проти напряму вісі Z), то потрібно поставити знак мінус.

Physics -> **Boundaries** – Fixed Constraint

Fixed Constraint 1:

Boundary Selection – **Selection** = 3 (*нижня грань куба закріплена*)

Крім граничних умов треба врахувати власну вагу тіла. Це об'ємна сила. Тому вона відноситься до групи навантаг, які задаються в випадному списку **Domains** панелі Physics.

Physics -> **Domains** – Body Load

Body Load 1:

Domain Selection – **Selection** = 1 (*куб*)

Force – $\mathbf{F_V}$ = 0,0,- solid.rho*g_const (*три поля*)

Тут $\mathbf{F_V}$ представляє вектор ваги одиниці об'єму. Оскільки цей вектор спрямований вниз, то ми поставили знак мінус. Зверніть увагу, як було використано ім'я густини бетону solid.rho – за допомогою точкової нотації. Стала g_const позначає прискорення вільного падіння.

Виконайте обчислення командою

Home->Compute

В гілці **Results** з'явиться гілка **Stress (solid)** тривимірної групи графіків з вузлом **Surface 1**, що містить поверхневий графіки напружень Мізеса.

Поверхневі графіки призначені для кольорового відображення значень величин на поверхнях тривимірних тіл.

Змінимо параметр, який відображається на поверхневому графіку.

Stress (solid) - Surface 1:

`Expression` – **Expression** = `solid.sz`

`Coloring and Style` – **Color table** = `RainbowLight`

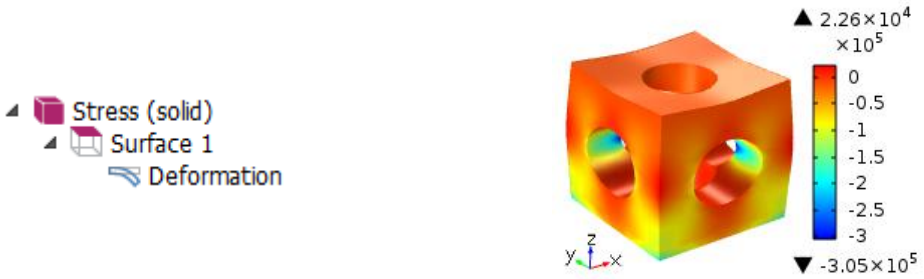
Тут `solid` являє ім'я «фізики» (ім'я, яке вказано в полі **Name** ПВ вузла **Solid Mechanics (solid)**), а змінна `sz` позначає σ_z напруження (внутрішню вертикальну силу, що припадає на одиницю площі перерізу нормального до вісі Z).

Для наочного зображення деформації система створила підвузол **Deformation**. У ньому в текстових полях розділу **`Expressin`** вказані імена координат вектора переміщення (`u,v,w`). Щоб краще побачити деформацію тіла ми лише змінимо масштабний коефіцієнт **Scale factor**. На зображенні це змінить зміщення кожної точки у зазначену кількість разів.

Stress (solid) - Surface 1 - Deformation:

`Scale` – **Scale factor** = `True, 25000`

LMB > Plot



Справа від тіла зображено шкалу, яка представляє відповідність кольорів значенням напруження σ_z. Щоб на графіку не відображався контур тіла, а на шкалі кольорів відобразилося мінімальне і максимальне значення напруження σ_z, знімемо і встановимо відповідні прапорці:

Results - Stress (solid):

`Plot Settings` – **Plot data set edges** = `False`

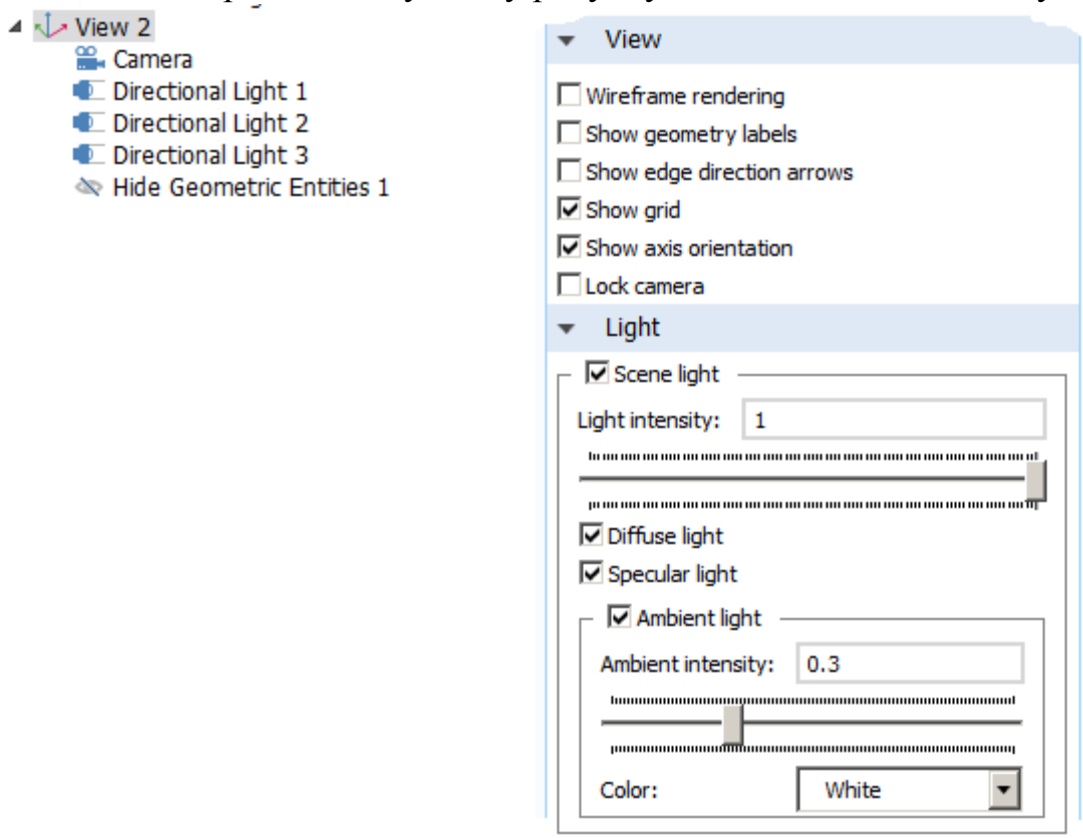
`Color Legend` – **Show mximum and minimum values** = `True`

LMB > Plot

Зверніть увагу на те, що хоча тіло стискається, максимальне (червоне) значення напруження σ_z додатне, тобто існують частини тіла, які розтягуються. Бетон добре витримує стискання, але погано працює на розтягнення. Отже в червоних частинах тіла можуть відбутися руйнування. Руйнування може відбутися і в стиснутій частині, якщо напруження стискання дуже велике. Щоб з'ясувати відбудеться руйнування чи ні, потрібно порівняти максимальні і мінімальні значення напруження з граничними значеннями для відповідної марки бетону.

Виникає питання, як роздивитися картину напружень σ_z на поверхні внутрішніх отворів. Для цього в системі існує інструмент, який дозволяє керувати відображенням частин геометрії і на графіку показувати не всі її елементи. Цим інструментом є вузли типу `View` (Вид). В них задаються налаштування, які визначають спосіб відображення геометрії в графічних вікнах. У ПВ `View` вузлів встановлюються ознаки (прапорці) відображення

службових елементів графіків: позначок геометричних елементів, координатної сітки і осей, спосіб освітлення і ступінь прозорості елементів геометрії. В тривимірних моделях у дочірніх підвузлах вузлів `View` визначаються джерела світла, налаштування камери спостереження і ознаки приховування деяких елементів геометрії. На наступному рисунку показана частина ПВ вузла `View`.



Зазвичай вузли `View` створюються в гілці **Definitions** компонента. Треба клацнути правою кнопкою миші по цьому вузлу і вибрати команду View з випадного меню. Один `View` вузол будується завжди при створенні компонента, але таких вузлів може бути скільки завгодно.

У вузлі `View` автоматично створюються підвузли трьох спрямованих джерел світла. Всього в системі існує чотири типи джерел світла:

- спрямоване джерело (Directional Light) поводиться як світло, промені якого виходять з нескінченності;
- точкове джерело (Point Light) схоже на лампочку, яка світить на всі боки і ви можете задавати її положення;
- ліхтар (Spotlight); користувач може задавати положення джерела світла і його напрямок;
- фара (Headlight) прикріплена до «камери» і світить від неї (тобто завжди світить від вас).

Для додавання до виду/сцени джерела світла необхідно до вузла `View` додати підвузол відповідного типу.

У автоматично створюваному підвузлі `Camera` розташовані налаштування камери. Вони задають кут зору, місцезнаходження, масштаб і перспективу камери. Ви можете уявити собі справжню камеру, яка розташована перед графічним вікном, і ви дивитесь на об’єкт прямо скрізь видошукач камери.

Модифікуючи параметри в ПВ вузла `Camera`, ви налаштовуєте камеру, змінюючи зображення, а не геометрію тіла.

Керувати камерою можна за допомогою кнопок миші:

- натискання правої кнопки миші в графічній області і переміщення миші зсовує зображення;
- натискання середньої кнопки миші (або колеса) в графічній області і переміщення миші масштабує зображення;
- утримування натиснутою клавішу `Ctrl` і однієї з кнопок миші, і переміщення миші змінює положення «точки спостереження/камери» і, як наслідок, змінює зображення (тільки для тривимірних зображень).
- утримування натиснутою клавішу `Alt` і однієї з кнопок миші, та переміщення миші обертає зображення навколо лінії зору.

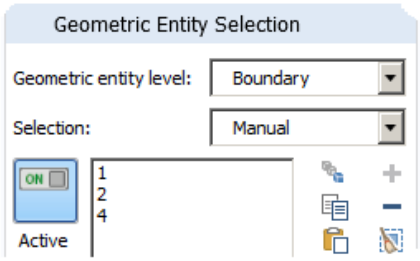
Рухаючись по графічному вікну ви зміщуєте камеру, а не саму геометрію.

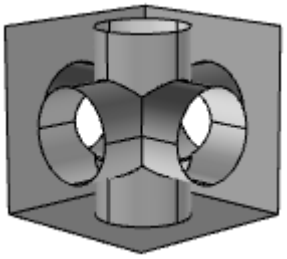
Для того, щоб сховати деякі елементи геометрії у вузлі **View** створюється дочірній вузол **Hide Geometric Entities**. Наприклад, спробуємо на попередньому зображенні сховати дві передні і верхню грань куба. Для цього потрібно створити «вид», в якому не будуть відображатися ці елементи геометрії. Виконайте команди

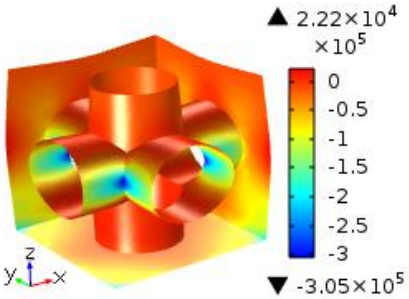
Component 1 – Definitions : RMB > View

Component 1 – Definitions – View 2 : RMB > Hide Geometric Entities

У ДМ створюється вузол **Hide Geometric Entities 1** (попередній рисунок ліворуч). В його ПВ в полі **Geometric entity level** виберіть `Boundary` рівень геометрії (щоб приховати елементи поверхні), і в секції вибору вкажіть індикатори `1, 2, 4` двох передніх і верхньої граней куба (наступний рисунок ліворуч). Для контролю можна клацнути кнопку Show Objects in Selection вгорі ПВ вузла **Hide Geometric Entities 1**. В графічній зоні буде побудовано зображення без зазначених поверхонь (наступний рисунок в центрі).







Перейдіть до вузла **Results - Stress (solid)** і в його ПВ введіть

`Plot Settings` – **View** = `View 2`

LMB > Plot

Поверхневий графік буде перебудований, і ви побачите зображення, яке показано на попередньому рисунку праворуч.

Отже налаштування в ПВ вузла **Hide Geometric Entities** дозволяють ховати деякі елементи геометрії (вони залишаються в геометрії, але не відображаються на графіках, в яких використовується батьківський «вид»). Зауважимо, що існує схожий вузол **Hide Geometric Objects** використання якого дозволяє ховати цілі об’єкти з геометрії компонента.

Зауваження. Підвузли типу `Hide Geometric Entities` створюються лише у `View` вузлах компонентів. У `View` вузли гілки **Results – Views** вони не додаються.

Щоб далі не було плутанини, поверніться у графічній групі **Stress (solid)** до попереднього виду.

Results - Stress (solid):

`Plot Settings` – **View** = `View 1`

LMB > Plot

Поверхневий графік відображає в кольорах величину на поверхні тіла. Якщо потрібно зобразити величину всередині тривимірного тіла, то для цього можна використати пошаровий (Slice) графік. Він показує розподіл величини всередині тіла, розфарбовуючи різними кольорами плоскі шари, які проходять скрізь тіло. Побудуємо один такий графік за допомогою наступних інструкцій.

`Results->`3D Plot Group

3D Plot Group 2:

`Data`– **Data set** = `Study 1/Solution 1`

`Plot Settings` – **View** = `View 1`

`3D Plot Group 2` **`->Add Plot`** – Slice

3D Plot Group 2 – Slice 1:

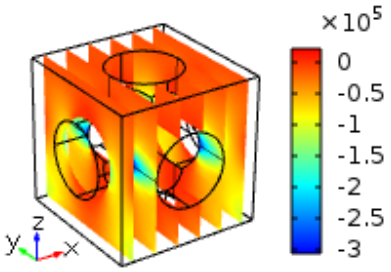
`Data`– **Data set** = `From parent`

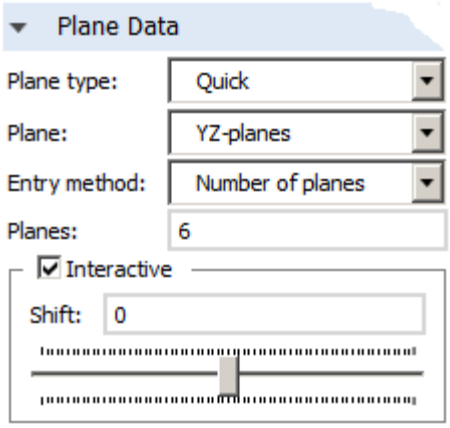
`Expression` – **Expression** = `solid.sz`

`Plane Data` – **Entry method, Planes** = `Number of planes,6`

`Coloring and Style` – **Color table** = `RainbowLight`

LMB > Plot





Зверніть увагу, що в ПВ вузла **Slice 1** в розділі **`Plane Data`** зазначається кількість зрізів (поле **Planes**). Крім того, якщо встановлено прапорець **Interactive**, то за допомогою бігунка **Shift** можна переміщати всю «пачку» зрізів як тверде тіло.

Зняти з геометрії «верхній шар» і зазирнути всередину моделі можна також за допомогою вузлів типу `Filter`. Вони створюються як підвузли графічних вузлів і містять логічний вираз з критерієм, що визначає які частини графіка зображати.

Створимо ще одну тривимірну графічну групу з поверхневим графіком, на якому будемо відображати переміщення (модуль вектора переміщення), що в інтерфейсі `Solid Mechanics` позначається змінною `disp`.

`Results->`3D Plot Group

3D Plot Group 3:

`Data`– Data set = `Study 1/Solution 1`

`Plot Settings` – **View** = `View 1`

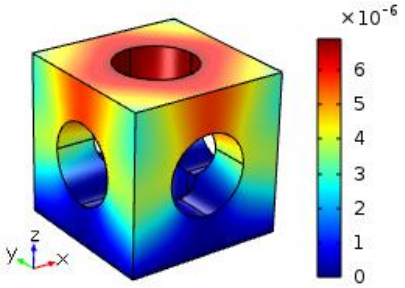
`3D Plot Group 3` **`->Add Plot`** – Surface

3D Plot Group 3 – Surface 1:

`Data`– Data set = `From parent`

`Expression` – **Expression** = `solid.disp`

LMB > Plot



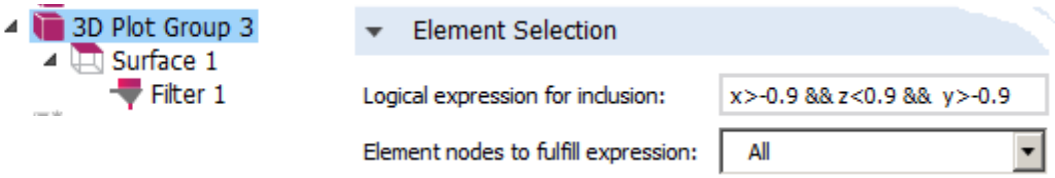
Тепер додамо до графіка фільтр.

3D Plot Group 3 – Surface 1: RMB > Filter

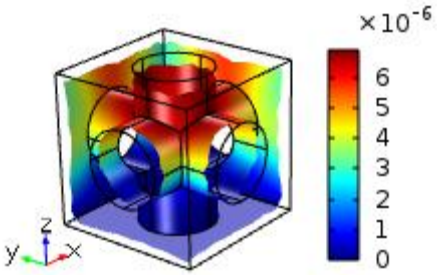
3D Plot Group 3 – Surface 1 – Filter 1:

`Element Selection` – **Logical expression for inclusion** = `x>-0.9 && z<0.9 && y>-0.9`

LMB > Plot



В результаті будується поверхневий графік величини переміщення, який показаний на наступному рисунку.



Тут логічний вираз `x>-0.9 && z<0.9 && y>-0.9` «зняв» з геометрії «передній шар» і залишив на графіку всі елементи, координати точок яких задовольняють заданому критерію.

Зауваження. Фільтри діють лише на джерелах даних, які спираються на СкЕ сітку. В графіках, що спираються на ДД типу `Grid`, фільтри не працюють.

Можна запропонувати ще одне корисне застосування «фільтрів». Повернемося до поверхневого графіка **Results – Stress (solid)** напруження σ_z. Ми бачили, що в деяких точках тіла напруження додатне $\sigma_z > 0$, тобто бетон розтягується. Щоб він не руйнувався при розтягуванні, зазвичай розтягнуті ділянки бетону армують. Але з графіка **Stress (solid)** не досить зрозуміло, які частини тіла розтягуються, а які ні. Спробуємо на поверхні тіла виділити кольором частини тіла, які зазнають розтягування. Для цього накладемо на існуючий графік інший поверхневий графік, який за допомогою відповідного фільтра буде відображати червоним (або іншим) кольором тільки ті частини поверхні, в яких напруження $\sigma_z > 0$ додатне. Для цього в нашу графічну групу додамо ще один `Surface` графік. Щоб він не відрізнявся по розташуванню від **Surface 1**, додамо для нього таку ж саму деформацію. Крім того, в якості виразу для забарвлення поверхні використаємо одиницю і для неї оберемо червоний колір. Якщо більше нічого не робити, то червоний графік закриє собою поверхневий графік напруження σ_z тому, що його вузол розташований в ДМ нижче. Але ми використаємо фільтр, який на новому графіку дозволить відобразити тільки частини поверхні з додатним напруженням $\sigma_z > 0$. В результаті закритою буде лише частина поверхні **Surface 1**.

Виконаємо інструкції, які реалізують описані побудови.

Results - Stress (solid): RMB > Surface

Results - Stress (solid) - Surface 2:

`Expression` – **Expression** = `1`

`Coloring and Style` – **Coloring, Color** = `Uniform, Red`

`Inherit Style` – **Plot** = `Surface 1`

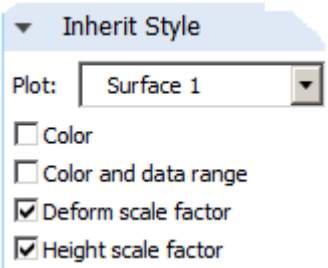
– **Color** = `False`

– **Color and data range** = `False`

– **Deform scale factor** = `True`

– **Height scale factor** = `True`

Пояснимо сенс полів розділу **`Inherit Style`** (успадкування стилю). Часто потрібно, щоб графіки, розташовані в одній групі, мали однакове кольорове оформлення і однакову деформацію. Ви можете в кожному з них задавати однакові параметри, але це клопітно. В ПВ багатьох графічних вузлів є розділ **`Inherit Style`** (успадкування стилю). Вибір в його списку **Plot** імені графічного вузла, означає використання параметрів оформлення цього вузла для поточного графіка. Графічний вузол може успадковувати стилі будь-якого з графіків його графічної групи, вузли яких розташовані вище в цій групі. Нижче списку **Plot** розташовані прапорці, які вказують, які елементи оформлення слід успадковувати.



Спадкування стилю також означає, що будь-які зміни оформлення в «батьківському» графіку автоматично призведуть до змін у всіх графіках, які успадковують його стиль.

В нашому прикладі ми відключили спадкування кольорової схеми і залишили спадкування деформацій. Вони успадковуються тільки тоді, коли для поточного вузла буде створено підвузол `Deformation`. Зробимо це.

Results - Stress (solid) - Surface 2:

RMB > Deformation

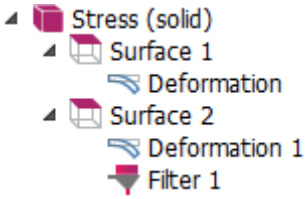
Залишилося додати фільтр. Для цього виконайте інструкції

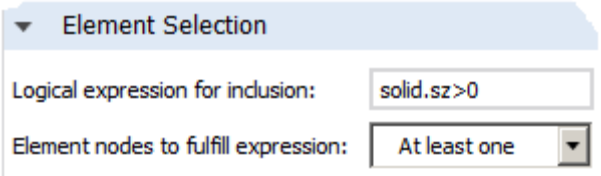
Stress (solid) - Surface 2: RMB > Filter

Stress (solid) - Surface 2 – Filter 1:

`Element Selection` – Logical expression for inclusion = `solid.sz>0`

`Element Selection` – Element nodes to fulfill expression = `At least one`





Вибір в списку **Element nodes to fulfill expression** значення `At least one` додає в вибірку всі скінченні елементи, для яких хоча б в одному вузлі задовольняється критерій, заданий в логічному виразі. Це приводить до забарвлення скінченних елементів, які повністю або частково задовольняють критерію.

Завершіть побудову графіка клацанням по кнопці Plot (наступний рисунок ліворуч).

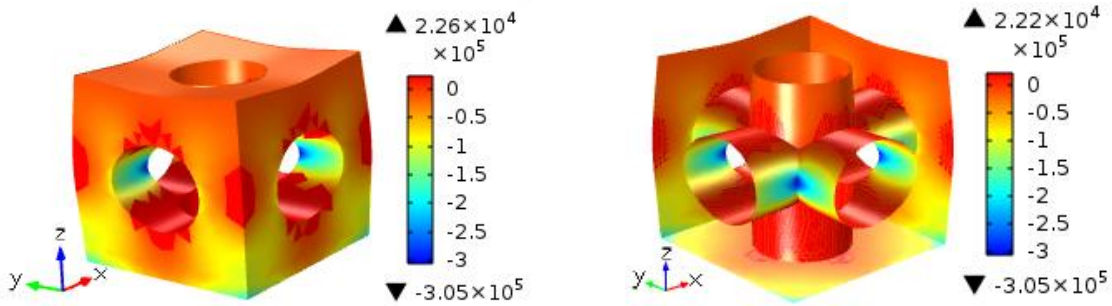
Тепер в ПВ вузла графічної групи **Stress (solid)** виберіть другий вид, в якому не відображаються три фронтальні грані.

Results - Stress (solid):

`Plot Settings` – View = `View 2`

LMB > Plot

Графік буде перебудований, і ви побачите зображення, яке показано на наступному рисунку праворуч.



Як бачите вид `View 2` було застосовано до обох графіків **Surface 1** і **Surface 2** графічної групи **Stress (solid)**.

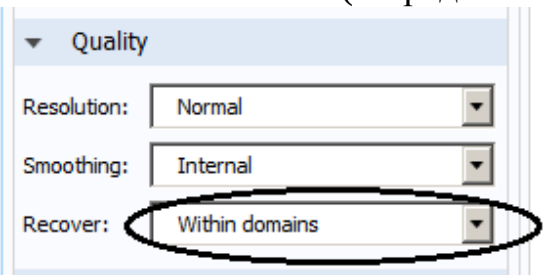
У вузлі **Stress (solid)** поверніться до попереднього виду:

Results - Stress (solid):

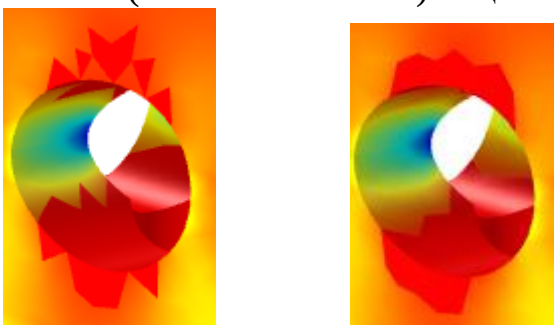
`Plot Settings` – **View** = `View 1`

LMB > Plot

В цьому місці доречно пояснити використання опції **Recover** (Відновлення) розділу **`Quality`** (Якість). У вузлі **Surface 2** розкрийте розділ **`Quality`** і в списку **Recover** виберіть `Within domains` (всередині областей).



Виконайте команду Plot і графік трохи зміниться. Щоб побачити ці зміни на наступному рисунку зображено околицю лівого отвору лівого графіка з виключеною (`Off`) і включеною (`Within domains`) опцією **Recover**.



Під час розв'язання задачі методом скінченних елементів у вузлах сітки обчислюються значення якої-небудь однієї основної величини, наприклад, вектора переміщення (`u,v,w`). Інші величини, в нашому прикладі напруження, зазвичай обчислюються за допомогою диференціювання основної величини. При цьому в COMSOL Multiphysics використовується спеціальний поліноміальний алгоритм обчислення похідних високого порядку. Цей алгоритм відновлює похідні з використанням значень основної величини на деякій множині скінченних елементів. Однак при побудові графіків для скорочення часу обчислення похідні відбудовуються за значеннями розв'язку в одному скінченному елементі. Це призводить до того, що на графіках величини, обчислювані з використанням таких похідних, мають меншу точність, ніж розраховані в розв'язку. Наприклад, другі похідні дорівнюють

нулю, коли в розрахунках використовуються лінійні елементи. Якщо використовувати поліноміальний алгоритм і значення розв'язку в вузлах декількох скінченних елементів, похідні відновлюються точніше (похідні відновлюються на основі множини елементів, а не в кожному елементі окремо). Оскільки ця техніка вимагає більше часу для обчислень, то за замовчуванням при побудові графіків вона відключена. В графічних вузлах вона включається при використанні опції **Recover** = `Within domains`. Це відновлює похідні в областях, не виконуючи згладжування розв'язку через границі.

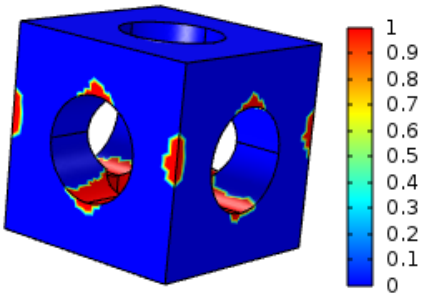
Таким чином, в нашій моделі налаштування **Recover** = `Within domains` обчислює напруження на основі багатьох скінченних елементів, а не одного. Остання пара графіків показує, як може змінитися зображення при включенні опції відновлювання похідних.

Відобразити зони додатного напруження можна інакше. Для цього достатньо побудувати графік виразу `solid.sz>0`. Це логічний вираз, який може набувати значення 1 (істина) або 0 (брехня). В областях, де його значення дорівнює 1, напруження розтягувальні.

Додамо нову 3D графічну групу, скопіюємо з групи **Stress (solid)** в буфер обміну вузол **Surface 1**, і вставимо його з буфера в створену графічну групу. В ній для вузла **Surface 1** відключимо деформування, і в полі **Expression** запишемо вираз `solid.sz>0`. Ці дії реалізуємо наступною послідовністю інструкцій:

`Results -> 3D Plot Group`
Stress (solid) - Surface 1: RMB > Copy
3D Plot Group 4: RMB > Paste Surface
3D Plot Group 4 - Surface 1:
`Expression` – **Expression** = `solid.sz>0`
`Plot Settings` – **View** = `View 1`
`Quality` – **Recover** = `Within domains`
3D Plot Group 4 - Surface 1 - Deformation: RMB > Disable
3D Plot Group 4: LMB > Plot

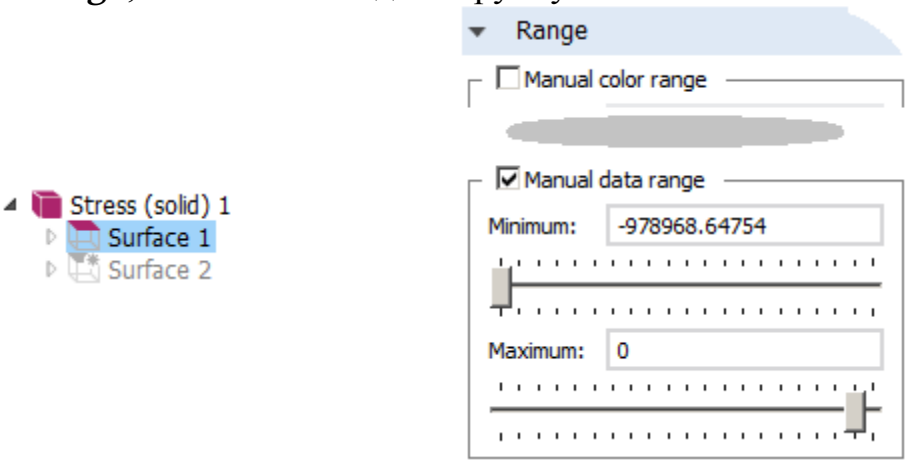
В результаті буде побудовано наступний двокольоровий графік.



Тут ми ще включили опцію відновлювання похідних **Recover** = `Within domains`.

Виділити зону розтягувальних напружень можна простіше, задавши максимальне значення виразу, який використовується при фарбуванні куба. Зазвичай екстремальні значення виразів в ПВ графічних вузлів обчислюються і

відображаються автоматично в розділі **Range**. Але, якщо встановити прапорець **Manual data range**, то їх можно задати вручну.



Ручні налаштування не змінюють колір в обраному діапазоні, а відображають лише ті дані, які лежать між заданим мінімумом і максимумом. Збільшуючи мінімум або зменшуючи максимум, ви будете видаляти точки даних з графіка.

Поверніться до вузла **Results - Stress (solid)**. Деактивуйте в ньому вузол **Surface 2**, а у вузлі **Surface 1** деактивуйте підвузол деформацій (ви можете створити копію графічної групи **Stress (solid)**, в якій видалити зайві вузли).

Stress (solid) - Surface 2: RMB > Disable

Stress (solid) - Surface 1 - Deformation: RMB > Disable

В ПВ вузла **Surface 1** встановіть максимальне значення виразу (в нашому прикладі напруження), яке слід відображати на графіку, наприклад, `−60000`.

Stress (solid) - Surface 1:

Range – Manual data range = `True`

– Manual data range – Maximum = `−60000`

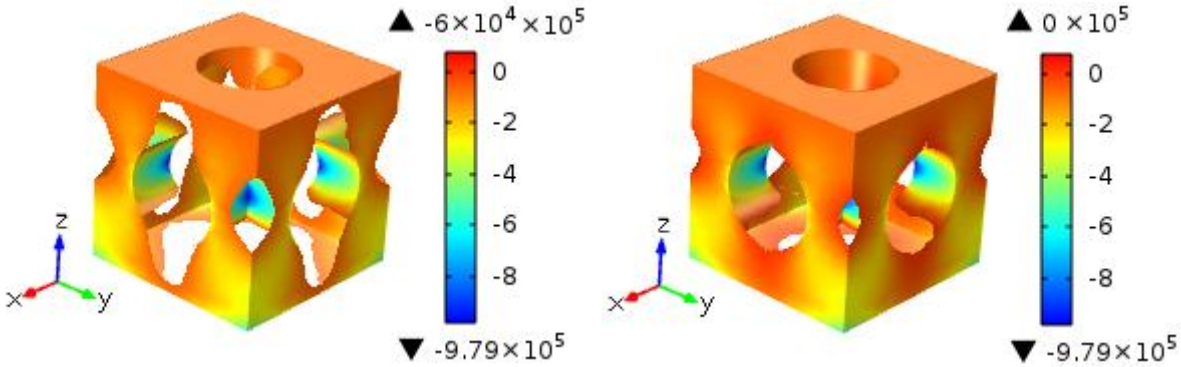
LMB >Plot (*наступний рисунок ліворуч*)

Якщо максимумом обрати нуль, то на поверхні куба не будуть відображатися зони з розтягувальним напруженням.

Stress (solid) - Surface 1:

Range – Manual data range – Maximum = `0`

LMB >Plot (*наступний рисунок правруч*)



«Дірки» на поверхні відповідають зонам, в яких вираз з поля **Expression** перевищує заданий максимум.

Розглянемо ще одне службове застосування вузлів `Deformation`. Припустимо, що ми бажаємо порівняти напруження σ_z в двох перерізах тіла

$z = -0.95$ та $z = -0.5$. Це можна було б зробити, використовуючи джерела даних, які співпадають з цими перерізами.

Results – Data Sets: RMB > Cut Plane

Data Sets - Cut Plane 1:

Data – **Data set** = Study 1/Solution 1

Plane Data – **Plane** = XY-planes

Plane Data – **Z-coordinate** =-0.95

Results – Data Sets: RMB > Cut Plane

Data Sets - Cut Plane 2:

Data – **Data set** = Study 1/Solution 1

Plane Data – **Plane** = XY-planes

Plane Data – **Z-coordinate** =-0.5

Results->3D Plot Group

3D Plot Group 5:

Data– **Data set** = None

3D Plot Group 5 **-> Add Plot** – Surface

3D Plot Group 5 – Surface 1:

Data– **Data set** = Cut Plane 1

Expression – **Expression** = solid.sz

3D Plot Group 5 **-> Add Plot** – Surface

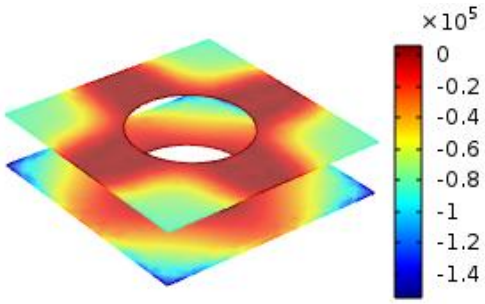
3D Plot Group 5 – Surface 2:

Data– **Data set** = Cut Plane 2

Expression – **Expression** = solid.sz

Inherit Style – **Plot** = Surface 1

LMB > Plot



Порівнювати зображення отриманих перерізів незручно. Це можна зробити краще, якщо розташувати перерізи поруч на одному графіку. Для цього треба побудувати їх зображення в спільній двовимірній графічній групі, і друге зображення зсунути відносно першого (без зсуву один переріз закриє інший). Зробимо це.

Results->2D Plot Group

2D Plot Group 6:

Data– **Data set** = None

2D Plot Group 6 **-> Add Plot** – Surface

2D Plot Group 6 – Surface 1:

Data– **Data set** = Cut Plane 1

Expression – **Expression** = solid.sz

2D Plot Group 6 **->Add Plot** – Surface

2D Plot Group 6 – Surface 2:

Data– **Data set** = Cut Plane 2

Expression – **Expression** = solid.sz

Coloring and Style – **Color legend** = False

Inherit Style – **Plot** = Surface 1

– **Color** = True

– **Color and data range** = True

– **Deform scale factor** = False

– **Height scale factor** = False

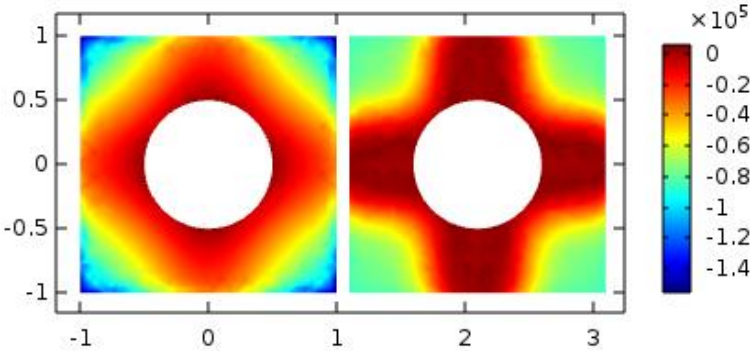
RMB > Deformation

2D Plot Group 6 – Surface 2 – Deformation 1:

Expression – **x component, y component** = 2.1,0 (*два поля*)

Scale – **Scale factor** = True, 1

LMB > Plot



Отже друге зображення зсунуто в графічній зоні на 2.1 одниці (метри) вправо.

На таких зображеннях корисно позначати положення точок максимального і мінімального значення. Для цього призначені графічні вузли типу Max/Min Surface. Графіки такого типу знаходять і будують на поверхні тіла/області точки мінімуму і максимуму виразу, який введено в полі Expression. Одночасно з цим створюється таблиця, в якій друкуються координати цих точок, і значення екстремумів в них.

Виконайте інструкції

2D Plot Group 6: RMB > More Plots - Max/Min Surface

2D Plot Group 6 – Max/Min Surface 1:

Data– **Data set** = Cut Plane 1

Expression – **Expression** = solid.sz

2D Plot Group 6: RMB > More Plots - Max/Min Surface

2D Plot Group 6 – Max/Min Surface 2:

Data– **Data set** = Cut Plane 2

Expression – **Expression** = solid.sz

Inherit Style – **Plot** = Surface 2

– **Color**= False

– **Deform scale factor** = True

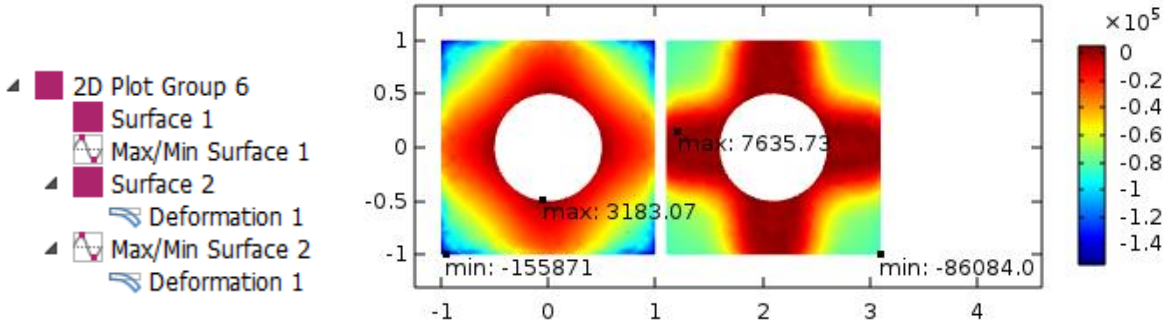
RMB > Deformation

2D Plot Group 6 – Max/Min Surface 2 – Deformation 1 :

Expression – **x component, y component** = 2.1,0 (*два поля*)

LMB > Plot

В результаті ми отримуємо графік напружень, на якому точками позначені максимальне і мінімальне (по перерізам) значення величини σ_z (наступний рисунок праворуч). Для зручності ми перемістили вузли Max/Min Surface в межах графічної групи так, як показано ліворуч.



Зверніть увагу на те, що графік **Max/Min Surface 2** успадковує від графіка **Surface 2** масштабний коефіцієнт, а не вектор зсуву, який треба задати явно. Зверніть також увагу на те, що в гілці **Tables** створюються дві таблиці з координатами точок екстремумів (по одній на кожний вузол **Max/Min Surface**).

Крім графіків, які позначають точки максимального і мінімального значення виразу по поверхні, можна будувати графіки максимуму і мінімуму по лінії або за об'ємом. Графік Max/Min Line знаходить і будує точки мінімуму і максимуму на лінії. Графік типу Max/Min Volume знаходить і малює точки максимуму і мінімуму в об’ємі тіла.

Перейдемо до розглядання лінійних графіків в тривимірних моделях. Їх сенс дещо відрізняється від одновимірних графіків функцій. Для пояснення створимо лінійний графік в новій тривимірній графічній групі.

Results->3D Plot Group

3D Plot Group 7:

Data– **Data set** = Study 1/Solution 1

Plot Settings – **Plot data set edges** = False

3D Plot Group 6 **-> Add Plot** – Line

3D Plot Group 7 – Line 1:

Data– **Data set** = From parent

Expression – **Expression** = solid.disp

Coloring and Style – **Line type** = Tube

– **Tube radius expression** = 0.02

– **Radius scale factor** = 1

RMB > Deformation

3D Plot Group 7 – Line 1 – Deformation 1:

Expression – **X component** = u

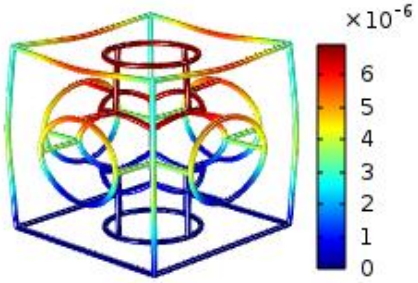
– **Y component** = v

– **Z component** = `w`

`Scale` – **Scale factor** = `True, 25000`

LMB > Plot

Клацніть ще по кнопці Show Grid, яка розташована вгорі графічного вікна, щоб прибрати відображення координатної сітки.



Отже `Line` графік «вибрав» з джерела даних `Study 1/Solution 1` всі одновимірні графічні елементи і розфарбував їх згідно виразу, зазначеному в полі **Expression**. В нашому випадку вираз `solid.disp` представляє модуль вектора переміщення. Зверніть увагу на те, що лінійний графік можна деформувати, додавши до нього дочірній вузол **Deformation**.

Таким чином, `Line` графік знаходить в двовимірному або тривимірному джерелі даних всі одновимірні елементи, наявні в геометрії, і розфарбовує їх відповідно до виразу, заданому в полі **Expression**.

Зауваження . В тривимірному випадку `Surface` графік діє аналогічно до `Line` графіків: розфарбовує відповідно до зазначеного виразу всі двовимірні елементи з геометрії тривимірного джерела даних.

Для побудови лінійного графіка не на всіх ребрах тіла можна створити джерело даних типу `Edge 3D` і вказати для нього бажані ребра.

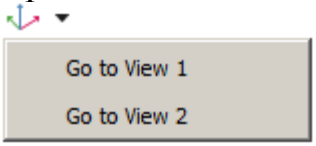
Results – Data Sets : RMB > Edge 3D

Data Sets - Edge 3D 1:

`Data` – **Data set** = `Study 1/Solution 1`

`Selection` – **Selection** =`1,14,77,84` (*вертикальні ребра куба*)

Зауваження. Якщо в графічному вікні при побудові набору даних **Edge 3D 1** використовується вид **View 2**, тобто фронтальні грані не відображаються, то переключіться на вид **View 1** за допомогою кнопки Go to Default 3D View (вона розташована вгорі ГВ), обравши бажаний вид з випадного списку.



Перейдемо до побудови графіка.

`Results->3D Plot Group`

3D Plot Group 8:

`Data`– **Data set** = `Edge 3D 1`

`3D Plot Group 8` **-> `Add Plot`** – Line

3D Plot Group 8 – Line 1:

`Data`– **Data set** = `From parent`

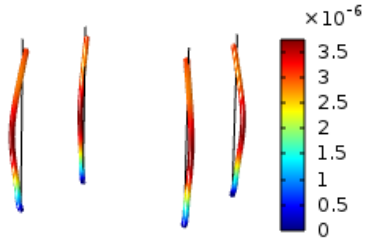
Expression – **Expression** = solid.disp
Coloring and Style – **Line type** = Tube
– **Tube radius expression** = 0.02
– **Radius scale factor** = 1

RMB > Deformation

3D Plot Group 8– Line 1 – Deformation 1:

Expression – **X component** = u
– **Y component** = v
– **Z component** = w
Scale – **Scale factor** = True, 50000

LMB > Plot



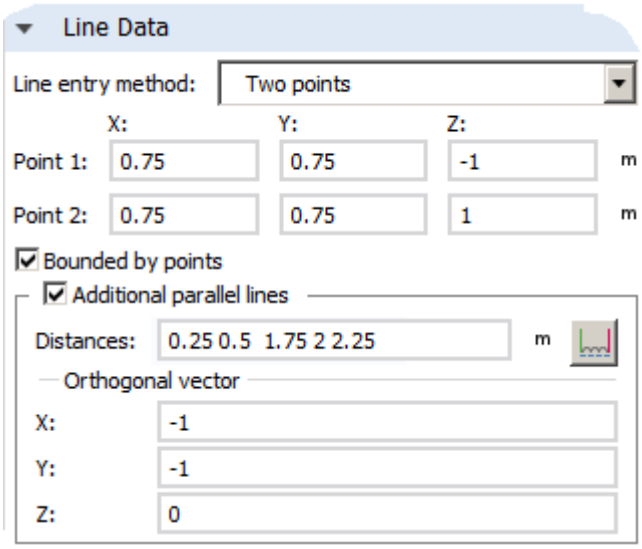
Для лінійного графіка можна створити окреме лінійне джерело даних. Для цього виконайте наступні інструкції.

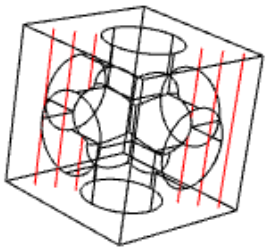
Results – Data Sets : RMB > Cut Line 3D

Data Sets - Cut Line 3D 1:

Data – **Data set** = Study 1/Solution 1

Далі заповніть ПВ створеного вузла **Cut Line 3D 1** згідно з наступним рисунком.





Тут показано, що лінійне джерело даних складається з базового відрізка, обмеженого двома точками із заданими координатами. Крім нього створюється ще 5 паралельних відрізків, розташованих на заданій відстані від базового відрізка в напрямку зазначеного вектора. Клацнувши по кнопці Plot, ви побудуєте зображення відрізків, з яких складається набір даних (попередній рисунок праворуч).

Тепер в кольорах зобразимо на заданих відрізках величину переміщення і додамо до лінійного графіка деформацію.

3D Plot Group 8 **-> Add Plot** – Line

3D Plot Group 8 – Line 2:

Data – **Data set** = Cut Line 3D 1

Expression – **Expression** = solid.disp

Coloring and Style – **Line type** = Tube

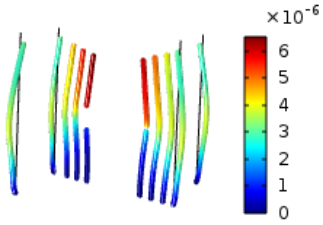
Coloring and Style – **Tube radius expression** = 0.02

Coloring and Style – **Radius scale factor** = 1

Inherit Style – **Plot** = Line 1

RMB > Deformation

LMB > Plot



Зверніть увагу на те, що лінійний графік **Line 2** успадковує всі стилі оформлення лінійного графіка **Line 1**, включно з забарвленням і масштабним коефіцієнтом.

Якщо одновимірне джерело даних складається з одного відрізка, то на ньому можна побудувати звичайний одновимірний графік, який можна розфарбувати схожим з лінійними графіками чином.

Results – Data Sets : RMB > Edge 3D

Data Sets - Edge 3D 2:

Data – **Data set** = Study 1/Solution 1

Selection – **Selection** =4 (*верхнє горизонтальне ребро куба*)

Results->1D Plot Group

1D Plot Group 9:

Data – **Data set** = Edge 3D 2

1D Plot Group 9 **-> Add Plot** – Line Graph

1D Plot Group 9 - Line Graph 1:

Data – **Data set** = From parent

y-Axis Data – **Expression** = solid.disp

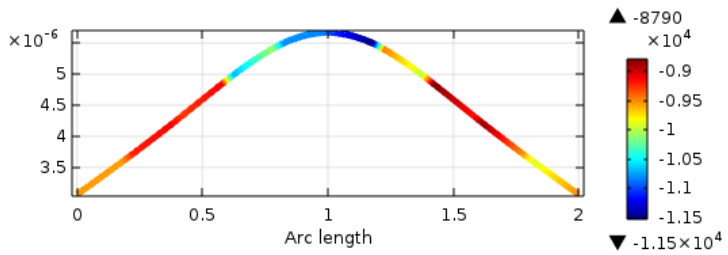
Coloring and Style – **Line, Width** = Solid, 5 (*два поля*)

RMB > Color Expression

1D Plot Group 9 - Line Graph 1 - Color Expression 1:

Expression – **Expression** = solid.sz

LMB > Plot



Отже крива зображає модуль вектора переміщення точок ребра, а колір – напруження σ_z в цих точках. Щоб задати колір лінії, який відповідає напруженню σ_z, до вузла **Line Graph 1** ми додали дочірній вузол **Color Expression 1**.

Покажемо, як можна побудувати зображення, яке наведено на початку цього прикладу. Воно ілюструє дію рівномірно розподіленої вертикальної сили на верхню грань тіла. Зображення самого тіла будується в тривимірній графічній групі за допомогою `Surface` графіка з одиничним виразом, щоб колір всіх точок тіла був однаковий. Вектори на поверхні будуються як стрілковий графік векторного поля (`0,0,-1`). `Arrow Surface` графік (поверхнева векторна діаграма) прив'язує векторні поля до поверхонь (тобто початок або кінець стрілок розташований на поверхні). Щоб вектори були намальовані лише на верхній грані, для неї було створено окреме джерело даних `Surface` типу.

Results – Data Sets : RMB > Surface

Data Sets - Surface 1:

`Data` – **Data set** = `Study 1/Solution 1`

`Parameterization` – **x-and-y-axes** = `XY-plane`

`Selection` – **Selection** = `4` (*верхня грань куба*)

`Results->`3D Plot Group

`3D Plot Group 10` **`->Add Plot`** – Surface

3D Plot Group 10 – Surface 1:

`Data`– **Data set** = `Study 1/Solution 1`

`Expression` – **Expression** = `1`

`Coloring and Style` – **Color table** = `GrayScale`

– **Color legend** = `False`

`3D Plot Group 10` **`->Add Plot`** – Arrow Surface

3D Plot Group 10 – Arrow Surface 1:

`Data`– **Data set** = `Surface 1`

`Expression` – **x component** = `0`

– **y component** = `0`

– **z component** = `-1`

`Coloring and Style` – **Arrow base** = `Head`

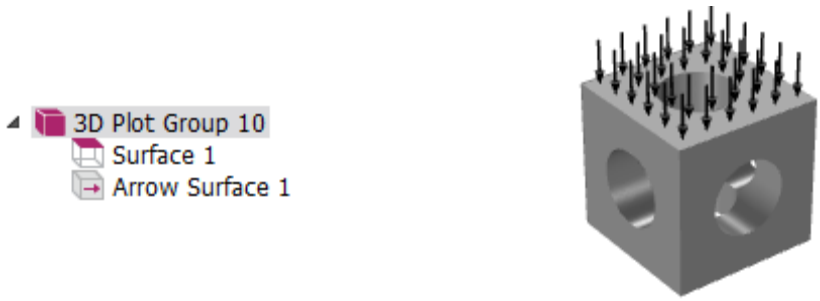
– **Scale factor** = `True, 0.6`

– **Placement** = `Uniform`

– **Number of arrows** = `30`

– **Color** = `Black`

LMB > Plot



Значення **Arrow base** = `Head` вказує на те, що на поверхні повинні бути розташовані кінці стрілок.

Звичайно, що в прикладних задачах замість діаграми сталого векторного поля (`0,0,-1`) використовуються розрахункові фізичні векторні поля.

Окрім поверхневих векторних діаграм можна будувати лінійні та об'ємні векторні діаграми.

Вправа 1. Побудуйте на поверхні тіла діаграму векторного поля переміщення (`u,v,w`).

Існує можливість створити копію основного джерела даних `Study 1/Solution 1`, в якому використати не всі елементи геометрії. При створенні вузла-копії в його підвузлі **Selection** можна обрати рівень геометричної сутності (level), і потім вказати потрібні елементи геометрії. Це схоже на приховування геометричних елементів у вузлах `View`. Наприклад, зобразимо тільки зовнішню поверхню нашого тіла і на ній контурний графік величини переміщення.

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** = `Boundary`

`Geometric Entity Selection` – **Selection** = `1,2,3,4,9,30` (*зовнішні поверхні тіла*)

Після створення копії джерела даних для неї (копії) ми додали підвузол **Selection**, який дозволяє обрати для цієї копії рівень геометрії, який буде використовуватися. В нашому випадку ми обрали зовнішні границі тіла.

Зверніть також увагу на те, що після створення копії ДД автоматично перейменовується вузол `Study 1/Solution 1`, який тепер всюди в моделі буде називатися `Study 1/Solution 1(1)`.

Новий набір даних `Study 1/Solution 1(2)` можна використовувати при побудові графіків.

`Results->`3D Plot Group

3D Plot Group 11:

`Data`– **Data set** = `Study 1/Solution 1(2)`

`3D Plot Group 11` **`->Add Plot`** – Surface

3D Plot Group 11 – Surface 1:

`Data`– **Data set** = `From parent`

`Expression` – **Expression** = `1`

Coloring and Style – **Color table** = GrayScale

Coloring and Style – **Color legend** = False

3D Plot Group 11 **-> Add Plot** – Contour

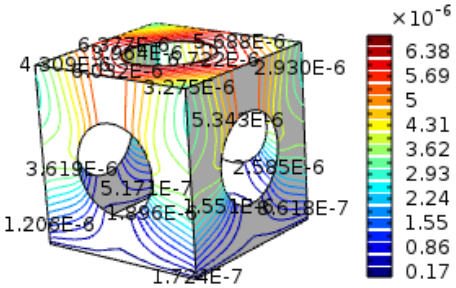
3D Plot Group 11 – Contour 1:

Data– **Data set** = From parent

Expression – **Expression** = solid.disp

Coloring and Style – **Level labels** = True

LMB > Plot



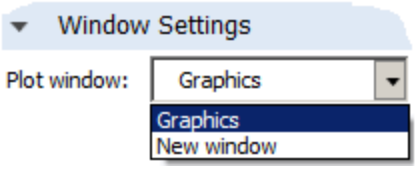
Використання для контурного графіка прапорця **Level labels**=True встановлює числові значення на контурних лініях.

В копії ДД можна обрати **Geometric entity level** = Edge. Тоді розв'язок буде дублюватися на кромках/ребрах тіла. Якщо при цьому задати

Geometric Entity Selection – **Selection** = All Edges

то нове ДД можна використати в лінійному графіку для креслення ребер геометрії (**Expression**=1, **Color**=Black). Використання вузла Deformation додасть до контуру геометрії деформацію.

При побудові графіків завжди є можливість відображати їх в окремому вікні. Для цього в ПВ графічних груп є розділ **Window Settings**, в якому виконується потрібний вибір.



Звичайно, що вікна графіків можна перетягати, розташовуючи кілька вікон поруч.

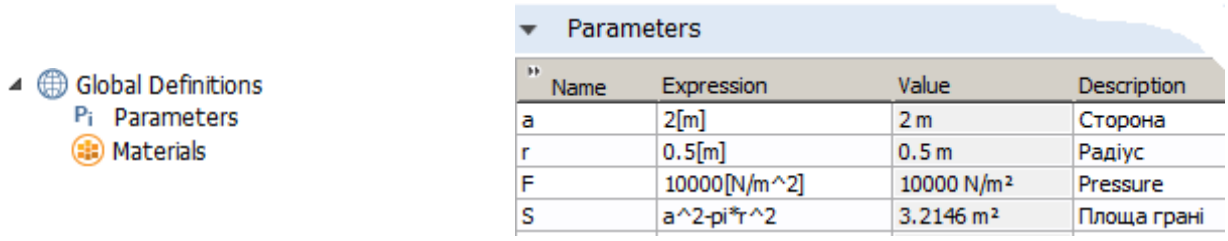
Збережіть файл, наприклад, з ім'ям cube_with_holes.mph. Ми продовжимо роботу з ним в наступному параграфі.

1.5.4. Параметри, змінні, зонди (probes).

Параметри. В багатьох виразах Comsol Multiphysics застосовуються параметри (Parameters) – іменовані числові константи, які можна використовувати в будь-яких місцях моделі замість чисел. Імена та значення параметрів задаються в таблиці параметрів в ПВ вузла **Parameters** глобальної гілки визначень **Global Definitions**. Сам вузол параметрів створюється командою

Home **-> Definitions** – Parameters

або аналогічною командою з випадного меню.



В таблиці параметрів в колонці **Name** вводять імена, а в колонці **Expression** – значення параметрів. Поля в стовпці **Value** заповнюються автоматично. В колонці **Description** вводяться необов'язкові коментарі. Вони іноді з'являються в позначках на графіках, в яких не передбачено використання кирилиці.

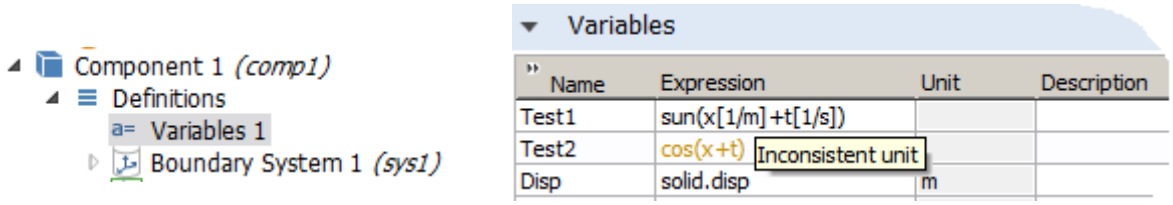
Змінні. Крім параметрів в системі використовуються змінні (Variables). Змінні – це назви просторових координат (наприклад, x, y, z), ідентифікатор часу `t`, імена шуканих величин, наприклад, T - температура. В прикладних режимах існує багато спеціальних змінних. Користувач може створювати власні змінні з інших, використовуючи припустимі операції. Змінні роблять модель простішою для розуміння шляхом введення коротких та наочних імен для складних виразів.

Зауваження. Не використовуйте літери кирилиці в іменах параметрів і змінних.

Параметр являє «статичну змінну», що ініціалізується один раз на початку дослідження (study) і не змінює свого значення впродовж всього процесу обчислення, в той час як змінна (variable) здатна це робити.

Змінні користувача задаються в таблицях вузлів **Variables**. Ці вузли створюються в глобальній гілці визначень **Global Definitions**, або локально в межах компонента. Наприклад, вузол локальних змінних можна створити командою

`Home` **`->Definitions`** – Variables – Local Variables



Вирази для обчислювання змінних можуть містити числа, параметри, математичні або фізичні константи, інші змінні, функції від змінних, знаки математичних операцій, та оператори, наприклад, диференціювання, інтегрування або обчислювання середнього значення. Змінні можуть також залежати від шуканих в задачі величин та їх похідних.

Більшість змінних, таких як координати, матеріальні властивості чи шукані величини, залежать від розрахункової області і являють «полеві» змінні, тобто вони залежать від просторових координат та часу. Наприклад, `solid.rho` – вбудована змінна густини в режимі `Solid Mechanics`, представляє її як функцію $\rho(x, y, z, t)$. Ви можете побудувати графік цієї функції, але не зможете використати в ЗДР, навіть якщо вона є сталою.

Змінні стають доступними після виконання команди Compute. При внесенні модифікацій в таблицю змінних, ви повинні виконати команду Update Solution, щоб прийняти додані зміни до уваги, або заново переобчислити модель. При цьому слід мати на увазі, що до виконання команди Compute перший раз, вирази в стовпці **Expression** майже завжди будуть виділені помаранчевим кольором тому, що система ще нічого не знає про імена змінних. Після виконання команди Compute змінні стають відомими, і система може проконтролювати вираз і його розмірність.

Параметри і змінні глобальної (зовнішньої) гілки визначень **Global Definitions** можна використовувати в будь-яких місцях моделі. Елементи локальної гілки **Component – Definitions** призначені для використання в межах компонента, але можуть застосовуватися і поза компонентом за допомогою точкової нотації. Отже параметри завжди глобальні, а змінні можуть бути глобальними і локальними в залежності від того, де ви їх визначили.

На відміну від параметрів змінні не можна застосовувати в гілках **Geometry** (наприклад, при визначенні розмірів геометричних фігур) та **Mesh**, а в гілках **Study** їх можна використовувати лише обмежено.

В Comsol Multiphysics існує ще один специфічний тип змінних, які називаються зондами (Probes). Вони створюються в локальних вузлах **Definitions** компонентів і являють собою скаляри (сталі, функції часу або параметрів, але не координат). Зазвичай вони характеризують якусь скалярну властивість компонента, наприклад, середнє значення розрахункової величини або значення величини в точці. Якщо задача динамічна, то ці змінні, можуть залежати від часу. Їх можна використовувати в коефіцієнтах рівнянь, граничних умовах, в умовах контролю або зупинки обчислень (в гілці **Study**). Зондові змінні (probe variables) використовуються також після завершення обчислень для аналізу результатів.

Продемонструємо використання змінних, продовживши роботу з файлом `cube_with_holes.mph`, який було створено в попередньому параграфі. Відкрийте цей файл.

Припустимо, що нам потрібно обчислити загальну силу, яку чинить тіло на основу. Вона дорівнює вазі тіла плюс зовнішня сила, яка діє на тіло зверху. Для виконання такого обчислення в COMSOL Multiphysics існує декілька інструментів. Найпростіший з них – це використання в гілці **Derived Values** вузла типу **Volume Integration**. Але цей спосіб доступний лише на етапі постобробки результатів (в гілці **Results**). Якби вам було потрібно використати значення об’єму тіла або загальну силу, яку воно чинить на основу, в іншому компоненті або всередині математичного виразу поточного компонента, то такий підхід не спрацює. Загальний спосіб полягає в використанні глобальних або локальних змінних і зондів. Вони можуть зберігати значення об’єму тіла, густину його речовини або якесь інше значення, що характеризує тіло загалом. Якщо зонд, локальна або глобальна змінна створені, то їх можна використовувати в будь-якому місці моделі: при побудові графіків, в виразах поточного компонента, або в іншому компоненті моделі. При цьому для зондів і

локальних змінних бажано (а часом обов’язково) використовувати точкову нотацію.

Глобальні змінні корисні переважно для позначення величин, які не залежать від точок обчислювання. Наприклад, вони можуть позначати змінні, які обчислюються в компоненті шляхом розв’язання ЗДР або алгебраїчного рівняння, і значення яких не залежить від геометрії моделі. Якщо змінна характеризує тіло, то ймовірно, її слід створювати як локальну до компонента, в якому створено геометрію тіла. Якщо змінна характеризує не тільки тіло, а залежить від зовнішніх факторів, то її логічно визначати як глобальну змінну.

Для обчислення змінної, яка представляє об’єм чи масу тіла потрібно виконувати інтегрування, але таке, яке може бути викликано з будь якого місця моделі. Операція інтегрування, яка виконується в межах компонента і може визиватися в будь якому місці моделі відноситься до групи операторів зв’язку (Component Couplings). Такі оператори можуть використовуватися для передачі значень між моделями різних фізичних процесів в межах одного компонента або між компонентами, а також на етапі обробки результатів. Характер значень, що передаються, може бути різним. Зокрема, це можуть бути деякі інтегральні характеристики моделі, які обчислюються оператором зв’язку `Integration`.

Оператори зв'язку (Coupling Operators) визначаються в гілці **Definitions** компонента, з якого вибираються дані. Тут оператори тільки визначаються, але не обчислюються. Однак після створення їх можна використовувати на будь-якому етапі моделювання в будь-якому місці поточного або іншого компонента. Для створення операторів зв'язку використовується кнопка - список Component Coupling панелі інструментів `Definitions` або однойменна команда з випадного меню локального вузла **Definitions**.

Припустимо, що ми бажаємо обчислити об'єм тіла за допомогою інтеграла $\iiint\limits_V dx\,dy\,dz$. Для цього зручно використовувати оператор зв'язку `Integration`. Подивимося, як це робиться.

`Definitions` -> **`Component Coupling`** – Integration

Активуйте ПВ створеного вузла **Component 1 (comp1) - Definitions - Integration 1 (intop1)**. Ім'я оператора інтегрування `intop1` вказується в полі **Operator name**. Далі введіть:

Label: = `Інтеграл за об'ємом`

`Source Selection` – **Geometric entity level** = `Domain`

Остання інструкція вказує «геометричний рівень» області інтегрування (лінія, поверхня, тривимірне тіло). В асоційованому графічному вікні клацніть по області тіла, і його індикатор 1 з'явиться в списку **Selection**, вказуючи, тим самим, конкретну область.

Обновіть розв’язок, клацнувши по кнопці Update Solution в ПВ вузла **Study 1**.

Тепер оператор `intop1` створений і доступний. Продемонструємо його використання в нашій моделі.

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

Results - Derived Values - Global Evaluation 1:

Expression – **Expression** = comp1.intop1(1)

LMB > Evaluate – New Table

В моделі з'явиться таблиця **Table 1** і в ній комірка з результатом обчислення. Оскільки інтегрування одиниці виконувалося по всьому тілу, то результатом є об'єм тіла.

comp1.intop1(1) (m^3)
4.7019

Як бачите, на етапі постобробки, використання оператора зв'язку Integration ідентично використанню залежної величини Volume Integration (тобто вузла **Derived Values – Integration – Volume Integration**).

Зауваження. В списку **Geometric entity level** вузла **Definitions - Інтеграл за об'ємом (intop1)** рівнем геометрії для інтегрування можна обрати Boundary (поверхня), Edge (ребро) і Point (точка). При виборі рівня геометрії Edge буде обчислюватися криволінійний інтеграл. Якщо в якості рівня інтегрування вказати точку, то оператор Integration поверне значення виразу в точці. Якщо на етапі створення оператора інтегрування intop(expr) вибрати кілька точок, то буде обчислюватися сума значень виразу в цих точках.

Оператор зв'язку Integration можна використовувати не тільки для обчислення значень на етапі обробки результатів. Його можна використовувати на будь-якому етапі моделювання, в тому числі при створенні глобальних і локальних змінних. Наприклад, створимо локальну змінну, яка являє об'єм тіла.

Якщо в вашій моделі ще немає вузла локальних змінних **Component 1 – Definitions – Variables 1**, то створіть його командою

Home -> **Definitions** – Variables –> Local Variables

Якщо такий вузол вже є, то очистіть рядки його таблиці змінних за допомогою кнопки Delete, яка розташована під таблицею.

В ПВ вузла **Component 1 – Definitions – Variables 1** в першому рядку таблиці змінних введіть:

Variables – **Name** = Vol1

– **Expression** = comp1.intop1(1)

Study 1: RMB > Update Solution

Тепер перевіримо працездатність змінної Vol1, надрукувавши її значення.

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

Results - Derived Values - Global Evaluation 2:

Expression – **Expression** = comp1.Vol1

LMB > Evaluate – Table 1

intop1(1) (m^3)	Vol1 (m^3)
4.7019	4.7019

Один і той же оператор intop1 можна використовувати для інтегрування різних виразів. Наприклад, intop1(густина) обчислить масу тіла.

Створимо локальну змінну `Mass1`, що містить значення маси тіла. Для цього в другому рядку таблиці вузла **Component 1 – Definitions – Variables 1** введіть:

`Variables` – **Name** = `Mass1`

– **Expression** = `comp1.intop1(comp1.mat1.def.rho)`

Study 1: RMB > Update Solution

Зверніть увагу на те, як використовується ім'я, що представляє густину тіла. Для доступу до властивості `rho` матеріалу `mat1` потрібно використовувати проміжне ім'я групи властивостей `def`.

Зауваження. Для посилання на властивості матеріалу потрібно використовувати його ім'я (наприклад, `mat1`), і дескриптор/тег групи базових властивостей, який зазвичай називається `def`. Наприклад, для використання густини речовини можна використовувати позначення `mat1.def.rho` (або повне ім'я `root.comp1.mat1.def.rho`). Для доступу до цієї величини в межах «фізики» перед її ім'ям використовуйте ідентифікатор «фізики», наприклад, `solid.rho`.

Надрукуємо масу.

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

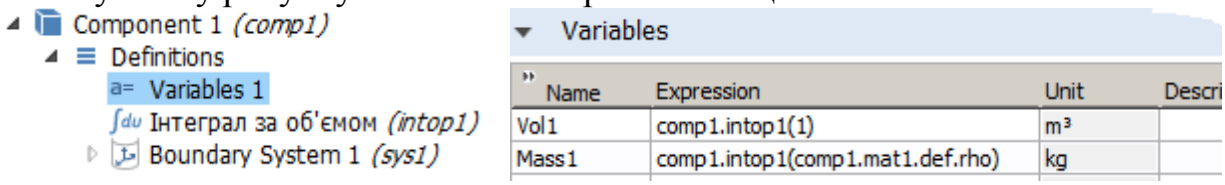
Results - Derived Values - Global Evaluation 3:

`Expression` – **Expression** =`comp1.Mass1`

LMB > Evaluate – Table 1

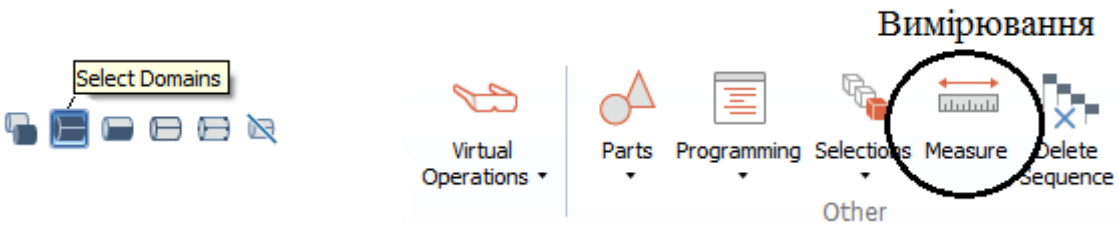
comp1.intop1(1) (m^3)	comp1.Vol1 (m^3)	comp1.Mass1 (kg)
4.7019	4.7019	10814

На наступному рисунку показана створена таблиця локальних змінних.

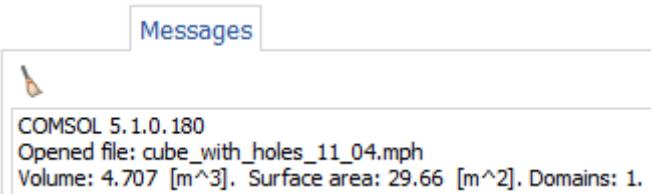


Зауваження. Дізнатися геометричні характеристики фігури (об'єм, площу, довжину, відстань між точками) можна за допомогою інструменту Mesure (вимірювання) панелі `Geometry`. Подивимось, як це робиться.

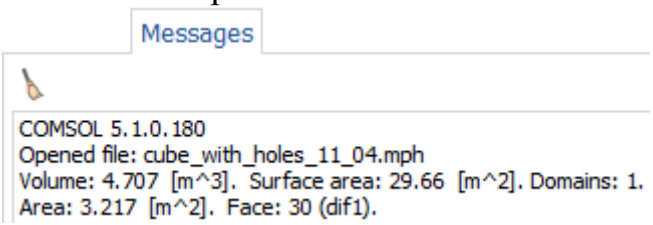
Виділіть в ДМ вузол **Geometry 1**. В графічному вікні встановіть режим вибору областей, натиснувши кнопку Select Domains, що розташована над вікном, і клацніть по кубу. Потім на панелі `Geometry` клацніть по кнопці Mesure.



У вікні повідомлень `Messages`, яке зазвичай розташовано нижче ГВ, з'явиться інформація про об'єм та площу поверхні обраного тіла.



Якщо в ГВ встановити режим вибору границь, натиснувши кнопку Select Boundaries, в ГВ виділити двовимірну область, наприклад, грань куба, та на панелі Geometry клацнути по кнопці Mesure, то у вікно повідомлень буде додано інформацію про площу вказаної поверхні.



Якщо в ГВ встановити режим вибору ребер, натиснувши кнопку Select Edges, потім виділити криві, наприклад, контур дірки на грані куба, та на панелі Geometry клацнути по кнопці Mesure, то у вікно повідомлень буде додано інформацію про сумарну довжину обраних кривих.

Length: 3.141 [m]. Edges: 80, 81, 82, 83 (dif1).

В режимі вибору точок Select Points, можна дізнатися координати виділеної точки, або відстань між двома точками.

Coordinates: (1, 1, 1) [m]. Point: 52 (dif1).
Distance: 2.828427125 [m], (2, 0, 2) [m]. Point 7 (dif1) to 52 (dif1).

Індикатори областей, граней, ребер або точок, використаних в обчисленні, друкуються після результівних даних.

Числові величини з вікна Messages можна скопіювати в буфер обміну і використати, наприклад, у вузлі **Parameters** для визначення параметрів моделі.

Повернемося до роботи зі змінними моделі. Створимо глобальну змінну, що зображає загальну силу Force, яку чинить тіло на основу. Вона дорівнює вазі тіла плюс зовнішня сила, яка діє на тіло зверху

Глобальні змінні створюються аналогічно до локальних. Оскільки змінні Vol1 і Mass1 відносяться до тіла, створеного в компоненті, то ми їх визначили як локальні. Змінна Force характеризує не тільки тіло, а залежить від зовнішнього фактора (тиску на верхню грань), і її логічно визначити як глобальну змінну.

Створимо таблицю глобальних змінних командою

Home **-> Definitions** – Variables – Global Variables

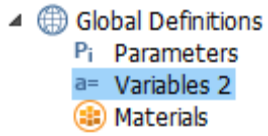
В ПВ створеного вузла **Global Definitions - Variables 2** в першому рядку таблиці змінних введіть:

Variables - Name = Force
- Expression = comp1.Mass1*g_const+F*S

Study 1: RMB > Update Solution

Нагадаємо, що стала g_const позначає прискорення вільного падіння.

Таблиця глобальних змінних набуває вигляду



Variables

Name	Expression	Unit	Descriptio
Force	comp1.Mass1*g_const+F*S	N·m/m	

Надрукуємо значення глобальної змінної `Force`.

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

Results - Derived Values - Global Evaluation 4:

`Expression` – **Expression** = `Force`

LMB > Evaluate – Table 1

comp1.intop1(1) (m^3)	comp1.Vol1 (m^3)	comp1.Mass1 (kg)	Force (N)
4.7019	4.7019	10814	1.3820E5

Окрім змісту, треба враховувати зону дії змінних. Іноді корисно обмежити контекст змінної до певної частини геометрії, в якій вона доступна, наприклад, до певної частини тіла, до поверхні, або лише до деяких ребер і навіть окремих точок.

Припустимо, що нам потрібно побудувати графік напруження σ_z вздовж квадратного контура основи тіла. Це можна зробити декількома способами. Один з них полягає в побудові локальної змінної, яка визначена лише на цьому контурі.

Створимо вузол локальних змінних, зона дії яких буде обмежена зазначеним контуром.

`Home` **`-> Definitions`** – Variables – Local Variables

Component 1 – Definitions – Variables 3:

`Geometric Entity Selection` – **Geometric entity level** = `Edge`

– **Selection** = `Manual; 2,3,15,78` (*ребра основи*)

В першому рядку таблиці змінних введіть:

`Variables` – **Name** = `Sz`

– **Expression** = `solid.sz`

Study 1: RMB > Update Solution

Нам знадобилося побудувати новий вузол **Variables 3** тому, що його змінні визначені на іншому рівні геометрії (на ребрах), в протилежність від змінних вузла **Variables 1**, які визначені по всій моделі.

Побудуємо графік змінної `Sz` над заданим контуром.

`Results->`3D Plot Group

3D Plot Group 12:

`Data`– **Data set** = `Study 1/Solution 1(1)`

`Plot Settings` – **Plot data set edges** = `False`

`3D Plot Group 12` **`-> Add Plot`** – Line

3D Plot Group 12 – Line 1:

`Data`– **Data set** = `From parent`

`Expression` – **Expression** = `comp1.Sz`

`Coloring and Style` – **Line type** = `Tube`

– **Tube radius expression** = `0.02`

– **Radius scale factor** = `1`

Quality – **Recover** = Within domains

RMB > Deformation

3D Plot Group 12 – Line 1 – Deformation 1:

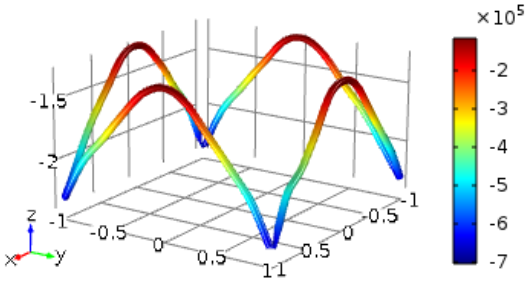
Expression – **X component** = 0

– **Y component** = 0

– **Z component** = comp1.Sz

Scale – **Scale factor** = True, 3e-6

LMB > Plot



Зверніть увагу, що для поліпшення графіка ми включили опцію відновлення похідних **Recover** = Within domains.

Для наочності побудуємо цей графік, «розгорнувши» контур в пряму лінію.

Results->1D Plot Group

1D Plot Group 13:

Data– **Data set** = Study 1/Solution 1(1)

1D Plot Group 13 **->Add Plot** – Line Graph

1D Plot Group 13 – Line Graph 1:

Data– **Data set** = From parent

Selection – **Selection** = Manual; 2,3,15,78 (*ребра основи*)

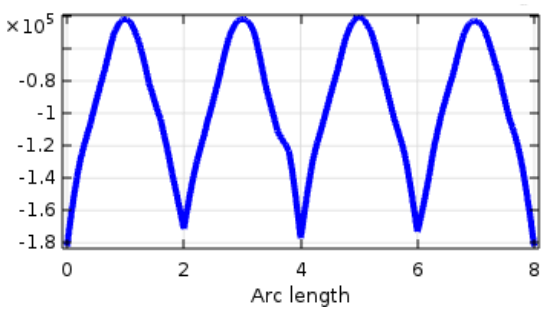
y-Axis Data – **Expression** = comp1.Sz

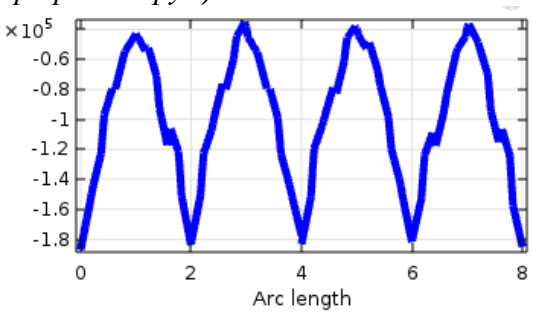
x-Axis Data – **Parameter** = Arc length

Coloring and Style – **Line, Width** = Solid, 3 (*два поля*)

Quality – **Recover** = Within domains

LMB > Plot (*наступний графік ліворуч*)





Якщо вимкнути опцію відновлення похідних **Quality** – **Recover** = Off, то графік буде виглядати так, як зображено на попередньому рисунку праворуч.

Існує також спосіб об’єднання різнойменних змінних, визначених в різних частинах геометрії, під єдиним ім’ям. Для цього слід створити кілька вузлів Variables, зону дії яких обмежити потрібними частинами геометрії. Потім слід задати в цих вузлах змінну з однаковим ім’ям і відповідним виразом для

обчислення. Звернення до такого імені, наприклад, при побудові графіка, буде автоматично використовувати вираз на відповідній частині геометрії.

Параметричний аналіз. Припустимо тепер, що нам потрібно дослідити модель в залежності від якогось параметра. Для цього можна було б змінити значення параметра, обчислити модель, запам'ятати розв'язок, змінити параметр, знову обчислити модель, знову запам'ятати розв'язок і т.д. Це клопітно, тому в системі існують відповідні інструменти аналізу залежності розв'язків від параметрів. Звичайно такі параметри повинні бути створені у вузлі глобальних параметрів (в COMSOL Multiphysics інших параметрів не буває). Потім у вузлі **Study** встановлюються ознаки (або створюється вузол, що містить ознаки) вмикання параметричного аналізу.

Якщо параметр керує геометрією моделі (репрезентує довжину, товщину, радіус, тощо), то в гілці **Study** слід створити вузол **Parametric Sweep**, і в його ПВ вказати параметр і список значень, які потрібно проаналізувати. Якщо параметр використовується в початкових або граничних умовах, являє матеріальний коефіцієнт або коефіцієнт рівняння (тобто не змінює геометрію), то краще використовувати `Auxiliary Sweep` (допоміжний) аналіз, який вмикається встановленням відповідного прапорця в ПВ вузла **Study – Step 1: Stationary**.

Виконаємо в нашій моделі допоміжний параметричний аналіз (`Auxiliary Sweep`) по параметру `F` – тиску на верхню грань. Для цього виконайте інструкції:

Study 1 – Step 1: Stationary:

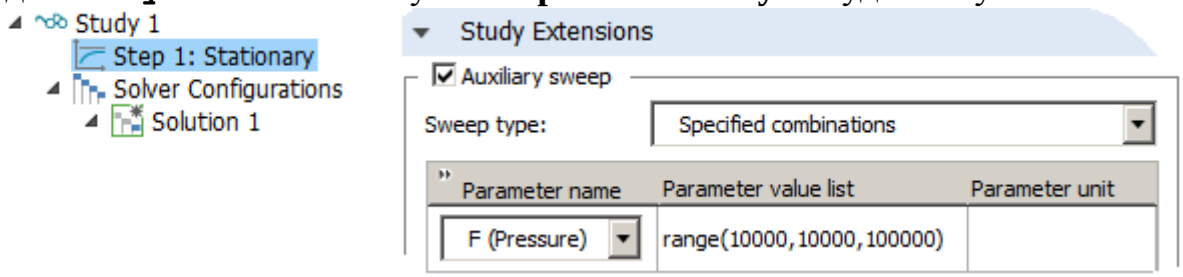
`Study Extensions` – **Auxiliary sweep** = `True`

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 `F`. У другому стовпці таблиці задайте діапазон зміни цього параметра `range(10000,10000,100000)`. Ці дії можна записати у вигляді наступних інструкцій:

`Study Extensions` – **Parameter name** = `F(Pressure)`

– **Parameter value list** = `range(10000,10000,100000)`

Розділ **`Study Extensions`** вузла **Step 1: Stationary** набуде наступного вигляду.



Після цього виконайте обчислення моделі.

`Study`-> Compute

В панелях властивостей майже всіх графічних груп і деяких графіків в розділі **`Data`** з'явиться список **Parameter value (F)** зі значенням параметра `F`, для

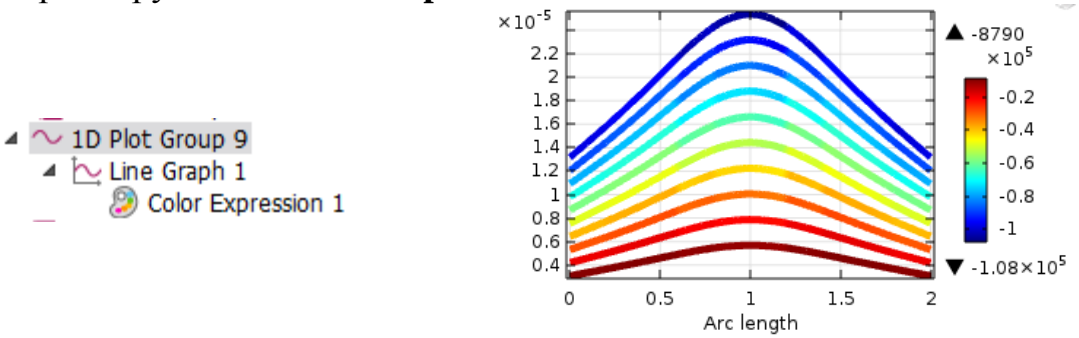
якого побудовано графік. Ви можете обрати будь-яке значення з діапазону `10000,20000,...,100000`, який було задано в таблиці.

Зверніть увагу, що у вузлі **Stress (solid) - Surface 2 - Deformation 1** потрібно встановити масштабний коефіцієнт, бо незважаючи на те, що він повинен успадковуватися з вузла **Surface 1**, це не відбувається.

Stress (solid) - Surface 2 - Deformation 1:

`Scale` – **Scale factor** = `True, 25000`

В одновимірних групах будуються графіки для всіх 10 значень параметра. Ось, наприклад, як будуть виглядати графіки переміщення точок ребра в одновимірній групі **1D Plot Group 9**.



Таким же чином перебудовуються графіки в групі **1D Plot Group 13**.

Пояснимо ще раз різницю між параметричними аналізами `Parametric Sweep` і `Auxiliary Sweep`. Останній зручно використовувати, коли при параметричному аналізі не виконуються геометричні зміни, тобто змінюються параметри зовнішнього впливу, граничні або початкові значення, коефіцієнти рівнянь, тощо. При `Parametric Sweep` аналізі можна змінювати параметри, які керують геометрією (тобто товщину, довжину, діаметр, тощо). Поведінка COMSOL залежить від того, який параметричний аналіз обрано. При `Parametric Sweep` аналізі зміни вносяться по всій моделі: в геометрію, СкЕ сітку і «фізику». Потім виконується обчислення моделі, параметр набуває нового значення і все повторюється. Тобто для кожного нового значення параметра дані у вузлах **Geometry** , **Mesh** і «**фізика**» оновлюються, і повторно виконуються обчислення.

При `Auxiliary Sweep` аналізі вважається, що параметр змінюється неперервно. Зміни вносяться лише в «фізику», і потім виконується обчислення. Результат, отриманий на попередньому кроці розглядається як початкове значення для нової ітерації при новому значенні параметра. У випадку `Auxiliary Sweep` солвер не виконує зміни у вузлах **Geometry** і **Mesh**. Це зменшує час обчислення і покращує точність.

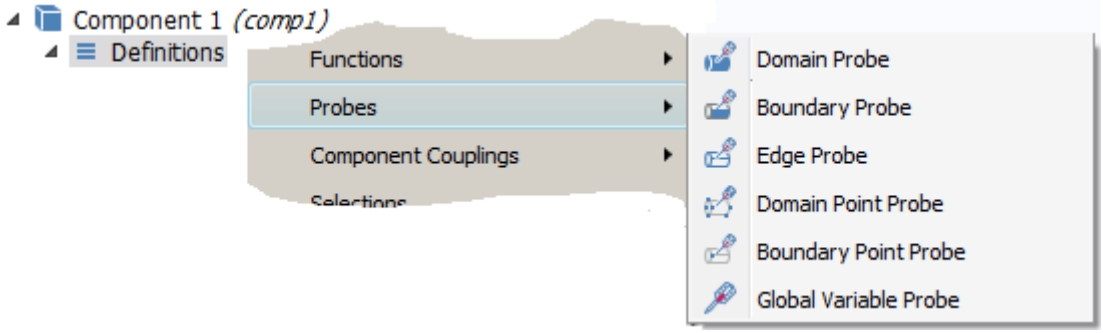
Зауваження. Параметричний аналіз можна виконувати відразу по кільком параметрам.

Зонди (probes). Уявіть собі експеримент, в якому використовуються деякі давачі (або сенсори), що контролюють фізичні параметри в певних точках. При комп'ютерному моделюванні фізичних явищ бажано мати схожі інструменти. В COMSOL Multiphysics вони називаються зондами (probes). Їх первісне призначення – контроль значень полевих змінних в точках під час

обчислювання. Але зонди COMSOL мають більш гнучкі можливості порівняно з давачами/сенсорами, наприклад, можуть обчислювати середнє значення полевої змінної по всьому тілу.

В COMSOL Multiphysics зонд – це скалярна змінна, яка не є функцією просторових координат. Як і фізичний давач, що підключений до зовнішніх інтерфейсів виведення, зонд «друкує» дані в таблицю і по ній будує графік відповідної змінної (в статичних задачах в залежності від параметра, а в динамічних – від часу). Крім того, для свого функціювання він автоматично «створює» власні джерела даних.

Список доступних типів зондів можна побачити, якщо розкрити у випадному меню вузла **Definitions** пункт Probes (зонди).



Перші 3 типи зондів схожі за дією. Це «просторові» зонди (domain probes), «поверхневі» зонди (boundary probes) і «реберні» зонди (edge probes). Вони дозволяють відслідковувати середнє, мінімальне, максимальне або інтегральне значення «полевої» величини по області, поверхні або вздовж ребра.

В розглядуваній моделі створимо `Boundary` зонд, що представляє середнє напруження σ_z на нижній грані.

Component 1 (comp1) – Definitions: RMB > Probes – Boundary Probe

В ПВ створеного вузла **Component 1 – Definitions - Boundary Probe 1 (bnd1)** спорожніть секцію вибору кнопкою Clear Solution і додайте індикатор 3 нижньої грані тіла. Потім введіть

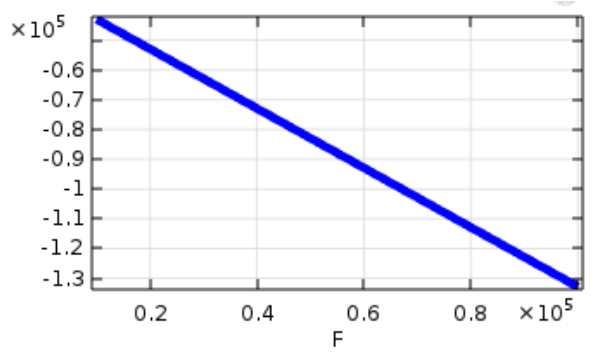
`Probe Type` – **Type** = `Average`

`Expression` – **Expression** = `solid.sz`

Study 1: RMB > Compute

Після виконання команди Compute в гілці **Data Sets** створюються два джерела даних: **Probe Solution 3** і **Boundary Probe 1**. Перше є копією **Study 1/Solution 1 (1)**, а друге – «звуженою» копією з поверхневим рівнем геометрії. В гілці **Derived Values** створюється вузол **Boundary Probe 1** типу `Point Evaluation`. В його списку **Data set** обрано «звужений» однойменний набір даних **Boundary Probe 1**, а поле **Expression** містить вираз `solid.sz`, взятий з такого ж поля зонда. Автоматичне обчислення вузла **Boundary Probe 1** створює таблицю **Probe Table 2** і друкує в ній середні значення напруження `solid.sz` при різних значеннях зовнішнього навантаження (параметра) `F` (наступний рисунок ліворуч).

F	Stress tensor, z co	, Boundary Probe 1
10000	-42991	
20000	-52980	
30000	-62970	
40000	-72959	
50000	-82949	
60000	-92938	
70000	-1.0293E5	
80000	-1.1292E5	
90000	-1.2291E5	
1.0000E5	-1.3290E5	



Також автоматично за цією таблицею створюється одновимірна графічна група **Probe Plot Group 14** і в ній будується графік (типу `Table Graph`) залежності середнього тиску від величини зовнішнього навантаження `F` (попередній рисунок праворуч).

В системі існують 2 типи точкових зондів: `domain point` і `boundary point`, які схожі на давачі, що контролюють фізичний параметр в певній точці. Ці зонди являють змінні, які містять значення «полевих» величин в точках, розташованих в областях або на поверхнях.

Створимо в нашій моделі точковий зонд, який відслідковує вертикальний тиск в нижній вершині куба.

Component 1 (comp1) – Definitions: RMB > Probes – Boundary Point Probe

В ПВ створеного вузла **Component 1 – Definitions - Boundary Point Probe 1** в секцію вибору додайте індикатор 3 нижньої грані тіла (тобто точка буде обиратися на цій грані). Потім введіть

Definitions - Boundary Point Probe 1:

`Point Selection` – **x,y,z** = `-1,-1,-1` (*три поля*)

Definitions - Boundary Point Probe 1 – Point Probe Expression 1 (ppb1):

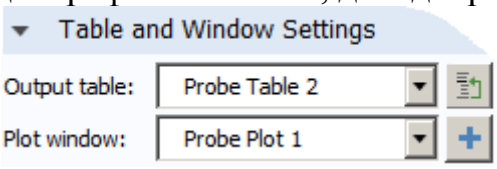
`Expression` – **Expression** = `solid.sz`

Study 1: RMB > Compute

Після виконання команди Compute в гілці **Data Sets** створюється два джерела даних: **Probe Solution 4** (типу `Solution`) і **Boundary Point Probe 1** (типу `Cut Point 3D`) з підвузлом **Point Probe Expression 1(ppb1)**. В гілці **Derived Values** створюється вузол **Point Probe Expression 1** типу `Point Evaluation`. Цей вузол обчислюється автоматично і результат розміщується в попередній таблиці **Probe Table 2**. Також автоматично за цією таблицею в існуючій графічній групі **Probe Plot Group 14** перебудовується графік, на який додається крива, що зображає залежність тиску σ_z в вершині основи від величини зовнішнього навантаження `F` (зелена крива на найближчому наступному графіку).

Перший створений зонд моделі побудував в гілці **Results – Tables** «зондову» таблицю для відображення чисельних результатів. Одночасно з цим була створена одновимірна графічна група з `Probe Table Graph` підвузлом, в якому зображені зондові дані у вигляді лінійного графіка. Всі наступні зонди за замовчуванням друкують свої значення в цю спільну таблицю (в нашому прикладі в **Probe Table 2**), і всі «зондові» криві кресляться на спільному

графіку (в нашому прикладі на графіку **Probe Table Graph 1** графічної групи **Probe Plot Group 14**). Це визначається налаштуваннями в розділ **`Table and Window Settings`** вікон властивостей зондів. В ньому можна налаштувати «адресу» вихідної таблиці і графічного вікна, де відображаються зондові змінні.



Зауваження. В точкових зондах можна створювати необмежену кількість підвузлів `Point Probe Expression`, в кожному з яких можна обчислювати власний вираз (в тій самій точці). Безпосередньо ці підвузли містять поле, в якому задається ім'я зондової змінної, і саме вони являють зондові змінні, а не батьківський вузол. За замовчування у вузлах `Domain Point Probe` і `Boundary Point Probe` завжди створюється один підвузол `Point Probe Expression`.

Для тренування створимо зонд типу `Global Variable Probe`. Такі зонди не використовують геометрію і відносяться до всього компонента. Їм надаються власні імена, і в полі **Expression** вказується вираз, який це ім'я буде заміняти. Отже виконайте наступні інструкції.

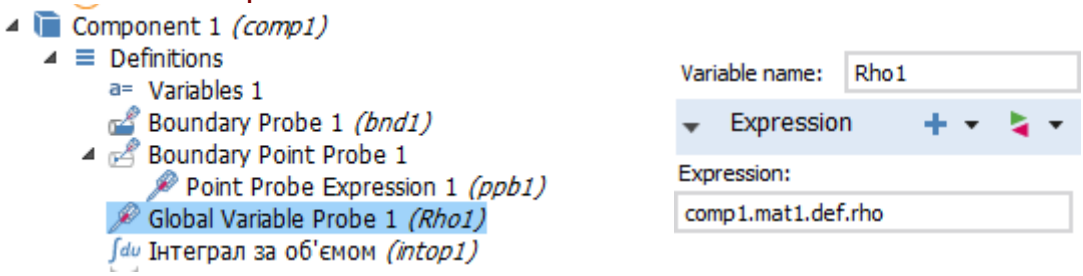
Component 1 (comp1) – Definitions: RMB > Probes – Global Variable Probe

Component 1 (comp1) – Definitions – Global Variable Probe 1:

Variable Name = `Rho1`

`Expression` – **Expression** = `comp1.mat1.def.rho`

Study 1: RMB > Compute



Так само, як і для попередніх зондів, після виконання команди Compute, створюються кілька вузлів ДМ, але нових джерел даних не створюється. Інформація про зонд зберігається в існуючому вузлі ДД **Probe Solution 3**. В гілці **Derived Values** створюється вузол **Global Variable Probe 1**. Він автоматично обчислюється і результат розміщується в попередній таблиці **Probe Table 2**. Також автоматично за цією таблицею в графічній групі **Probe Plot Group 14** перебудовується графік, на який додається крива, що зображає залежність густини від величини зовнішнього навантаження `F`. Звичайно залежності немає і в таблиці в відповідній колонці друкуються однакові значення.

Для зручності скоротимо назви стовпців таблиці **Probe Table 2**. Активуйте ПВ цього вузла і відредагуйте таблицю в розділі **`Column Headers`** так, як

зображено на наступному рисунку ліворуч. Натисніть кнопку Update і отримайте таблицю, яка зображена на наступному рисунку праворуч.

Column Headers

Column	Header
1	F
2	Boundary Probe 1
3	Point Probe 1
4	Global Probe 1

F	Boundary Probe 1	Point Probe 1	Global Probe 1
10000	-42991	-1.8501E5	2300.0
20000	-52980	-2.4389E5	2300.0
30000	-62970	-3.0278E5	2300.0
40000	-72959	-3.6166E5	2300.0
50000	-82949	-4.2055E5	2300.0
60000	-92938	-4.7944E5	2300.0
70000	-1.0293E5	-5.3832E5	2300.0
80000	-1.1292E5	-5.9721E5	2300.0
90000	-1.2291E5	-6.5609E5	2300.0
1.0000E5	-1.3290E5	-7.1498E5	2300.0

Разом з таблицею оновляться графіки в групі **Probe Plot Group 14**. Щоб в групі відобразити легенду, виконайте інструкції

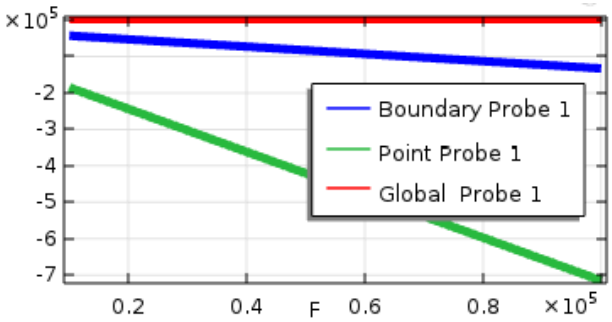
Probe Plot Group 14:

`Legend` – **Show legends, Position** = `True, Middle right`

Probe Plot Group 14 - Probe Table Graph 1:

`Legends` – **Show legends** = `True`

LMB > Plot



Зверніть увагу на те, що графік побудовано в окремому графічному вікні `Probe Plot 1`, а не в звичному вікні `Graphics`. Це видно з ярлика вікна графічної групи **Probe Plot Group 14**.

Підсумовуючи можна сказати, що перші три змінні, які створені з випадного списку доступних типів зондів, відслідковують інтегральні/зональні характеристики величин (інтеграл, середнє, екстремум) по області, поверхні або ребру. Зонди `Domain Point Probe` та `Boundary Point Probe` дозволяють вибрати точку всередині об'єму або на поверхні, і контролювати в ній значення виразу. `Global Variable Probe` зонд обчислює вираз, який не залежить від геометрії або точки.

Якщо ви редагували зонд і внесли до нього зміни, то клацніть по кнопці Update Results вгорі ПВ зонда, щоб оновити налаштування зонда і його дані. Потім клацніть правою кнопкою миші по вузлу **Definitions** і виконайте команду Update Probes, щоб перебудувати вторинні вузли.

Пам'ятайте, що всі зонди мають імена і відрізняються від звичайних змінних лише тим, що після обчислення моделі вони автоматично «створюють» ще кілька вузлів ДМ. Зонди можна використовувати в моделі як будь-які інші змінні, наприклад, в рівняннях, граничних умовах або в умовах зупинки обчислень.

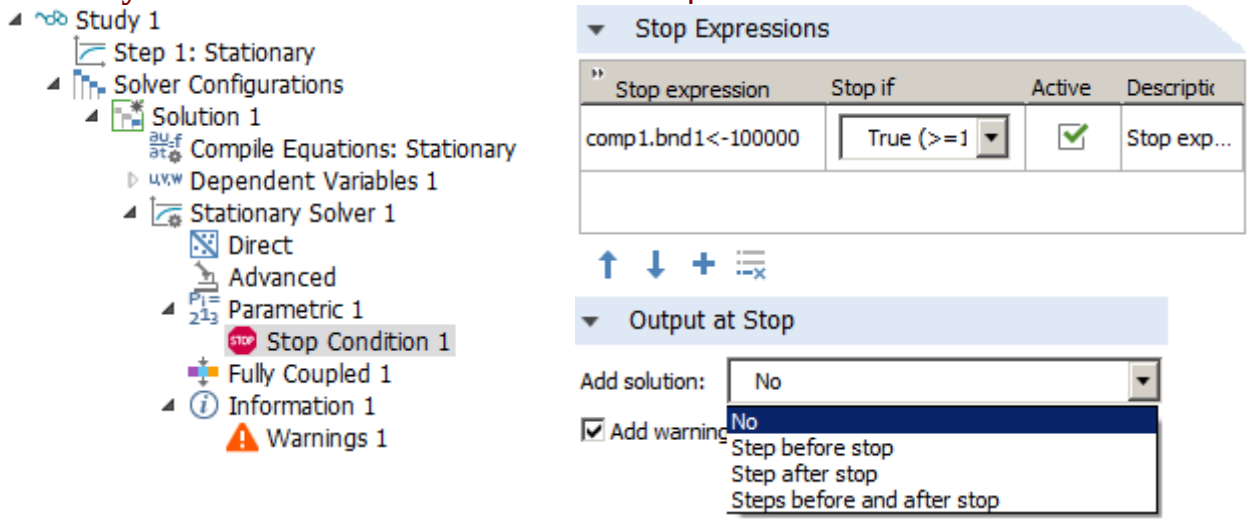
Зауваження. Зондові змінні можна створювати в динамічних задачах. В цьому випадку вони будуть залежати від часу, тобто будуть функціями параметра `t`.

Зупинка обчислень. Крім спостереження за змінними та зондами, ви можете використовувати їх для автоматичної зупинки обчислень. Зупиняти обчислення буває потрібно в різних випадках, наприклад, коли обчислення тривають занадто довго, або коли немає сенсу їх продовжувати (наприклад, виконалася умова, при якій речовина перейшла до фази руйнування). В COMSOL Multiphysics умови зупинки обчислень можна використовувати в трьох випадках: в динамічних задачах (time-dependent), при частотному аналізі (frequency-domain), та в статичних задачах при виконанні параметричного аналізу (parametric або auxiliary sweep).

Щоб передбачити автоматичну зупинку обчислень, по-перше треба мати скалярні змінні, що будуть використовуватися в логічному виразі, який буде постійно перевірятися. Наступним кроком є додавання підвузла **Stop Condition**. В разі параметричного аналізу він додається до вузла **Parametric** в гілці **Solver Configurations – Stationary Solver**. В разі динамічної задачі він додається до вузла **Time-Dependent Solver** в гілці **Solver Configurations**. Останнім кроком є введення в створеному вузлі **Stop Condition** умови зупинки обчислень. Вона є логічним виразом, який може набувати значення `true` або `false`.

Подивимось як можна зупинити обчислення, коли виконується допоміжний параметричний аналіз (Auxiliary Sweep). Нехай зупинка обчислень в нашій задачі відбувається, коли середній тиск на основу перевищує якийсь поріг, наприклад, більше `100000 Н/м`^{2}. Нагадаємо, що середнє значення напруження σ_z по нижній грані тіла зберігається в зондовій змінній `bnd1`. З точністю до знака це є середній тиск на основу.

Розкрийте гілку **Study 1 - Solver Configurations – Solution 1 - Stationary Solver 1 – Parametric** і додайте в неї вузол `Stop Condition` командою **Stationary Solver 1 – Parametric:** RMB > Stop Condition



У вікні властивостей створеного вузла **Stop Condition** натисніть кнопку + Add, розташовану нижче розділу **`Stop Expressions`**, і в першому рядку таблиці в активованому полі стовпця **Stop expression** введіть вираз

`Stop Expressions` - **Stop expression** = `comp1.bnd1<-100000`

В другому стовпці відображається умова, при якій треба робити зупинку. Солвер буде обчислювати введений вираз після кожного кроку зміни параметра і налаштування в стовпці **Stop if** визначають, що зупинка відбувається, коли логічний вираз істинний (`True`, ≥1), або хибний (`Negative`,<0).

Прапорець **Add warning** вказує, чи треба друкувати попередження, коли умова зупинки виконана.

Налаштування в списку **Add solution** визначають, що робити солверу при виконанні умови: залишати результати обчислення всіх «істинних» кроків, або всіх кроків, включно з тим, для якого умова порушилася. Нехай буде вибрана опція

Output at Stop - Add solution = `Step before stop`

Отже в нашій задачі обчислення повинні зупинятися, коли середнє напруження на нижній грані стане менше `-100000 Н/м`^{2}, тобто тиск на основу за модулем стане більше `100000 Н/м`^{2}.

Виконайте команду

`Home` **->** Compute

Почнеться процес обчислень, який зупиниться при значенні параметра `F=60000`.

Щоб дізнатися значення, на якому було зупинено обчислення, треба відкрити ПВ вузла **Warnings 1** (див. попередній рисунок ліворуч). В розділі **`Warnings`** буде надруковано попередження з інформацією, на якому кроці солвер зупинився.

Warnings

Stop condition fulfilled for F=70000 (Stop expression 1).

Цю інформацію також можна знайти у вікні журналу/реєстру (Log). Звичайно, що скоротиться кількість рядків таблиці **Probe Table 2**.

F	Boundary Probe 1	Point Probe 1	Global Probe 1
10000	-42991	-1.8501E5	2300.0
20000	-52980	-2.4389E5	2300.0
30000	-62970	-3.0278E5	2300.0
40000	-72959	-3.6166E5	2300.0
50000	-82949	-4.2055E5	2300.0
60000	-92938	-4.7944E5	2300.0

Зверніть увагу на те, що відредаговані нами назви стовпців після обчислення моделі треба відновлювати.

Зауваження. Вираз, який використовується в умові зупинки (Stop expression) повинен розпізнаватися солвером за будь яких обставин. В ньому потрібно використовувати змінні з глобальним контекстом видимості. Тобто там можуть використовуватися імена глобальних змінних, або повні імена локальних змінних з префіксом – ім'ям компонента.

Залишимо в нашій моделі вузол **Stop Condition**, але деактивуємо його, щоб зупинка обчислень не відбувалася.

Study 1 - Solver Configurations – Solution 1 - Stationary Solver 1 – Parametric - Stop Condition 1: RMB > Disable

Підбір змінних. Коли ми кажемо про змінні задачі, то маємо на увазі не тільки ті, які створені у вузлах **Variables**. Часом нам потрібно мати змінні для визначення яких неможливо записати простий вираз. Їх треба знаходити, розв'язуючи алгебраїчні або диференціальні рівняння.

Поставимо таке питання, яким повинен бути зовнішній тиск на верхню грань, щоб середній тиск на основу дорівнював `100000 Н/м`^{2}. Попередній параметричний аналіз з зупинкою обчислень показав, що він буде розташований в діапазоні від `60000 Н/м`^{2} до `70000 Н/м`^{2}. Але як його знайти точніше.

Спочатку в нашій задачі відключимо параметричний аналіз.

Study 1 – Step 1: Stationary:

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

і виконаємо обчислення командою

`Home` **`->`** Compute

Задача буде розв'язана при зовнішньому тиску `10000 Н/м`^{2}, який визначається початковим значенням параметра `F`. Будуть також обчислені значення всіх зондів, але зараз вони вже не будуть функціями параметра, а будуть сталими величинами. Зондові криві в графічній групі **Probe Plot Group 14** перетворяться на точки.

Зауваження. Зонди можна створювати в статичних задачах без використання параметричного аналізу. В цьому випадку зонди являють сталі величини.

Після переобчислення моделі в таблиці **Probe Table 2** в першій колонці буде надруковано значення `-42991` зондової змінної `bnd1`, яка містить середнє значення напруження σ_z на нижній грані тіла. Різниця між бажаним середнім значенням тиску на основу і реальним середнім значенням дорівнює

$$100000 - |\text{bnd1}| = 100000 - 42991 = 57009 .$$

Нам потрібно зробити її нульовою.

Створимо змінну, яка буде містити шукане значення зовнішнього тиску. Назвемо її `Fout`. Цю, поки що невідому величину (тиск на верхню грань), треба використати в граничній умові у вузлі `Boundary Load`.

Відкрийте ПВ вузла **Component 1 - Solid Mechanics (solid) - Boundary Load 1** і введіть:

`Force` – $\mathbf{F_A}$ = `0,0,-Fout` (*три поля*)

Оскільки зовнішній тиск `Fout` на верхню грань спрямований вниз (проти напряму вісі Z), то потрібно використати знак мінус.

Додамо в нашу фізичну модель рівняння відносно нової змінної `Fout`. Для цього виконайте команду

`Physics` **`-> Global`** – Global Equations

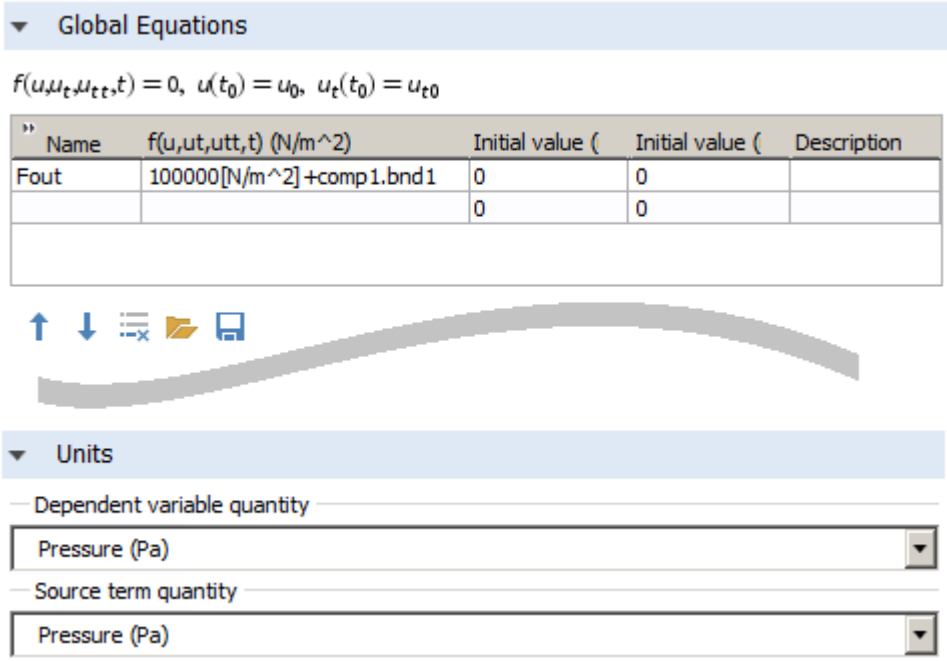
В ПВ створеного вузла **Solid Mechanics (solid) - Global Equations 1** в таблиці рівнянь введіть:

`Global Equations` – **Name** = `Fout`

`Global Equations` – **f(u,ut,utt,t)(1)** = `100000[N/m^2]+comp1.bnd1`

Тут ми використали знак плюс тому, що змінна bnd1 насправді від'ємна.

Щоб введення стало доступним, потрібно клацнути по першому рядку в таблиці рівнянь. Потім в стовпці **Name** треба ввести ім'я змінної, а в другому стовпці – вираз, який повинен обернутися на нуль. Скоріш за все вираз буде виділено помаранчевим кольором. Це означає, що розмірності величин невизначені. Щоб задати розмірності змінної Fout і виразу f(u,ut,utt,t) заповніть розділ **Units** так, як показано на наступному рисунку.



Розмірність залежної (шуканої) змінної задається в полі **Units-Dependent variable quantity**, а розмірність виразу f(u,ut,utt,t) – в полі **Units-Source term quantity**.

В результаті ми додали в задачу рівняння $100000-|\mathrm{bnd1}(\mathrm{Fout})|=0$ і додаткову невідому Fout. Тепер виконайте команду

Home -> Compute

Після обчислення в першому стовпці таблиці **Probe Table 2** буде стояти значення -1.0000E5 зондової змінної bnd1. Тобто середній тиск на основу дорівнюватиме 100000[N/m^2].

Щоб надрукувати значення шуканої невідомої Fout, виконайте інструкції:

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

Results - Derived Values - Global Evaluation 6:

Expression – **Expression** = comp1.Fout

LMB > Evaluate – New Table

Буде створена таблиця **Table 3** зі значенням змінної Fout=67070. Якщо це число використати в граничній умові для зовнішнього навантаження, то середній тиск на основу дорівнюватиме бажаному 100000 Н/м^2. По-перше, це зрозуміло зі значення зондової змінної bnd1, яке надруковано в таблиці **Probe Table 2**. По-друге, це можна перевірити, якщо зробити обчислення задачі при зовнішньому тиску на верхню грань 67070 Н/м^2.

Деактивуйте вузли **Derived Values - Global Evaluation 6** та **Solid Mechanics (solid) - Global Equations 1**. А в ПВ вузла **Solid Mechanics (solid) - Boundary Load 1** введіть

Force – $\mathbf{F_A}$ = 0,0,- 67070 (*три поля*)

Переобчисліть задачу командою

Study 1: RMB > Compute

Потім виконайте команду

Derived Values - Boundary Probe 1: LMB > Evaluate – Table 3

В другому стовпці таблиці **Table 3** з'явиться бажане значення -1.0000E5 середнього напруження σ_z на нижній грані тіла. За модулем воно співпадає з середнім тиском на основу.

State variable Fout (Pa)	Stress tensor, z component (N/m^2), Boundary Prob
67070	-1.0000E5

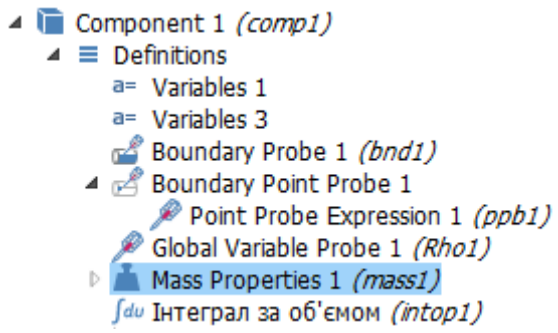
Тепер можете повернути модель в попередній стан, виконавши скасування останніх інструкцій деактивації, і повернувши в граничній умові вузла **Boundary Load 1** навантаження Fout.

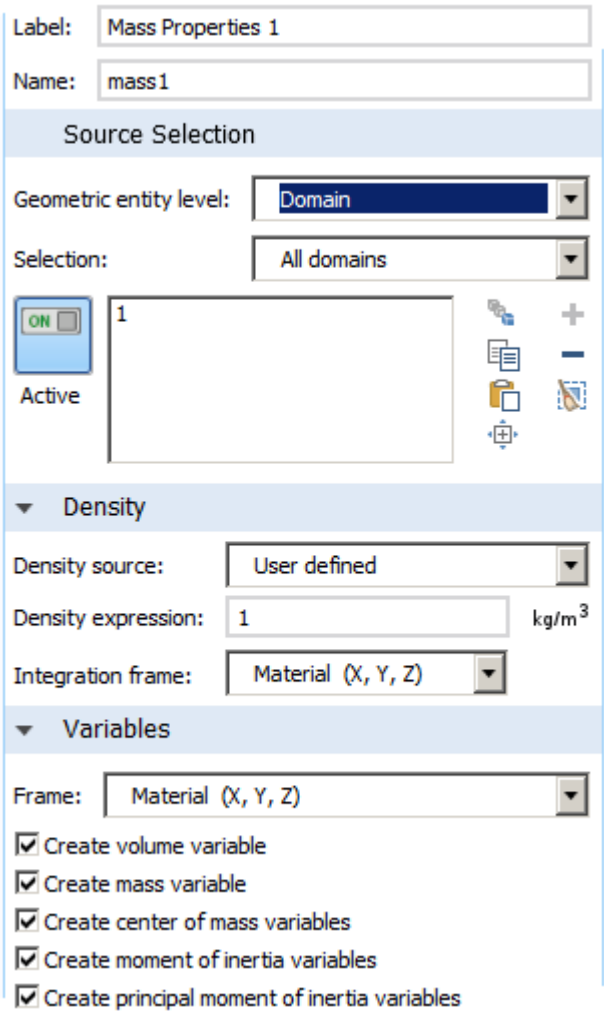
Змінні маси (**mass properties**).

В цьому параграфі ми створили багато змінних, які зберігали такі розповсюджені властивості тіла, як об'єм та маса. Ці величини потрібні досить часто, і розробники системи це передбачили. Щоб створити такі змінні достатньо виконати команду

Definitions -> **Mass properties** – Mass properties

В гілці **Component – Definitions** створюється вузол **Mass Properties 1 (mass1)**, який буде представляти групу механічних і геометричних властивостей компонента. В ПВ створеного вузла в розділі **Source Selection** вказується рівень геометрії (Domain, Boundary, Edge або Point) і в секції вибору – множина, для якої будуть виконуватися обчислення. Для пошуку маси є розділ, в якому задається густина речовини. За наявності цього вузла 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)**.





Змінні маси стають доступними після виконання або оновлення обчислень.

Study 1: RMB > Update Solution

В результаті створюються наступні змінні:

Величина	**Ім'я змінної за замовчуванням**
Об'єм	`mass1.volume`
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`

Для тренування надрукуємо об'єм нашого тіла, обчисливши його за допомогою змінної `mass1.volume`.

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

Results - Derived Values - Global Evaluation 7:

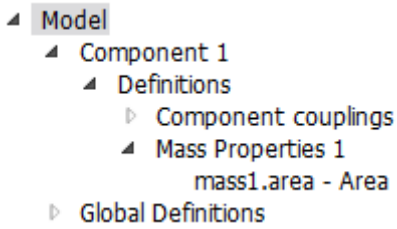
`Expression` – **Expression** = `comp1.mass1.volume`

LMB> Evaluate – Table 1

comp1.intop1(1) (m^3)	comp1.Vol1 (m^3)	comp1.Mass1 (kg)	Force (N)	Volume (m^3)
4.7019	4.7019	10814	1.3820E5	4.7019

В перших комірках таблиці **Table 1** вже було надруковано значення об'єму і воно співпадає з тим, що ми зараз отримали.

Зауваження 1. В компоненті може бути кілька вузлів `Mass Properties`.
Зауваження 2. В двовимірному компоненті для визначення площі фігури застосовується змінна `mass1.area`.



Підведемо короткий підсумок. В локальних вузлах **Definitions** створюються конструкції, які допомагають працювати з компонентом: змінні, функції, оператори, зонди, маси. Всі ці об'єкти мають імена, і звернення до них виконується з використанням точкової нотації. Щоб прийняти додані визначення до уваги потрібно оновити обчислення.

Скалярні змінні можна додавати в модель як невідомі алгебраїчних або звичайних диференціальних рівнянь, які формуються у вузлах `Global Equations`.

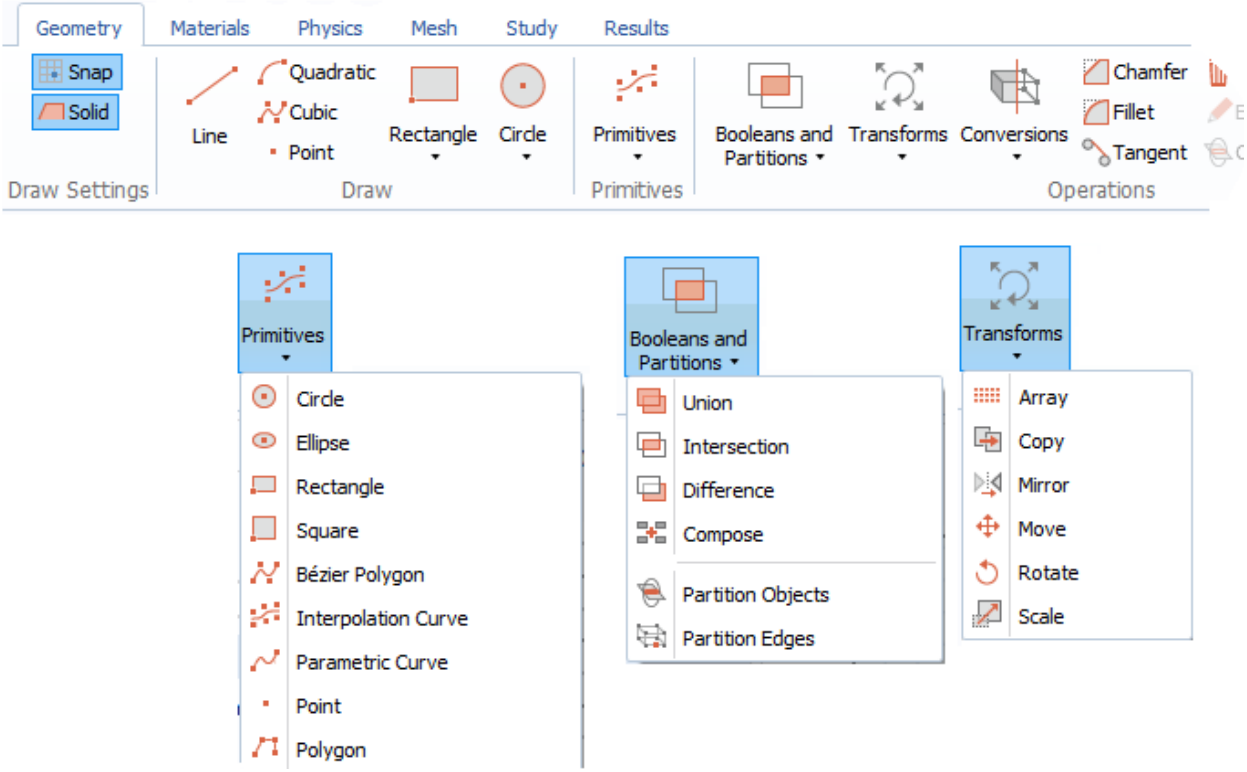
1.5.5. Побудова геометрії.

Побудова розрахункової області, яку ми часто називаємо геометрією, є одним з перших кроків моделювання в COMSOL Multiphysics. В системі існує багато інструментів та операцій, призначених для цього.

Головними кроками створення геометрії є:

- креслення геометричних фігур (їх набір залежить від просторової розмірності);
- використання різноманітних операцій (булевих, перетворення та ін.) для створення нових фігур з вже побудованих;
- визначення способу використання сполучених геометричних об’єктів за допомогою завершальної операції Form Union або Form Assembly.

Всі інструменти, які призначені для створення геометрії, зосереджені на стрічці `Geometry` (панель креслення). Для двовимірних моделей основна частина стрічки `Geometry` показана на наступному рисунку. Там же зображено вміст трьох найбільш часто використовуваних груп команд (меню): **`Primitives`**, **`Booleans and Partitions`** та **`Transforms`**. З можливостями деяких з них ви вже познайомилися.



Основою конструювання геометрії є «солідні» (solid) об'єкти. Цей термін буде використовуватися стосовно фігур на площині, що мають скінченну площу, і до тіл в просторі, які мають скінченний об’єм. Солідні об'єкти складаються з граничної і внутрішньої частини і повністю визначаються в термінах власних границь. Іноді солідний об'єкт має внутрішні межі, що розділяють його підобласті, в яких можуть бути задані різні матеріальні властивості або СкЕ сітка.

Найпростіший спосіб конструювання геометрії складається в виборі об’єкта з вбудованого списку геометричних фігур, які зазвичай називають примітивами. При виборі команди з групи примітивів (**`Primitives`**) система створює вузол ДМ і додає креслення фігури в геометрію. Після цього в ПВ створеного вузла можна налаштувати розміри і положення фігури.

Інший спосіб створення геометричних об’єктів, доступний в 2D і 1D моделях, полягає в кресленні примітивів за допомогою миші. Відповідні інструкції розташовані в групі команд **`Draw`**. Вони виконуються наступним чином:

- клацніть по кнопці фігури, яку слід створити;
- натисніть ліву кнопку миші в графічному вікні для визначення центра або кута фігури;
- протягніть мишу (з натиснутою кнопкою), задавши бажаний розмір, і відпустіть мишу; фігура негайно з’явиться в графічному вікні, а її вузол в дереві моделі;
- якщо треба, відредагуйте властивості фігури у ПВ створеного вузла.

Після побудови кількох фігур можна почати комбінувати їх багатьма способами, використовуючи геометричні операції. Булеві (boolean) операції об’єднання (union), перетину (intersection) та різниці (difference),

використовуються для комбінування фігур, створюючи з них нові більш складні об’єкти. Операції перетворення (transformation) - Move, Copy, Mirror та Array дозволяють змінити положення (Move) фігури, створити її копію (Copy), дзеркально відобразити відносно точки, лінії чи площини (Mirror), або розмножити її копії у вигляді масиву (Array).

В 3D моделях ви не можете креслити геометричні фігури мишею, але можете створювати поперечні перерізи на робочих площинах, які потім можете поширювати в 3D об’єкти за допомогою операцій перетворення:

- операція витягування (Extrude) дозволяє поширити об’єкт, що створений в робочій площині або на плоскій грані, в тривимірне тіло;
- операція обертання (Revolve) навколо вісі створює з 2D фігури тривимірне тіло;
- операція розгортання (Sweep) виконує перенесення з замітанням 2D фігури вздовж криволінійного шляху, створюючи тривимірний об’єкт.

Інший підхід полягає в побудові межі тривимірного тіла. В цьому випадку ви явно конструюєте замкнуту поверхню твердого тіла, прорисовуючи кожну його грань. Потім така поверхня перетворюється в тверде тіло. Цей спосіб називається граничним (або поверхневим) моделюванням. Граничне моделювання неявно використовується при двовимірному кресленні, коли включений Solid режим. Якщо на інструментальні панелі `Geometry` (в 2D режимі) натиснута кнопка Solid Solid, то при кресленні на площині замкнутих кривих утворюються двовимірні солідні об’єкти.

Звісно, що розміри та положення фігур можна задавати, використовуючи глобальні параметри.

Всі дії по створенню геометрії відображаються в гілці **Geometry** дерева моделей, яка фактично містить список інструкцій креслення розрахункової області компонента. При конструюванні геометрії система виконує побудови з цього списку в послідовності зверху вниз. Отже крім створення елементів геометрії в гілці **Geometry** визначається послідовність їх побудови і комбінування. Безпосередньо ПВ самого вузла **Geometry** містіть налаштування одиниць вимірювання довжин та кутів.

При роботі з геометричною гілкою дерева моделі можна використовувати стандартні операції роботи з буфером: копіювання і вставку вузлів-інструкцій. Але скопійований об'єкт може бути вставлений тільки в геометричну послідовність ідентичної розмірності.

Останній вузол **Form Union/Assembly** гілки **Geometry** визначає завершальний спосіб формування геометрії.

- За замовчуванням використовується створення об'єднання (union). Тоді система виконує логічне об'єднання всіх об'єктів послідовності, тобто створює суцільне тіло. Інколи воно може бути поділене на області, відокремлені внутрішніми межами, які призначені, наприклад, для задання різних матеріальних властивостей об’єднаних частин.
- Альтернативою є збірка (assembly). Тоді система інтерпретує геометрію як колекцію об'єктів (замість їх об'єднання). Це означає, що користувач сам буде визначити способи зв’язку розрізнених тіл за допомогою

інструментів, які називаються «парами» (pairs). Пари, наприклад, будуть визначати межі, де шукане поле неперервно. За замовчуванням пари створюються автоматично при формуванні збірки.

В гілці **Geometry** після вузла **Form Union/Assembly** можуть бути розташовані лише вузли віртуальних операцій, які використовують існуюче геометричне представлення як базове для конструювання альтернативної геометрії виключно до цілей дискретизації і постановки фізичної задачі.

Кнопка Build All в вгорі ПВ геометричного вузла будує всі об'єкти поточної геометрії.

Приклад 1. Побудуємо тіло, яке зображено на наступному рисунку.

Створіть порожню модель.

File->New->Blank Model

Додайте в модель тривимірний компонент.

Untitled.mph (root) : RMB > Add Component ->3D

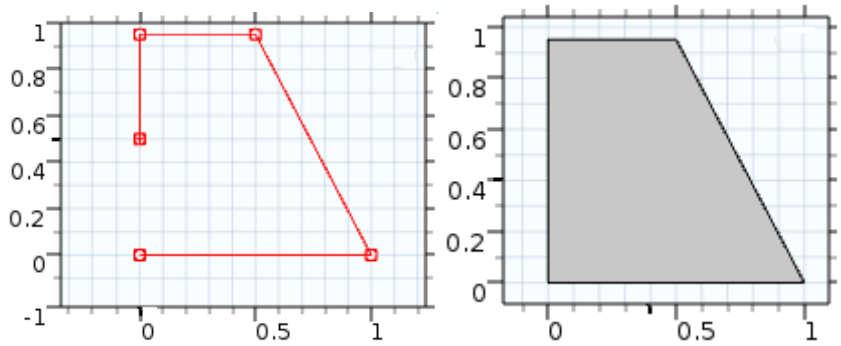
З'явиться гілка **Component 1** з вузлом **Geometry 1**. Креслення фігури в цьому вузлі складається з кількох кроків.

1. *Створення робочої площини*. При активному вузлі **Geometry 1** виконайте команду

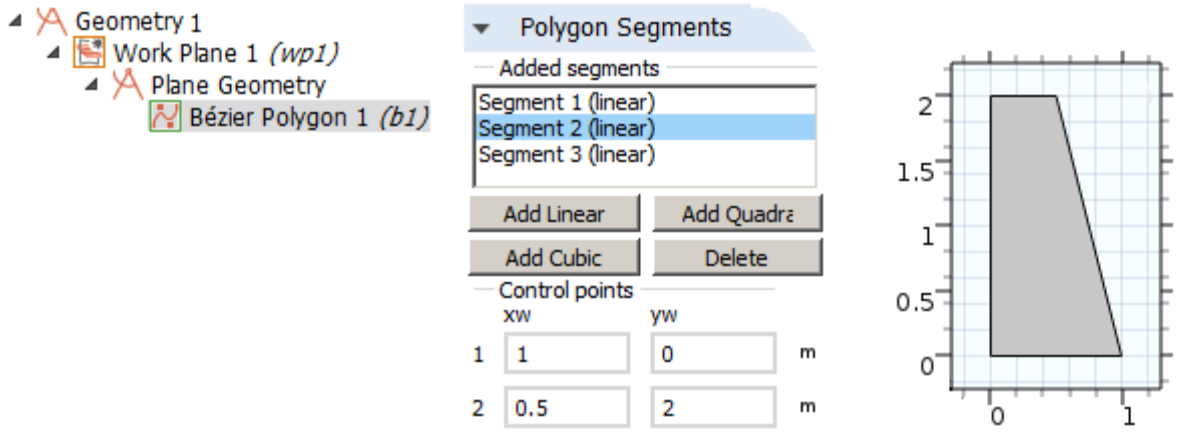
Geometry->**Work Plane**->Work Plane

В ПВ створеного вузла **Work Plane 1** всі параметри залиште без змін. Вони відповідають тому, що робоча площина буде співпадати з площиною XY. Клацніть по кнопці Show Work Plane вгорі панелі. Цим ви активуєте вузол **Work Plane 1 – Plane Geometry**.

2. *Креслення фігури в робочій площині*. На закладці Work Plane інструментальної панелі перевірте, що натиснуті кнопки Snap і Solid. Кнопка Snap включає режим «притягання» миші до вершин вірівнювальної решітки (grid). На панелі в секції **Draw** натисніть кнопку Line, переведіть покажчик миші в графічну область і намалюйте багатокутник, показаний на наступному рисунку праворуч. Закінчіть креслення ламаної (коли вона буде вже замкненою) клацанням правою кнопкою миші в зоні малюнка. Крива автоматично перетвориться на солідний чотирикутник.



Відредагуйте координати його вершин так, щоб вони дорівнювали (0,0), (1,0), (0.5,2), (0,2). Для цього в розділі **`Polygon Segments`** послідовно виділяйте сегменти ламаної і редагуйте значення координат вершин в полях **Control points**.



Зверніть увагу на те, що вузол ДМ отримав назву **Bézier Polygon 1 (b1).**

Додайте до багатокутника фаску і скруглення. Для цього виконайте команди

`Work Plane->`**`Operations`**`->` Chamfer

Work Plane 1 (wp1) - Plane Geometry - Chamfer 1 (cha1):

`Points` – **Vertices to chamfer** = `b1:4` (*права нижня вершина*)

`Distance` – **Distance from vertex** = `0.25` (*розмір фаски*)

RMB > Build Selected (*наступний рисунок ліворуч*)

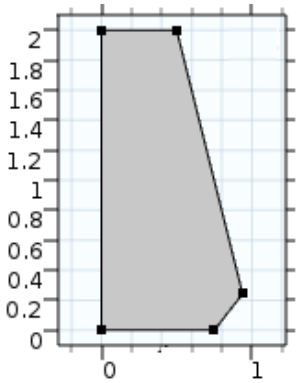
`Work Plane->`**`Operations`**`->` Fillet

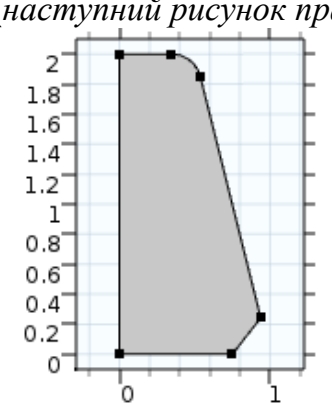
Work Plane 1 (wp1) - Plane Geometry - Fillet 1 (fil1):

`Points` – **Vertices to fillet** = `cha1:3` (*права верхня вершина*)

`Radius` – **Radius** = `0.2` (*радіус скруглення*)

RMB > Build Selected (*наступний рисунок праворуч*)





Створіть профіль ребер. Для цього побудуйте прямокутник, а з нього – масив прямокутників. Потім цей масив видаліть з вже побудованої фігури.

Work Plane->**Primitives**-> Rectangle

Work Plane 1 (wp1) - Plane Geometry - Rectangle 1 (r1):

Size – **Width, Height** = 0.5,0.25 (*два поля; розміри*)

Position – **xw, yw** = 0.5,0.4 (*два поля; положення кута*)

RMB > Build Selected (*наступний рисунок ліворуч*)

Work Plane->**Transforms**-> Array

Work Plane 1 (wp1) - Plane Geometry - Array 1 (arr1):

Input – **Input objects** = r1 (*фігура для розмноження*)

Size – **xw size, yw size** = 1,3 (*кількість стовпців і рядків масиву*)

Displacement – **xw, yw** = 0,0.5 (*вектор зміщення елементів масиву*)

RMB > Build Selected (*наступний рисунок всередині*)

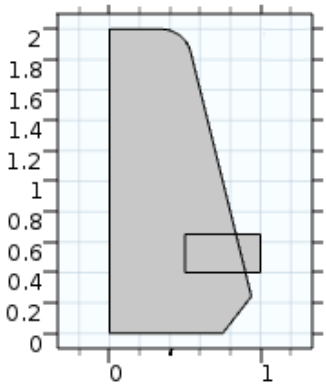
Work Plane -> **Booleans and Partitions** -> Difference

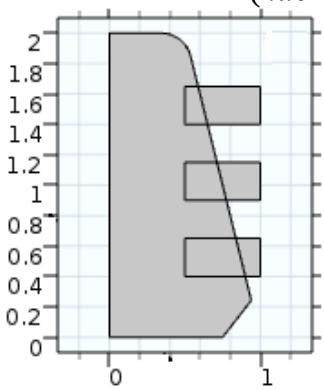
Work Plane 1 (wp1) - Plane Geometry - Difference 1 (dif1):

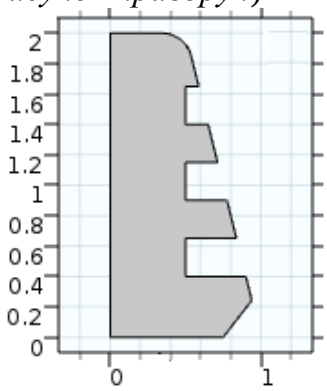
Difference – **Objects to add** = fil1

Difference – **Objects to subtract** =arr1(1,1),arr1(1,2),arr1(1,3)

RMB > Build Selected (*наступний рисунок праворуч*)







Створіть профіль отвору.

Work Plane->**Primitives**-> Rectangle

Work Plane 1 (wp1) - Plane Geometry - Rectangle 2 (r2):

Size – **Width, Height** = 0.6,1.5 (*два поля; розміри*)

Position – **xw, yw** = -0.3,0 (*два поля; положення кута*)

Work Plane->**Primitives**-> Circle

Work Plane 1 (wp1) - Plane Geometry – Circle 1 **(c1):**

Size and Shape – **Radius** = 0.3

Position – **Base, xw, yw** = Center;0,1.5 (*три поля*)

RMB > Build Selected (*наступний рисунок ліворуч*)

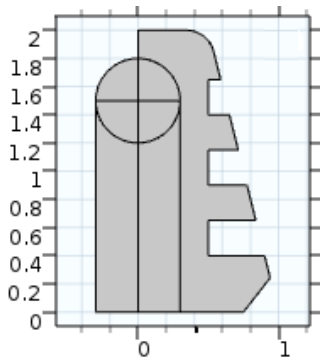
Work Plane->**Booleans and Partitions**-> Difference

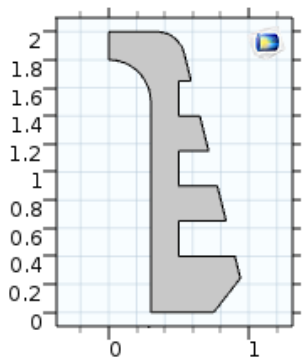
Work Plane 1 (wp1) - Plane Geometry - Difference 2 (dif2):

Difference – **Objects to add** = dif11

Difference – **Objects to subtract** =r2,c1

RMB > Build Selected (*наступний рисунок праворуч*)

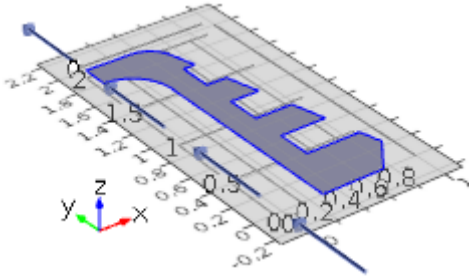




3. *Створення тіла обертання*. Операція Revolve призначена для утворення тіла обертання з профілю, який побудований в робочій площині. Для неї додатково треба вказати орієнтацію вісі обертання.

Work Plane 1 (wp1):

`Geometry` ->**`Operations`**-> `Revolve`



Geometry 1 – Revolve 1 (rev1):

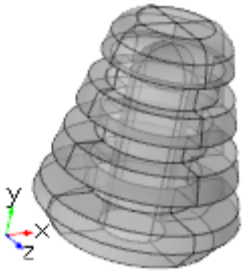
`General` – **Input objects** = `wp1` (*робоча площина*)

Задання в робочій площині вісі обертання за допомогою точки та вектора (двовимірного) напрямку.

`Revolution Axis` – **Points on the revolution axis – xw, yw** = `0,0`

`Revolution Axis` – **Direction of revolution axis – xw, yw** = `0,1`

RMB > Build Selected (*наступний рисунок повернуто*)



Зауваження. Вісь обертання не повинна пересікати контур фігури, що обертається (торкатися може).

4. *Надбудова тривимірних елементів*. Добудуємо зверху до тіла прямокутну призму. Для цього верхню грань побудованого тіла використаємо в якості нової робочої площини. На ній накреслимо квадрат, який «витягнемо» вгору для утворення призми.

`Geometry` -> **`Work Plane`** -> `Work Plane`

Work Plane 2 (wp2):

`Plane Definition` – **Plane type** = `Quick` (*спосіб вибору площини*)

`Plane Definition` – **Plane** = `zx-plane`

Plane Definition – **Offset type, y-coordinate** = Distance; 2

(*наступний рисунок ліворуч*)

Work Plane 2 (wp2) - Plane Geometry:

Work Plane->**Primitives**-> Square

Work Plane 2 (wp2) - Plane Geometry - Square 1 (sq1):

Size – **Side length** = 0.3

Position – **Base, xw, yw** = Center;0,0 (*три поля*)

RMB > Build Selected (*наступний рисунок всередині*)

Операція Extrude призначена для витягування профілю робочої площини в простір. Зміщення профілю за замовчуванням виконується в напрямку нормальному до площини. Додатково слід вказати відстань, на яку витягується профіль.

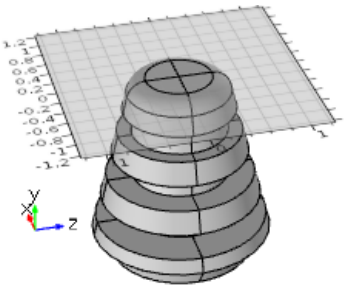
Work Plane 2 (wp2):> Geometry -> **Operations**-> Extrude

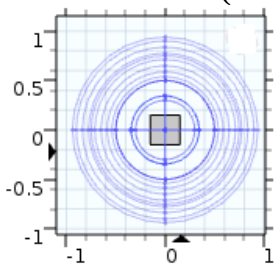
Geometry 1 – Extrude 1 (ext1):

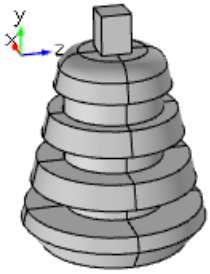
General – **Input objects** = wp2 (*робоча площина*)

Distance from Plane – **Distance** = 0.4 (*відстань витягування*)

RMB > Build Selected (*наступний рисунок праворуч*)







Створимо зверху кільце, яке проходить крізь призму. Кільце побудуємо за допомогою операції Sweep (розгортання). Вона переносить 2D фігуру вздовж шляху (кривої) в просторі, замітаючи тривимірну область і утворюючи 3D тіло. Для цього нам потрібно побудувати 2D фігуру (коло) і криву, вздовж якої буде виконуватися перенесення. Коло накреслимо в робочій площині XY, а криву – в площині YZ. Звісно, що ці робочі площини теж треба створити.

Побудуйте коло.

Geometry->**Work Plane**->Work Plane

Work Plane 3 (wp3):

Plane Definition – **Plane** = xy-plane

Plane Definition – **Offset type, y-coordinate** = Distance; 0

Work Plane 3 (wp3) - Plane Geometry:

Work Plane->**Primitives**-> Circle

Work Plane 1 (wp1) - Plane Geometry - Circle 1 (c1):

Size and Shape – **Radius** = 0.1

Position – **Base, xw, yw** = Center; 0, 2.2 (*три поля*)

RMB > Build Selected (*наступний рисунок ліворуч*)

Створіть криву перенесення.

Geometry 1: Geometry->**Work Plane**->Work Plane

Work Plane 4 (wp4):

Plane Definition – **Plane** = yz-plane

Plane Definition – **Offset type, y-coordinate** = Distance; 0

Work Plane 4 (wp4) - Plane Geometry:

Work Plane->**Primitives**-> Circle

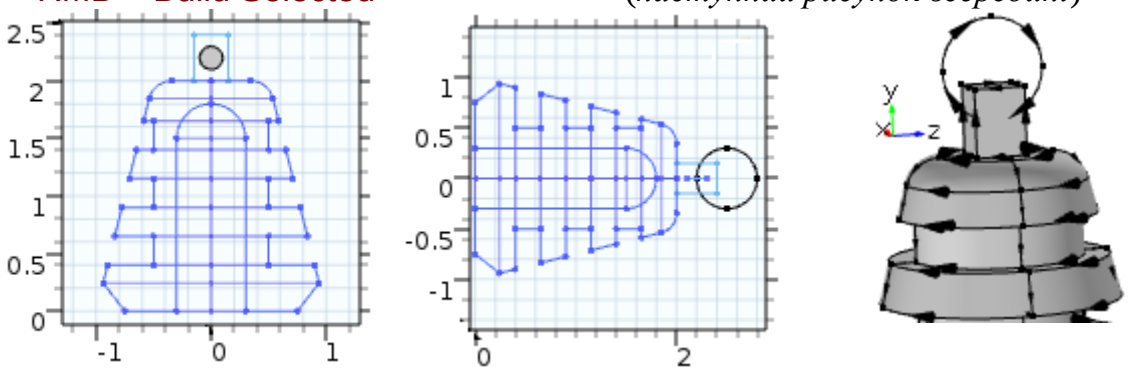
Work Plane 4 (wp4) - Plane Geometry - Circle 1 (c1):

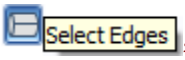
Object Type – **Type** = Curve (*створюємо окружність, а не коло*)

Size and Shape – **Radius** = 0.3

Position – **Base, xw, yw** = Center; 2.5, 0 (*три поля*)

RMB > Build Selected (*наступний рисунок всередині*)



Розгортання неможна виконати по замкненій кривій. Тому для побудови кільця створимо дві розгортки (sweep) по половинах окружності в різні боки. Слід мати на увазі, що перенесення виконується вздовж напрямку зростання параметра кривої. Його можна наочно побачити, якщо у вузлі **Component 1 (comp1)** – **View 1** включити опцію **Show edge direction arrows**, а в графічному вікні кнопкою 

, розташованою над цим вікном, включити режим Select Edges.

Work Plane 4 (wp4): RMB > Build Selected (*попередній рисунок праворуч*)

Створіть розгортки.

Geometry ->**Operations**-> Sweep

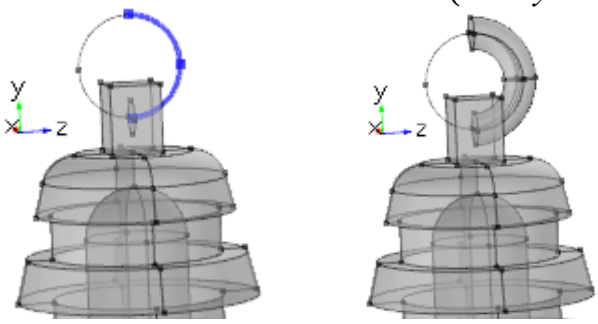
Geometry 1 – Sweep 1 (swe1):

Cross Section – **Faces to sweep** = wp3:1(*площина з круговим перерізом*)

Spline Curve – **Edges to follow**= wp4:3,4 (*наступний рисунок ліворуч*)

Spline Curve – **Reverse direction**= True (*в зворотному напрямку*)

RMB > Build Selected (*наступний рисунок праворуч*)



Зверніть увагу на опцію **Reverse direction** = True, яка включає перенесення кола в оберненому напрямку вздовж кривої (інакше перенесення виконується не в той бік).

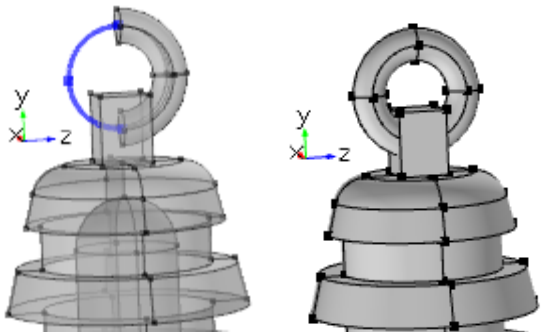
`Geometry -> Operations ->` Sweep

Geometry 1 – Sweep 2 (swe2):

`Cross Section` – Faces to sweep = `wp3:1`

`Spline Curve` – Edges to follow= `wp4:1,2` (*наступний рисунок ліворуч*)

RMB > Build Selected (*наступний рисунок праворуч*)

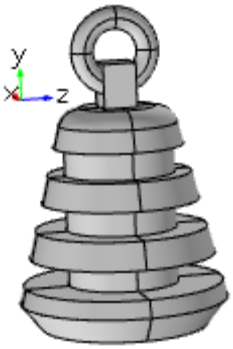


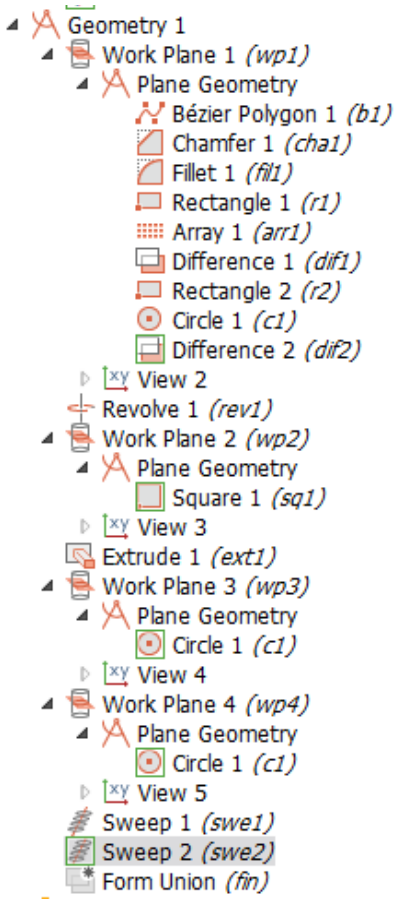
5. *Визначення завершального способу формування геометрії.* Останнім вузлом в гілці **Geometry 1** залиште **Form Union (fin)**.

RMB > Build All Objects (*наступний рисунок ліворуч*)

Коли ви побажаєте розглядати кільце як рухому частину конструкції, наприклад, в задачі динаміки тіл (Multibody Dynamics), то в ПВ вузла **Form Union** в списку **Action** слід обрати значення `Form an assembly`. Але тоді в верхній призмі потрібно буде ще спроектувати отвір під кільце.

Якщо ви виконували інструкції в послідовності з нашим прикладом, то гілка **Geometry 1** ДМ буде мати вигляд, зображений на наступному рисунку праворуч.



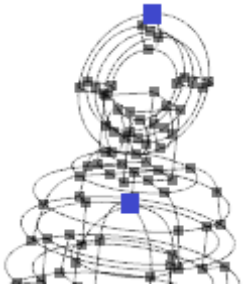


Зауваження. Кільце можна було б створити з використанням тривимірного графічного примітива `Torus` (тор).

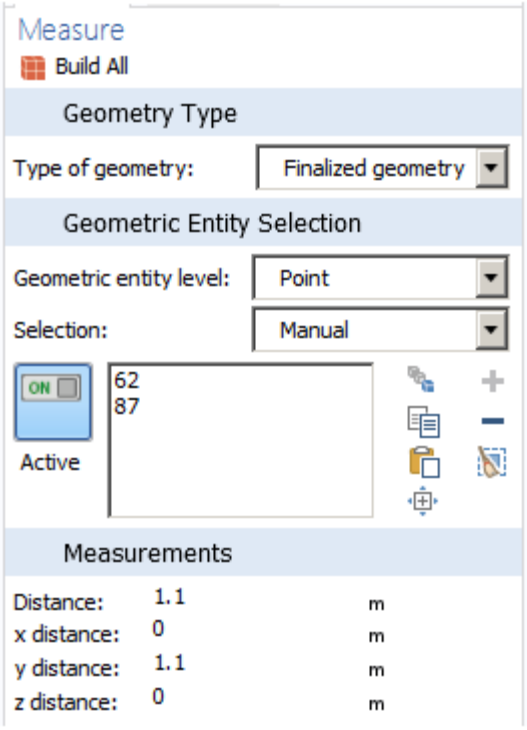
Тут доречно продемонструвати ще один спосіб дізнавання геометричних характеристик тіл. Команда

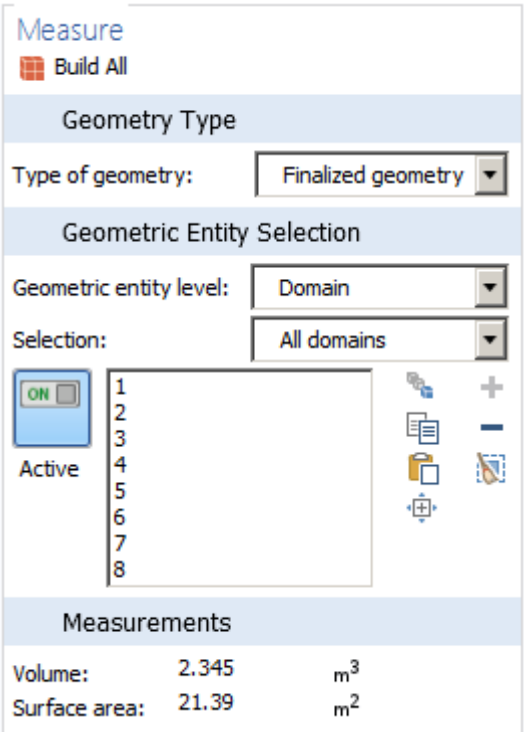
Geometry 1: RMB > Measure

відкриває панель `Measure` вимірювань в геометрії моделі.. В полі **Geeometric entity level** обирається рівень геометрії, в секції вибору вказуються елементи. Тоді внизу панелі в розділі **`Measurements`** автоматично з'являються результати. Якщо рівнем геометрії обрати `Point`, і в секції вибору вказати пару точок, то буде обчислюватися відстань між ними. Для прикладу було обрано верхівку кільця (точка з індикатором 62) і верхівку внутрішнього отвору фігури (точка 87).



В результаті панель вимірювань `Measure` набула вигляду, зображеного на наступному рисунку ліворуч. Внизу панелі надрукована відстань (Distance) `1.1 m` між обраними точками. У разі вибору геометричного рівня `Domain`, обчислюються об'єм `2.345` m^3 і площа поверхні `21.39` m^2 (наступний рисунок праворуч) обраного тіла.

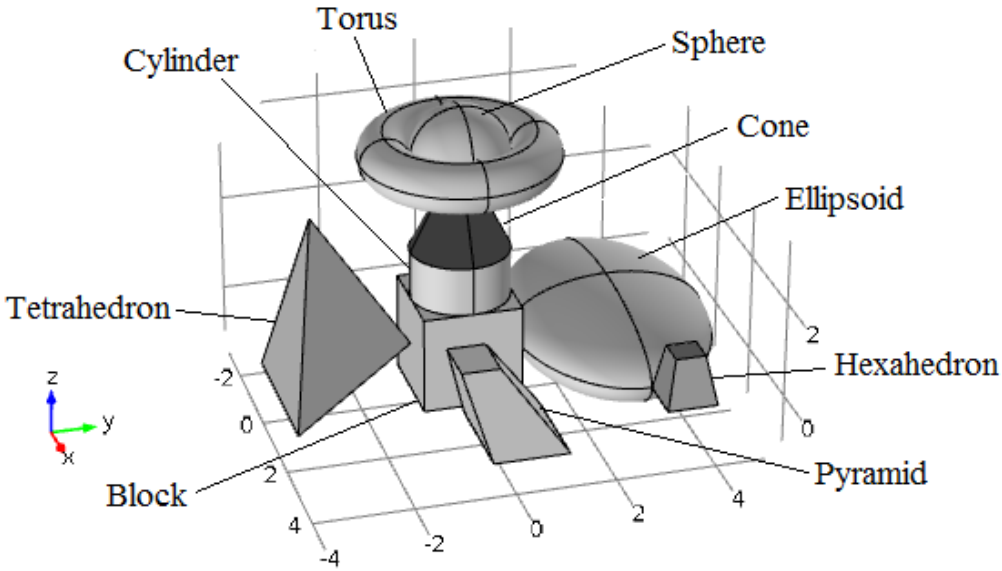




■

Ми не описували процес створення тривимірних графічних примітивів. Потренуйтесь в цьому самостійно.

Вправа. Використовуючи тривимірні графічні примітиви, створіть збірку, яка зображена на наступному рисунку (звісно без виносок).



Команди/кнопки побудови тривимірних примітивів розташовані на панелі креслення `Geometry` інструментальної стрічки.

Оскільки не всі тіла на рисунку перетинаються, то створити з них єдине тіло неможливо. Набір неперетинних солідних об'єктів називається збіркою (assembly). Щоб її створити в гілці **Geometry** в її останньому вузлі **Form Union** в полі **`Form Union/Assembly`** - **Action** слід вибрати значення `Form an assembly`. Інакше при виконанні команди Build All ви отримаєте повідомлення про помилку.

1.6. Демонстраційні приклади.

1.6.1. Охолодження куба.

Розв’яжемо задачу поширення тепла в одиничному кубі, поверхня якого підтримується при нульовій температурі, а початкова температура дорівнює 1.

Створіть модель послідовністю команд:

File->New->Model Wizard->3D->

Heat Transfer - Heat Transfer in Solids (ht) ->Time Dependent

Будемо розв’язувати безрозмірні рівняння. Для цього в ПВ вузла **Component 1 (comp1)** введіть:

`General` - **Unit system** = `None`

Геометрія моделі складається з одного куба. Намалюємо його.

При активному вузлі **Geometry 1** виконайте послідовність команд:

`Geometry`**->Pimitives->**Block

За замовчуванням параметри блоку збігаються з тими, які нам потрібні. Не змінюйте їх. Завершіть креслення куба клацанням по кнопці Build All Objects.

Відкрийте ПВ вузла **Heat Transfer in Solids(ht) - Heat Transfer in Solids 1**. В розділі **Equation** цієї панелі наведено загальний вигляд нестаціонарного рівняння теплопровідності

$$\rho C_\rho \frac{\partial T}{\partial t} + \rho C_\rho \mathbf{u} \cdot \nabla T + \nabla \cdot \mathbf{q} = Q + Q_{ted},$$

де T - шукана функція температури, $\mathbf{q} = -k\nabla T$ – тепловий потік, k - коефіцієнт теплопровідності, $\nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right]$ – оператор градієнта. Поле швидкостей **u** в нашій задачі не використовується, так само як і джерела тепла, які представлені в правій частині рівняння. За замовчуванням вони всі дорівнюють нулю. Тому рівняння має вигляд $\rho C_\rho \frac{\partial T}{\partial t} - k(\nabla \cdot \nabla T) = 0$.

В ПВ вузла **Heat Transfer in Solids 1** введіть:

Heat Conduction,Solid – **Thermal conductivity -k**= User defined
– **Thermal conductivity -** <> = 1

Thermodynamics,Solid – **Density** – ρ, <> = User defined,1

Heat capacity at constant pressure – C_ρ, <>= User defined,1

Перевірте, що в ПВ вузла **Heat Transfer in Solids 1** в розділі **Domain Selection** в секції **Selection** обрана область з індикатором 1.

Задайте початкову температуру. Для цього в ПВ вузла **Initial Values 1** введіть:

Initial Values – **Temperature** – **T** = 1

Задайте граничну температуру. Для цього виконайте послідовність команд:

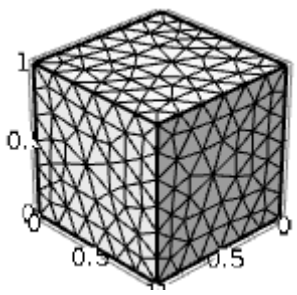
Physics->**Boundaries** ->Temperature

Активуйте вузол **Temperature 1**, натисніть кнопку Active в розділі **Boundary Selection** і в списку **Selection** виберіть All boundaries. Номери всіх граней 1,2,3,4,5 і 6 додадуться до списку **Selection**. В розділі **Temperature** в поле $\mathbf{T_o}$ введіть 0.

Побудуйте СкЕ сітку. Для цього в ПВ вузла **Mesh 1** введіть:

Mesh Settings – **Element size** = Normal

і натисніть кнопку Build All. У графічному вікні ви побачите зображення СкЕ сітки в кубі.



Зауважимо, що тривимірні обчислення тривають значно довше двовимірних. Тому не намагайтеся відразу будувати дрібну сітку. Виконайте всі обчислення для «нормальної» сітки і потім, якщо буде потреба, подрібніть сітку.

В дереві моделі активуйте вузол **Step 1: Time Dependent**. В його ПВ задайте послідовність моментів часу, в яких буде обчислюватися розв'язок:

Study Settings –**Times** = 0.001 0.005 range(0.01,0.01,0.1)

Розв'яжіть задачу, виконавши команду

Study-> Compute

В дереві моделі в гілці **Results** з'явиться гілка **Temperature (ht)** з вузлом **Surface 1**, що містить функціональне забарвлення поверхні куба, яке представляє температуру. З'явиться також гілка **Isothermal Contours (ht)**, яка містить контурний графік температури на поверхні куба. Обидва графіка неінформативні, оскільки на поверхні куба у нас температура незмінна і дорівнює 0. Видалимо ці графіки. Клацніть правою кнопкою миші по вузлу **Temperature (ht)** і в випадному меню виберіть команду Delete. Аналогічно видаліть гілку **Isothermal Contours (ht)**.

Побудуємо графік розподілу температури всередині куба. Створіть тривимірну графічну групу 3D Plot Group командою

Results -> 3D Plot Group

3D Plot Group 3 :

Data– **Data set** = Study 1/Solution 1

Data– **Time (s)** = 0.02

Color Legend – **Show legends** =True

В графічній групі **3D Plot Group 3** створіть підвузол Multislice.

3D Plot Group 3: RMB > More Plots - Multislice

3D Plot Group 3 – Multislice 1:

Data– **Data set** = From parent

Expression– **Expression** = T

Multiplane Data – **x-planes, y-planes, z-planes** = 1,1,1 (*три поля*)

Це кількість площин – по одній площині кожної орієнтації.

Range – **Manual color range** = True

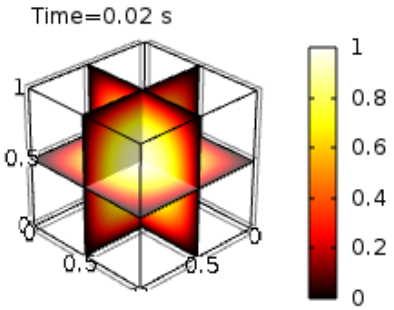
– **Minimum, Maximum** = 0,1

Coloring and Style – **Coloring** =Color table

– **Color** =Thermal

– **Color legend** = True

Виконайте команду Plot і ви отримаєте зображення трьох взаємно ортогональних площин, що проходять через центр куба, з функціональним забарвленням, яке відповідає температурі куба в момент часу t = 0.02.



Змінюючи в ПВ вузла **3D Plot Group 3** значення в полі **Data - Time (s)**, ви будете отримувати аналогічні графіки для інших моментів часу.

В динамічних задачах можна будувати графіки залежності розв'язку від часу в точках. Для цього треба використовувати графічні вузли типу **Point Graph**, які слід прив'язувати до точкового джерела даних `Cut Point 3D` або просто до точок, що вже існують в геометрії. Якщо в ПВ такого вузла в списку **Time selection** вибрати `All`, то буде побудовано часовий графік по значенням розв'язку в точці в усі моменти часу.

Побудуємо графік залежності температури від часу в центрі куба. Для цього спочатку створимо точкове джерело даних, виконавши інструкції

`Results` **`-> Data Set ->`** Cut Point 3D

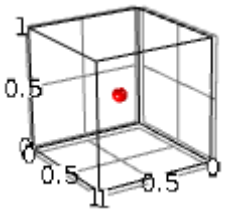
Data Sets - Cut Point 3D 1:

`Data` – **Data set** = `Study 1/Solution 1`

`Point Data` –**Entry method** = `Coordanates`

`Point Data` – **x, y, z** = `0.5, 0.5, 0.5` (*три поля*)

Перевірте введення точки командою Plot.



Створіть групу одновимірних графіків командою

`Results` -> 1D Plot Group

1D Plot Group 4:

`Data` – **Data set** = `Cut Point 3D 1`

`Data` – **Time selection** = `All`

На стрічці панелей з'являється закладка `1D Plot Group 4`. З неї виберіть команду

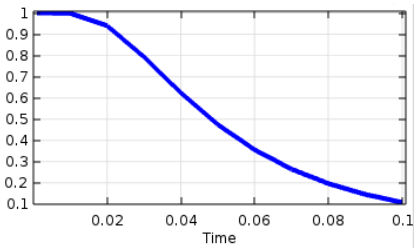
`1D Plot Group 4`-> Point Graph

Point Graph 1:

`y-Axis Data` – **Expression** = `T`

`x-Axis Data` – **Parameter** = `Time`

Побудуйте графік залежності температури в точці від часу командою Plot. Якщо треба, то налаштуйте колір і стиль кривої в розділі **`Coloring and Style`** ПВ вузла **Point Graph 1**.



Надрукуємо значення температури в центрі куба в різні моменти часу. Для цього у вузлі **Derived Values** створіть вузол типу `Point Evaluation` наступною командою

Derived Values: RMB > Point Evaluation
В ПВ створеного вузла **Point Evaluation 1** введіть:

`Data` – **Data set** = `Cut Point 3D 1`

`Data` – **Time selection** = `All`

`Expression` – **Expression** = `T`

LMB > Evaluate – New Table

В інформаційній зоні буде створена таблиця **Table 1** така, як на наступному рисунку, але без правого стовпця.

Побудувати графік таблиці

Time (s)	Temperature, Point: (	Temperature
0.0010000	1.0000	0.93156
0.0050000	1.00000	0.62964
0.010000	0.99856	0.48506
0.020000	0.94055	0.32646
0.030000	0.79333	0.23262
0.040000	0.62396	0.16965
0.050000	0.47457	0.12501
0.060000	0.35554	0.092424
0.070000	0.26429	0.068276
0.080000	0.19583	0.050504
0.090000	0.14484	0.037346
0.10000	0.10719	0.027645

Таблиця містить значення температури в обраній точці в моменти часу, які задані у вузлі **Step 1: Time Dependent** в полі `Study Settings` –**Times**. Ці моменти `0.001 0.005 0.01 ... 0.09 0.1` відображені в лівому стовпці таблиці.

Побудуємо графік середньої температури куба в залежності від часу. Для цього виконайте інструкції

Derived Values: RMB > Average - Volume Average
В ПВ створеного вузла **Volume Average 1** введіть:

`Data` – **Data set** = `Study 1/Solution 1`

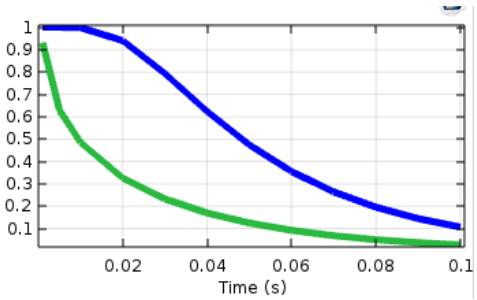
`Data` – **Time selection** = `All`

В секцію вибору введіть 1, клацнувши в графічній зоні по кубу. Далі введіть

`Expression` – **Expression** = `T`

LMB > Evaluate – Table 1 - Point Evaluation 1(T)

В таблиці **Table 1** буде додано стовпець зі значеннями середньої температури в кубі в різні моменти часу (див. попередній рисунок). Якщо клацнути мишею по кнопці Table Graph, яка розташована над таблицею, то будуть побудовані графіки ламаних, абсциса яких буде взята з першого стовпця таблиці, а ординати – з другого і третього. При цьому верхню криву ми будували раніше.



Існує інший спосіб побудови графіка залежності середньої температури куба від часу. Для цього потрібно створити джерело даних, яке буде представляти інтеграл від температури по об'єму куба (або середнє по об'єму).

Results – Data Sets: RMB > More Data Sets - Integral

В ПВ створеного вузла **Integral 1** введіть:

Data – **Data set** = Study 1/Solution 1

Settings – **Geometry level** = Volume

В графічну групу **1D Plot Group 4** додайте і налаштуйте графічний вузол типу Global. Для цього виконайте інструкції

1D Plot Group 4-> Global

В ПВ створеного вузла **Global 1** введіть:

Data – **Data set** = Integral 1

Data – **Time selection** = All

y-Axis Data – **Expression** = T (*введення виконується в таблиці*)

x-Axis Data – **Parameter** = Time

Виконайте команду Plot і отримайте графік такий же, як і попередній, побудований по табличним даним.

Побудуємо графік середньої температури по прерізу куба $x = 0.9$. Спочатку треба створити джерело даних для цього перерізу.

Results -> **Data Set** -> Cut Plane

В ПВ створеного вузла **Cut Plane 1** введіть:

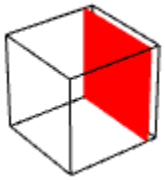
Data – **Data set** = Study 1/Solution 1

Plane Data – **Plane type** = Quick

Plane Data – **Plane:** = yz-planes

Plane Data – **x-coordinate:** = 0.9

Побудуйте зображення цього джерела даних командою Plot.



Створіть джерело даних, яке буде представляти середнє значення температури по площині перерізу. Для цього виконайте інструкції

Results – Data Sets: RMB > More Data Sets - Average

В ПВ створеного вузла **Average 1** введіть:

Data – **Data set** = Cut Plane 1

Settings – **Geometry level** = Take from data set

В графічну групу **1D Plot Group 4** додайте і налаштуйте графічний вузол типу `Global`. Для цього виконайте інструкції

`1D Plot Group 4->` Global

В ПВ створеного вузла **Global 2** введіть:

`Data` – **Data set** = `Average 1`

`Data` – **Time selection** = `All`

`y-Axis Data` – **Expression** = `T` (*введення в таблиці*)

`x-Axis Data` – **Parameter** = `Time`

Щоб на графіку відобразити легенду, виконайте інструкції

1D Plot Group 4:

`Legend` – **Show legends, Position** = `True, Upper right`

1D Plot Group 4 - Point Graph 1:

`Legends` – **Show legends, Legends** = `True, 'Center'`

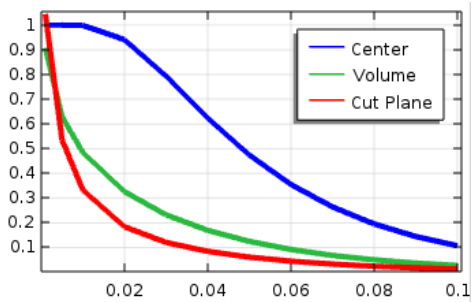
1D Plot Group 4 - Global 1:

`Legends` – **Show legends, Legends** = `True, 'Volume'`

1D Plot Group 4 - Global 2:

`Legends` – **Show legends, Legends** = `True, 'Cut Plane'`

Закінчіть побудову графіка командою Plot.



В динамічних задачах, так само як і при параметричному аналізі, зручно використовувати зонди. Для прикладу створимо два зонди. Перший `Domain Point` зонд буде відслідковувати температуру в центрі куба, другий – `Domain` зонд буде обчислювати середню температуру по об'єму.

Спочатку створимо точковий зонд.

Component 1 (comp1) – Definitions:

RMB > Probes – Domain Point Probe

Definitions - Domain Point Probe 1:

`Point Selection` – **Line entry method** = `None`

`Point Selection` – **Coordinates – x, y, z** = `0.5,0.5,0.5` (*три поля*)

Definitions - Domain Point Probe 1 - Point Probe Expression 1 (ppb1):

`Expression` – **Expression** = `T`

LMB > Update Results

Тепер створимо «просторовий» зонд.

Component 1 (comp1) – Definitions:

RMB > Probes – Domain Probe

Definitions - Domain Probe 1 (dom1):

`Probe Type` – **Type** = `Average`

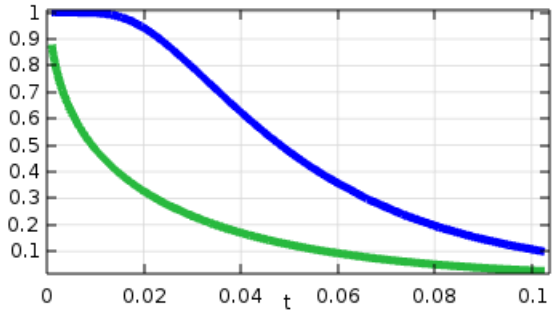
`Source Selection` – **Selection** = `All domains`

Expression – **Expression** = T

Study 1: RMB > Compute

В результаті переобчислення моделі будуть створені всі «зондові» вузли, побудована таблиця залежності зондових змінних від часу, і в графічній групі **Probe Plot Group 7** буде створено дві криві, такі ж самі, як раніше.

t	Temperature, Po	Temperature, D
0.0051887	1.00000	0.61987
0.0062042	0.99995	0.58466
0.0072196	0.99978	0.55376
0.0082351	0.99950	0.52623
0.0092505	0.99906	
0.010266	0.99831	
0.080738	0.19173	0.049451
0.088049	0.15382	0.039663
0.095361	0.12337	0.031815
0.10267	0.098948	0.025520



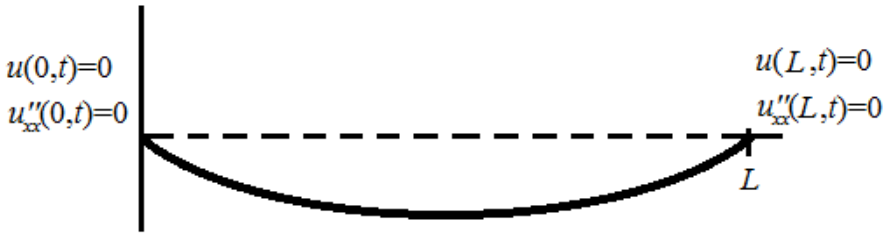
1.6.2. Коливання вільно опертої балки.

Дослідіть малі коливання тонкої балки довжиною L, кінці якої вільно оперті.

Урахування факту, що балка чинить опір згину, призводить до рівняння четвертого порядку

$$u''_{tt} = -\alpha^2 u''''_{xxxx}, \quad 0 < x < L, 0 < t < \infty, \tag{1}$$

де $\alpha^2 = K/\rho$, K - згинна жорсткість, ρ – лінійна густина балки (маса/од. довжини), $u(x,t)$ - вертикальне зміщення точки x балки в момент часу t.



Зрозуміло, що на кінцях балки виконуються граничні умови

$$u(0,t) = 0, u(L,t) = 0 \quad (0 \le t < \infty). \tag{2}$$

Використовуючи теорію тонких балок, можна показати, що згинальний момент в балці пропорційний величині u''_{xx}. Оскільки згинальні момент на кінцях вільно опертої балки дорівнює нулю, то отримуємо додаткові граничні умови

$$u''_{xx}(0,t) = 0, u''_{xx}(L,t) = 0 \quad (0 \le t < \infty). \tag{3}$$

В результаті, коливання балки описуються рівнянням (1), граничними умовами (2), (3), і початковими умовами

$$u(x,0) = \varphi(x), u'_t(x,0) = \psi(x), 0 \le x \le L.$$

Вивчимо коливання балки одиничної довжини $L = 1$, що вільно обперта на кінцях. Нехай $\alpha^2 = 1$ і початкові умови мають вигляд

$$u(x,0)=\sin\pi x+0.5\sin 3\pi x$$
$$u_t'(x,0)=0 \qquad (4)$$

Аналітичний розв’язок цієї задачі відомий

$$w(x,t)=\cos(\pi^2 t)\sin\pi x+0.5\cos(9\pi^2 t)\sin(3\pi x). \qquad (5)$$

Побудуємо чисельний розв'язок задачі за допомогою COMSOL і порівняємо його з (5).

COMSOL вміє розв’язувати системи ДРЧП не вище другого порядку. Тому задача повинна бути переписана в термінах системи двох ДР другого порядку. Якщо ввести позначення $v=u''_{xx}$, то рівняння (1) набуде вигляду $u''_{tt}=-v''_{xx}$ (при $\alpha^2=1$). В результаті задача може бути переписана у вигляді наступної системи

$$\begin{cases} u''_{tt}+v''_{xx}=0 \\ -u''_{xx}+v=0 \end{cases} \qquad (6)$$

з граничними умовами

$$u(0,t)=0,\ u(1,t)=0,\ v(0,t)=0,\ v(1,t)=0,$$

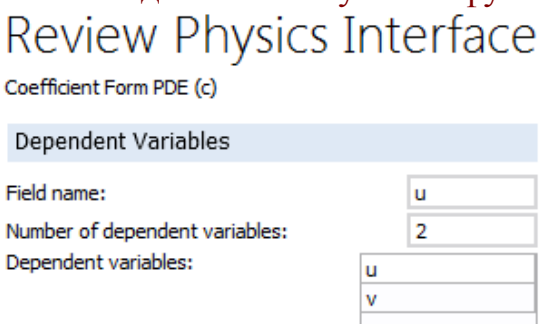
і початковими умовами (4).

Виконайте послідовність команд:

File->New->Model Wizard->1D->

Mathematics – PDE Interfaces – Coefficient From PDE(c)

Після натискання на кнопку Add в поле **Added physics interfaces** додається назва моделі. Справа в панелі Review Physics Interface в розділі **Dependent Variable** в поле **Number of dependent variables** введіть 2, а в поле **Dependent variables** введіть імена шуканих функцій u і v.



Натисніть кнопку Study. У відкрившомуся вікні Select Study виберіть варіант TimeDependent. Завершіть створення дерева моделі натисканням на кнопку Done.

Відкрийте ПВ вузла **Component 1 (comp 1)** і введіть

General - Unit system = None

Геометрія моделі складається з одного відрізка. Побудуйте його.

Geometry->**Pimitives->** Interval

За замовчуванням параметри відрізка збігаються з тими, які нам потрібні. Не міняйте їх. Завершіть креслення геометрії клацанням по кнопці Build All Objects.

Побудуйте СкЕ сітку. Для цього в ПВ вузла **Mesh 1** введіть:

Mesh Settings –Element size = Finer

і натисніть кнопку Build All. В графічному вікні ви побачите зображення відрізка з точками розбиття.

Відкрийте ПВ вузла **Coefficient Form PDE (c) - Coefficient Form PDE 1**. В розділі **Equation** наведено загальний вигляд коефіцієнтної форми системи ДРЧП.

$$e_a \frac{\partial^2 \boldsymbol{u}}{\partial t^2} + d_a \frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \left(-c \nabla \boldsymbol{u} - \alpha \boldsymbol{u} + \gamma\right) + \beta \cdot \nabla \boldsymbol{u} + a \boldsymbol{u} = f \ ,$$

де $\boldsymbol{u} = \begin{pmatrix} u \\ v \end{pmatrix}$ - шукана вектор - функція і $\nabla = \dfrac{\partial}{\partial x}$.

Система рівнянь (6) в матричному запису має вигляд

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \frac{\partial^2}{\partial t^2} \begin{pmatrix} u \\ v \end{pmatrix} + \frac{\partial}{\partial x} \left(- \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \frac{\partial}{\partial x} \begin{pmatrix} u \\ v \end{pmatrix} \right) + \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Щоб коефіцієнтна форма набула такого вигляду, необхідно ввести наступні значення:

Diffusion Coefficient $- \mathbf{c} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$

Absorption Coefficient $- \mathbf{a} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$

Source Term $- \mathbf{f} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

Mass Coefficient $- \mathbf{e_a} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$

Тензори d_a, α, β і вектор γ в нашій задачі слід покласти рівними нулю. Перевірте також, що в поточній панелі (`Coefficient Form PDE 1`) в розділі **Domain Selection** стоїть 1, тобто обраний відрізок.

Задайте граничні умови, виконавши команду

`Physics->`**Boundaries**`->`Dirichlet Boundary Condition

Активуйте створений вузол **Dirichlet Boundary Condition 1** і натисніть кнопку Active в розділі **Boundary Selection**, якщо вона ще не натиснута. Мишкою клацніть по лівому кінцю відрізка (точка 1), а потім по правому кінцю (точка 2), додавши індикатори точок в секцію вибору **Selection**. В розділі **Dirichlet Boundary Condition** в полях $\mathbf{r_1}$ і $\mathbf{r_2}$ введіть 0 - граничні значення функцій `u` і `v`.

Зайдіть в ПВ вузла **Initial Values 1** і перевірте, що в списку **Domain Selection** - **Selection** стоїть одиниця (індикатор відрізка). Потім введіть

Initial Values – Initial value for u – u =
`sin(pi*x)+0.5*sin(3*pi*x)`

Initial Values – Initial value for v – v =
`-pi^2*sin(pi*x)-0.5*9*pi^2*sin(3*pi*x)`

Initial Values – Initial time derivative of u – $\frac{\partial u}{\partial t}$ = 0

Initial Values – Initial time derivative of v – $\frac{\partial v}{\partial t}$ = 0

Зверніть увагу, що потрібно задати початкові умови для функції v, хоча за змістом задачі вони непотрібні. Оскільки $v = u''_{xx}$, то записати початкові умови для функції v нескладно: $v(0,x) = u''_{xx}(0,x)$. Однак для функції v можна залишити нульові початкові значення або ввести будь-які інші. Результат не зміниться, оскільки початкові умови для функції v при розв'язанні задачі не використовуються.

Налаштуйте часовий інтервал і похибки обчислень. Для цього в ПВ вузла **Step 1: Time Dependent** введіть

Study Settings – **Times** = `range(0,0.05,0.5)`

Встановіть перемикач **Study Settings - Relative tolerance** і задайте відносну похибку обчислень `0.00001`.

Виконайте команду

`Study`-> Show Default Solver

відобразивши підвузли гілки **Study 1** дерева моделі (якщо в дереві їх ще немає). Розкрийте гілку **Study 1** і активуйте вузол **Study 1 - Solver Configurations - Solution 1 - Time - Dependent Solver 1**. В його ПВ введіть

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

Виконайте обчислення командою

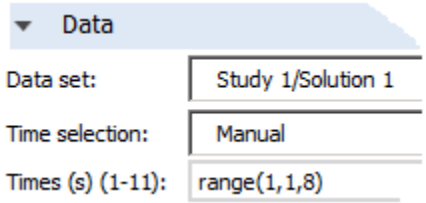
`Home` -> Compute

В гілці **Results** з'явиться гілка **1D Plot Group 1** групи графіків з вузлом **Line Graph 1**, що містить графіки розв'язку $u(t,x)$ в різні моменти часу. В ПВ вузла **1D Plot Group 1** ведіть

Data – **Time selection** = `Manual`

Data – **Time(s)(1-11)** = `range(1,1,8)`

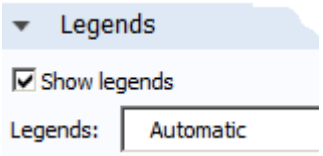
вибравши цим перші 8 моментів часу, щоб побудувати графіки тільки для них.



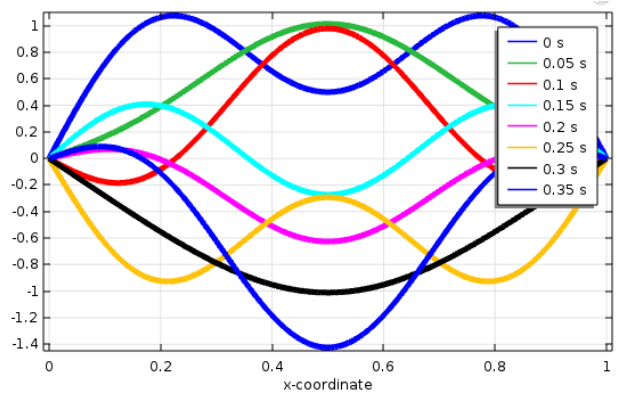
В ПВ вузла **1D Plot Group 1 - Line Graph 1** введіть

x-Axis Data – **Parameter, Expression** = `Expression, x`

щоб віднести графіки до координати x вздовж осі балки. Для відображення легенди в ПВ вузла **1D Plot Group 1 - Line Graph 1** встановіть перемикач **Show legends**.



В результаті ви отримаєте графіки нейтральної вісі балки для моментів часу, наведених в «легенді».



Для порівняння побудуємо графіки нейтральної вісі, використовуючи чисельний розв'язок і аналітичну формулу (5).

Створіть групу одновимірних графіків командою

`Results` -> 1D Plot Group

В ПВ створеного вузла **1D Plot Group 2** введіть:

Data – **Data set** = `Study 1/Solution 1`

Data – **Time selection** = `Manual`

Data – **Time(s)**(`1-11`) = `1,4,8`

Потім двічі виконайте команду

`1D Plot Group 2`-> Line Graph

В ПВ створеного вузла **Line Graph 1** в розділі **Selection** оберіть відрізок (його індикатор 1 з'явиться в секції вибору) і введіть

y-Axis Data – **Expression** = `u`

x-Axis Data – **Parameter, Expression** = `Expression, x`

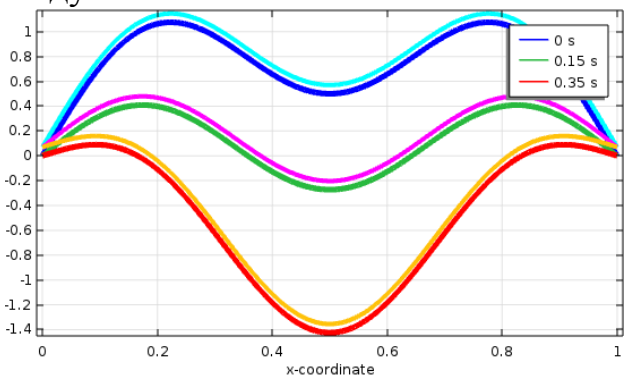
Legends – **Show legends** = `True`

В ПВ створеного вузла **Line Graph 2** в розділі **Selection** виберіть відрізок і введіть

y-Axis Data – **Expression** = `cos(pi^2*t)*sin(pi*x)+`
`0.5*cos(9*pi^2*t)*sin(3*pi*x)+0.07`

x-Axis Data – **Parameter, Expression** = `Expression, x`

В результаті графічна група буде містити графіки нейтральної вісі, побудовані для трьох моментів часу (`0`, `0.15` і `0.35`) з використанням чисельного розв'язку та аналітичної формули (5). Для наочності графіки аналітичного розв'язку зсунуті вгору на невелику константу `0.07`. Якщо її з формули забрати, то криві співпадуть.



Вправа. Побудуйте графік різниці чисельного і аналітичного розв'язків. Знайдіть абсолютну похибку (максимум модуля різниці цих розв'язків). Побудуйте графік різниці перших похідних по x чисельного і аналітичного розв'язків. Побудуйте і порівняйте графіки функції $v(x,t)$, $u_{xx}(x,t)$ та $w_{xx}(x,t)$.

1.6.3. Осесиметричні коливання круглої мембрани.

Вільні осесиметричні коливання кругової мембрани радіуса R, яка закріплена на контурі, описуються рівняннями

$$\frac{1}{a^2}\frac{\partial^2 u}{\partial t^2}=\frac{\partial^2 u}{\partial r^2}+\frac{1}{r}\frac{\partial u}{\partial r},\quad u(t,R)=0 \tag{1}$$

Розв'яжемо задачу (1) з початковими умовами $u(0,r)=R^2-r^2$, $u'_t(0,r)=0$, які описують коливання з параболоїдальною початковою формою і нульовою початковою швидкістю. Для прикладу візьмемо $R=1$ і $a=1$.

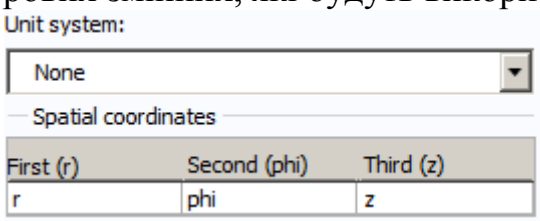
Створіть модель послідовністю команд:

File->New->Model Wizard->1D Axisymmetric->
Mathematics – PDE Interfaces – Coefficient From PDE(c) ->
TimeDependent

Відкрийте ПВ вузла **Component 1 (comp 1)** і введіть

General - **Unit system** = None

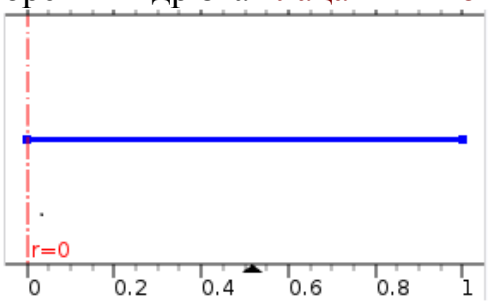
Тут же в таблиці розділу **Spatial coordinates** ви можете дізнатися (або поміняти) імена просторових змінних, які будуть використовуватися в моделі.



Геометрія складається з одного відрізка, обертанням якого створюється осесиметрична модель. Побудуйте його.

Geometry -> **Pimitives** -> Interval

За замовчуванням параметри відрізка збігаються з тими, які нам потрібні. Не міняйте їх. Завершіть створення відрізка клацанням по кнопці Build All Objects.



Відкрийте ПВ вузла **Coefficient Form PDE (c) - Coefficient Form PDE 1**. В розділі **Equation** цієї панелі наведено загальний вигляд коефіцієнтної форми ДРЧП

$$e_a\frac{\partial^2 u}{\partial t^2}+d_a\frac{\partial u}{\partial t}+\nabla\cdot(-c\nabla u-\alpha u+\gamma)+\beta\cdot\nabla u+\mathrm{a}u=f,$$

де $\nabla = \frac{\partial}{\partial r}$. Щоб це рівняння прийняло вид (1), необхідно ввести наступні значення:

Diffusion Coefficient – **c** = 1

Absorption Coefficient – **a** = 0

Source Term – **f** = 0

Mass Coefficient – $\mathbf{e_a}$ = 1

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Convection Coefficient – $\boldsymbol{\beta}$ = -1/r

Значення коефіцієнтів α і γ залиште без змін (нульовими). Перевірте також, що в поточній панелі (Coefficient Form PDE 1) в секції вибору розділу **Domain Selection** стоїть 1, тобто обраний відрізок.

Задайте граничні умови, виконавши команду

Physics->**Boundaries**->Dirichlet Boundary Condition

Активуйте створений вузол **Dirichlet Boundary Condition 1**. В його ПВ в розділі **Boundary Selection** натисніть кнопку Active (якщо вона ще не активна) і мишею виділіть праву точку відрізка. Її індикатор 2 буде додано в секцію вибору **Selection**. В розділі **Dirichlet Boundary Condition** в полі **r** залиште значення 0.

Відкрийте в ПВ вузла **Zero Flux 1** і переконайтеся, що для лівої точки (індикатор 1) гранична умова не застосовується (насправді в цій точці воно автоматично задається умовою симетрії $\left.\frac{\partial u}{\partial r}\right|_{point1} = 0$).

Зайдіть в ПВ вузла **Initial Values 1** і перевірте, що в списку **Domain Selection** – **Selection** стоїть одиниця (індикатор відрізка). Потім введіть

Initial Values – **Initial value for u** – **u** = 1-r^2

Initial Values – **Initial time derivative of u** – $\frac{\partial u}{\partial t}$ = 0

Налаштуйте часовий інтервал і похибку обчислень. В ПВ вузла **Step 1: Time Dependent** введіть

Study Settings – **Times** = range(0,0.05,10)

Цим ми задаємо інтервал часу (від 0 до 10), на якому буде визначатися чисельний розв'язок, а також інтервал між моментами часу, для яких розв'язок буде зберігатися.

Встановіть прапорець в перемикачі **Study Settings** – **Relative tolerance** і задайте відносну похибку обчислень 0.00001.

Виконайте команду

Study-> Show Default Solver

відкривши підвузли гілки **Study 1** дерева моделі. Розкрийте цю гілку і активуйте вузол **Study 1 - Solver Configurations - Solution 1 - Time - Dependent Solver 1**. В його ПВ введіть

Absolute Tolerance – **Tolerance** = 0.00001

Побудуйте СкЕ сітку. Для цього в ПВ вузла **Mesh 1** введіть:

Mesh Settings –**Element size** = Finer

і натисніть кнопку Build All. У графічному вікні ви побачите зображення відрізка з точками розбиття.

Виконайте обчислення командою

Home -> Compute

В гілці **Results - Data Sets** з'являться два вузли **Study 1/Solution 1** і **Revolution 1D 1** з розв'язками, а також вузли, що містять їх графіки. Спочатку графічне вікно, пов'язане з вузлом **1D Plot Group 1** містить графіки розв'язку $u(t,r)$ для всіх моментів часу з масиву range(0,0.05,10). Його ми задали в ПВ вузла **Step 1: Time Dependent** в полі **Times**. Побудований графік малоінформативний, оскільки кривих багато. Виберемо з них тільки деякі. Для цього в ПВ вузла **1D Plot Group 1** введіть:

Data– **Data set** = Study 1/Solution 1

Data– **Time selection** = Manual

Data– **Time(s)**(1-101) = range(1,1,14)

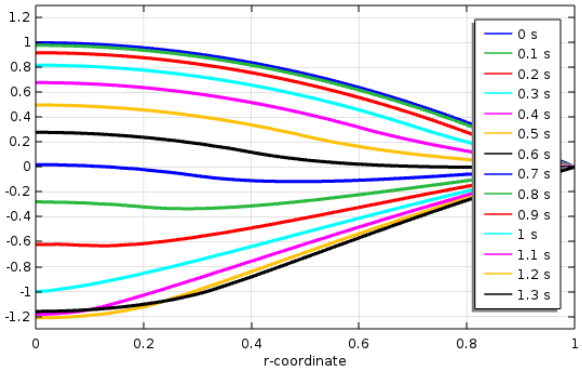
Axis–**Manual axis limits**= Включити

Axis–**x minimum, x maximum** = 0, 1 (*два поля*)

Axis–**y minimum, y maximum** = -1.5 , 1.5 (*два поля*)

Legend–**Show legends** = True

Задання в полі **Time(s)** списку номерів моментів часу дозволяє побудувати графіки тільки в бажані моменти. Вмикання «легенди» (**Show legends** = True) в графічній групі (а також в дочірньому графічному вузлі **Line Graph 1**) відображає на графіку таблицю відповідності моментів часу і кривих.



Графічна група **2D Plot Group 2** відображає графік розв'язку $u(t,x,y)$ для моменту часу, який обраний в списку **Data** – **Time(s)**. Встановимо для відображення нульовий момент часу.

2D Plot Group 2: Data – **Time (s)** = 0

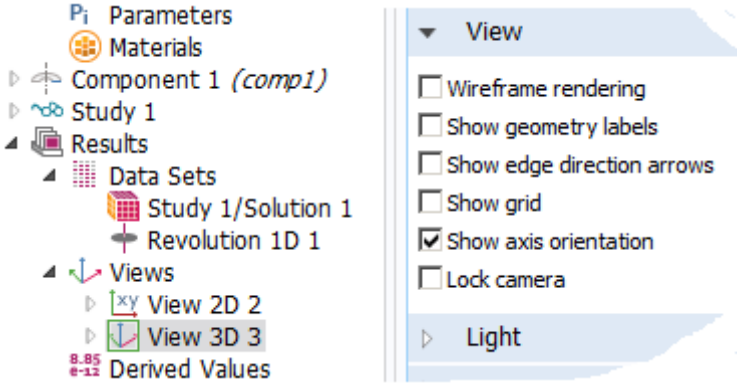
Скорегуємо джерело даних **Revolution 1D 1**, щоб воно представляло повне коло. Для цього в ПВ вузла **Data Sets - Revolution 1D 1** введіть

Revolution Layers – **Start angle, Revolution angle** = 0, 360

і натисніть кнопку Plot.

Клацніть правою кнопкою миші по гілці **2D Plot Group 2 - Surface 1** і додайте підвузол **Height Expression 1**. В його ПВ в полі **Scale factor** встановіть масштабний коефіцієнт вертикальних розмірів 1, і натисніть кнопку Plot. На

зображенні, крім поверхні, буде показана просторова координатна сітка. Щоб її видалити в ПВ вузла **Results - Views -View 3D 3** зніміть перемикач **Show grid**.



Включати і виключати координатну сітку можна також кнопкою Show Grid, яка розташована над графічним вікном.

Змінюючи в ПВ вузла **2D Plot Group 2** в списку **Data – Time(s)** значення моменту часу, ви будете отримувати новий графік поверхні мембрани. Однак кольорова палітра для кожного графіка буде своя. Щоб графіки в різні моменти часу використовували однакову палітру і розташовувалися в одній і тій же просторовій зоні, потрібно «вручну» задати діапазони зміни кольору і даних. Для цього в ПВ вузла **2D Plot Group 2 – Surface 1** введіть

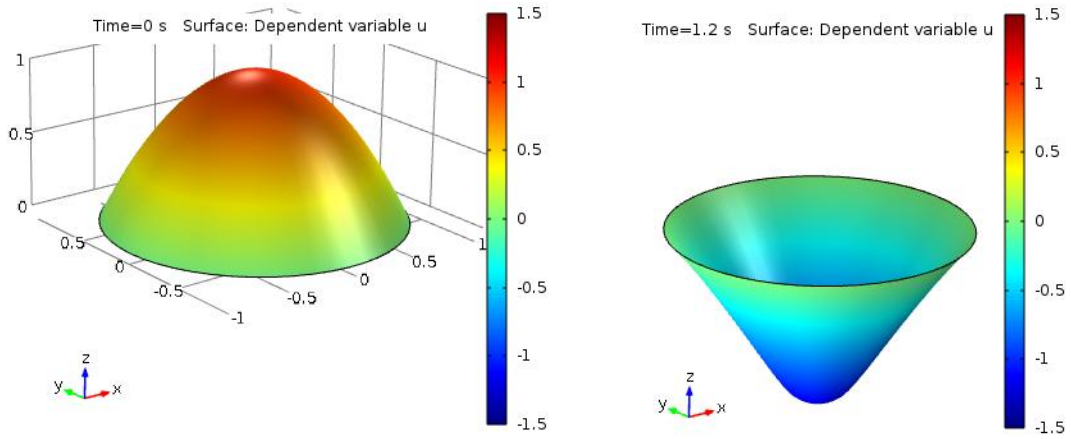
Range – Manual color range = `True`

Range – Manual color range – Minimum, Maximum = `-1.5, 1.5`

Range – Manual data range = `True`

Range – Manual data range – Minimum, Maximum = `-1.5,1.5`

Зображення поверхні мембрани в нульовий момент часу (з координатної сіткою), і в момент `1.2` (без сітки) показані на наступному рисунку. Щоб отримати такі зображення, можливо, вам буде потрібно змістити об'єкти в графічному вікні. Для цього підведіть покажчик миші до зображення, натисніть праву кнопку і, не відпускаючи, перемістіть мишу (а разом з нею зображення) в нове місце.



Побудуйте графік лінії перерізу поверхні мембрани вертикальною площиною. Для цього спочатку додайте відповідне джерело даних, виконавши команду

`Results`-> Cut Line 2D

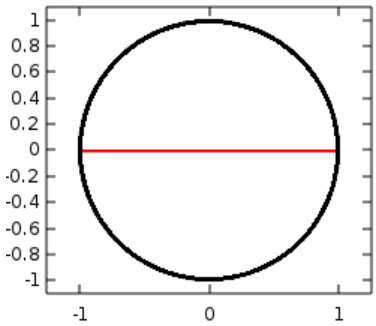
В ПВ створеного вузла **Cut Line 2D 1** введіть

Data – Data set = `Revolution 1D 1`

Line Data – Point 1 – r = -1

Line Data – Point 2 – r = 1

Побудуйте зображення джерела даних командою Plot.



Тепер створіть групу одновимірних графіків командою

Results –> 1D Plot Group

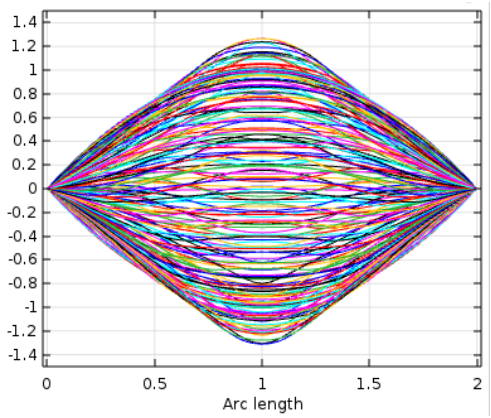
В ПВ створеного вузла **1D Plot Group 3** введіть:

Data – Data set = Cut Line 2D 1

Потім виконайте команду

1D Plot Group 3–> Line Graph

Налаштування ПВ створеного вузла **Line Graph 1** залиште без змін. В результаті графічна група буде містити велику кількість графіків для всіх моментів часу з масиву range(0,0.05,10).



Зверніть увагу, що горизонтальна вісь віднесена до довжини дуги джерела даних **Cut Line 2D 1**. Наведені графіки малоінформативні, але вони знадобляться для побудови анімації. Для цього виконайте команду

Results –> Animation –> Player

В ПВ створеного вузла **Animation 1** введіть:

Scene – Subject = 1D Plot Group 3

Target – Target = Player

Animation Editing – Loop over = Time

Animation Editing – Time selection = All

Frames – Frame selection = All

Згенеруйте кадри анімації, натиснувши кнопку Generate Frame. По завершенні процесу запустіть анімацію кнопкою Play, яка розташована над графічним вікном.

Збережіть поточний файл, наприклад, з ім'ям vibrmembrane.mph, бо він нам ще знадобиться.

Вправа. Розв'яжіть задачу про коливання квадратної мембрани $\Omega = [0,1] \times [0,1]$, яка закріплена на контурі. Для цього розгляньте рівняння $\frac{\partial^2 u}{\partial t^2} = a^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$ з граничною умовою $u\big|_{\partial\Omega} = 0$. Задайте початковий профіль у вигляді $u(0, x, y) = \sin \pi x \sin \pi y$, і нульову початкову швидкість. Порівняйте чисельний розв'язок з аналітичним, який має вигляд $u(t, x, y) = \cos\left(\sqrt{2}\, \pi a t\right) \sin \pi x \sin \pi y$, побудувавши графік різниці розв'язків уздовж діагоналі квадрата. Побудуйте анімацію коливань квадратної мембрани.

1.7. Математичні вирази та оператори.

У попередніх прикладах ми кілька разів використовували математичні вирази. Коротко опишемо правила їх побудови.

Математичні операції позначаються стандартними значками.

Унарні операції: !, –, + (логічне заперечення, мінус, плюс).

Бінарні арифметичні операції: ^, *, /, +, – (піднесення до степеня, множення, ділення, додавання і віднімання).

Операції порівняння: <, <=, >, >=, ==, !=

Логічні операції: &&, || (логічне І , логічне АБО).

Пріоритет арифметичних операцій звичайний. Для зміни порядку виконання операцій використовуються круглі дужки.

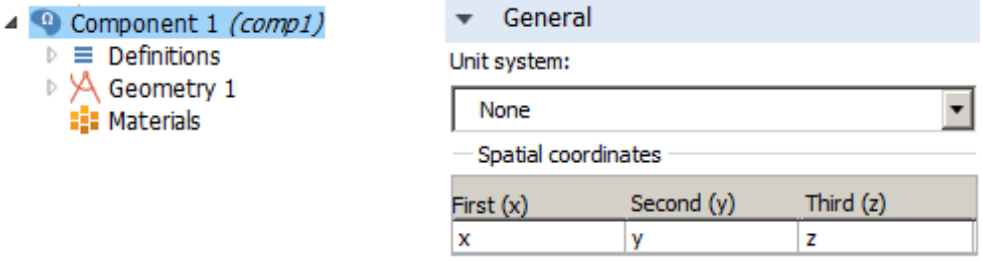
У виразах COMSOL Multiphysics можна використовувати різні вбудовані константи і функції. Їх імена зарезервовані і не можуть бути перепозначені користувачем. Якщо ви спробуєте використати таке ім'я для змінної, параметра або функції, то текст, який ви введете, стане помаранчевим (попередження) або червоним (помилка). При підведенні покажчика миші до такого тексту спливе пояснювальна підказка.

Ось приклади деяких зарезервованих імен:

- константа `pi` (3.141592...),
- `eps` – мінімальна відносна похибка (машинна іпсилон 2^{-52}, або приблизно $2.2204 \cdot 10^{-16}$);
- `inf`, `Inf` – нескінченність, ∞;
- `Nan`, `nan` (Not-a-number) – невизначене значення таке, як 0/0 або ∞/∞;
- фізичні сталі, наприклад, `g_const` (прискорення вільного падіння), `R_const` (універсальна газова стала)
- уявна одиниця `i` (або `j`) (тобто $\sqrt{-1}$);
- змінна часу `t`;
- назви математичних функцій, таких як `cos`, `sin`, `exp`, `log` та інших. Перелічимо позначення основних з них (імена аргументів можуть бути іншими): `abs(x)`, `acos(x)`, `asin(x)`, `cos(x)`, `cosh(x)`, `erf(x)`, `exp(x)`, `floor(x)`, `log(x)` (натуральний логарифм), `log10(x)`

(десятковий логарифм), `max(a,b)`, `min(a,b)`, `sign(u)` (функція знаку), `sin(x)`, `sinh(x)`, `sqrt(x)`, `tan(x)`, `tanh(x)`.

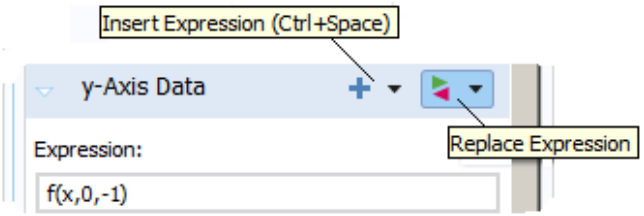
В моделях розмірності 1D, 2D, 3D використовуються зумовлені імена декартових координат `x,y,z`. Однак їх можна змінити в панелі властивостей гілки **Component**. В таблиці **`Spatial coordinates`** можна ввести інші позначення просторових змінних.



В осесиметричних моделях `1D Axisymmetric` і `2D Axisymmetric` циліндричні координати зазвичай позначаються `r, phi, z`.

У виразах часто використовуються частинні похідні. Якщо `u` позначає ім'я розв'язку, наприклад, двовимірного рівняння Пуассона, а `x` і `y` представляють декартові координати на площині, то у виразах для позначення перших і других похідних слід використовувати наступні ідентифікатори: `ux, uy, uxx, uyy, uxy`. Наприклад, ім'я `ux` позначає частинну похідну розв'язку `u` по просторовій координаті `x`, а `uxx` відповідає другий частинній похідній розв'язку по `x`. Якщо просторові координати мають інші імена, наприклад, `r` і `psi`, то тоді потрібно використовувати позначення `ur, upsi, urr, urpsi` і т.д. Коли розв'язок залежить від часу `t`, то можна використовувати частинні похідні по `t`, які позначаються наступним чином: `ut, uxt, uyt, utt, uxxt, uxyt, uyxt,uyyt, uxtt, uytt, uxxtt, uxytt, uyytt`. Ім'я змінної часу `t` зарезервовано і користувач не може його змінити.

Більшість позначень змінних, використовуваних в конкретній моделі, можна дізнатися з вікна, яке відкривається після клацання мишею по кнопках Replace Expression або Insert Expression, розташованих вгорі полів введення виразів.



Частина вмісту цього вікна показана на наступному рисунку.

- Model
  - Component 1
    - Poisson's Equation
      - Gradient of u
        - ux - Gradient of u, x component
        - uy - Gradient of u, y component
      - u - Dependent variable u
  - Geometry
    - Coordinate
      - x - x-coordinate
      - y - y-coordinate
    - dom - Entity index
  - Global Definitions
  - Mesh
- Built In
  - Mathematical functions
    - abs(x) - Absolute value
    - acos(x) - Inverse cosine (in radians)
    - acosh(x) - Inverse hyperbolic cosine (in radians)

Якщо вікно відкривалося після вибору кнопки Replace Expression, то подвійне клацання мишею по елементу цього вікна заміняє вміст поля **Expression**. Вибір елемента цього вікна після вибору кнопки Insert Expression вставляє відповідний текст в поточну позицію курсора в поле **Expression**.

Крім функцій, для моделювання та обчислювання результатів застосовуються спеціальні оператори. Опишемо деякі з них.

- `d(f,x)`, `pd(f,x)` – диференціювання виразу `f` по змінній `x`. Для похідних розв'язку `u` другого порядку можна використовувати різні позначення. Наприклад, `uxx`, `d(ux,x)` і `d(d(u,x),x)` еквівалентні всередині розрахункової області. На межі області `d(u,x)=0`, а `ux` дорівнює середньому по суміжним областям. Оператор `pd` діє аналогічно оператору `d`. Основна відмінність полягає в тому, що `pd(u,x)` дорівнює нулю, а не `ux`, тому, що в `pd` операторі `u` розглядається як незалежна змінна. Наприклад, вирази `d(u^2,u)` і `pd(u^2,u)` обидва повертають `2*u`. Вираз `d(u+x,x)` дорівнює `ux + 1`, а вираз `pd(u+x,x)` дорівнює 1, тому, що `u` розглядається як незалежна змінна. Вираз `d(u,t)` дорівнює `ut`, тоді як `pd(u,t)` дорівнює нулю. Вираз `d(u*v,t)` дорівнює `u*vt+ut*v`, а `pd(u*v,t)` дорівнює нулю.
- `if(condition, expr1, expr2)` – умовний вираз, повертає другий або третій аргумент в залежності від істинності аргументу `condition`. Наприклад, оператор `if(x==0,1,sin(x)/x)` являє неперервний вираз для всіх дійсних `x`, включаючи `x = 0`.
- Оператор `integrate(expr,var,lower,upper)` обчислює інтеграл від виразу `expr` по змінній `var` на відрізку `[lower,upper]`. Можна використовувати п'ятий параметр – відносну похибку обчислення. Вона за замовчуванням дорівнює $1 \cdot 10^{-3}$.
- Оператор підсумовування має наступний синтаксис:
  `sum(expr, index, lower, upper)`
  Наприклад, `sum(i^2,i,1,4)`$=1^2+2^2+3^2+4^2=30$.

- Оператори timeint і timeavg обчислюють інтеграл і інтегральне середнє за часом. Оператор timeint(t1,t2,expr) обчислює інтеграл виразу expr від моменту часу t1 до моменту t2. Інтеграл визначається чисельно. Оператор обчислювання середнього за часом має синтаксис timeavg(t1,t2,expr) і відрізняється від timeint тим, що результат інтегрування ділиться на t2-t1.
- Оператори timemax(t1,t2,expr) і timemin(t1,t2,expr) визначають максимальне і мінімальне значення виразу expr на відрізку часу $t_1 \le t \le t_2$. Ці оператори можна використовувати тільки під час обробки результатів обчислювання, тобто в полях вікон властивостей вузлів гілки **Results**. Якщо вираз expr крім змінної часу t залежить ще від просторових змінних, то максимум і мінімум шукається в кожній точці окремо і оператор повертає просторовозалежний вираз. Наприклад, якщо змінна u залежить від просторової координати r і часу t, то вираз v=timemax(t1,t2,u) повертає максимуми u по часу (t1≤t≤t2) в кожній точці r розрахункового інтервалу.
- Оператор with призначений для доступу до елементів розв'язку під час обробки результатів. В динамічних задачах він дозволяє отримати значення розв'язку на будь якому часовому кроці; при параметричному аналізі за його допомогою можна отримати доступ до розв'язку при фіксованому значенні параметра; в задачах на власні значення він дає доступ до власних функцій, які відповідають потрібним власним числам. Його першим аргументом є номер (моменту часу, параметра або власного числа). Другий аргумент являє ім'я розв'язку або вираз. Наприклад, в задачі на власні значення оператор with(3,u^2) представляє квадрат третьої власної функції.
- Оператор at(t0,sol) дає доступ до часозалежного розв'язку sol в будь який момент часу t0, викорстовуючи, якщо треба, інтерполяцію. Цей оператор можна використовувати при обробці результатів обчислення (в гілці **Results**) і неприпустимо використовувати при побудові моделі. Можна створити змінну моделі, яка містить цей оператор, але використовувати її треба лише на етапі постобробки.
- В системі існує кілька просторових операторів ati(координати,expr) з назвами at0(...), at1(...), at2(...) і at3(...). Вони обчислюють вираз expr в існуючій геометрії на *i* – вимірному примітиві в точці з заданими координатами. Оператор at0 використовується для нульвимірних примітивів (в точках), at1 – одновимірних (на ребрах), at2 – двовимірних (на гранях), at3 – тривимірних (в областях). Наприклад, at1(x_0,y_0,expr) обчислює вираз expr в двовимірній геометрії на ребрі в точці (x_0,y_0). Для коректного використання оператора ati повинен існувати примітив відповідної розмірності в точці з заданими координатами. Наприклад, at0 працює тільки у випадку, коли геометрія містить вершину в точці обчислювання. Майте на увазі, що кількість

координат точки, що передаються оператору `ati`, повинна співпадати з просторовою розмірністю геометрії моделі.

- Оператори `attimemax(t1,t2,expr1,expr2)` і аналогічний йому `attimemin(...)` шукають момент часу t_0 $(t_1 \le t_0 \le t_2)$, в який вираз `expr1` досягає максимуму (мінімуму), і в цей момент t_0 обчислюють вираз `expr2`. Аргументи `t1` і `t2` повинні бути константами, наприклад, вони не можуть обчислюватися за допомогою інших операторів. Оператори `attimemax` і `attimemin` можна використовувати лише на етапі обробки результатів обчислювання.
- Оператор `subst(expr,var1,expr1,...)` робить підстановку. Першим аргументом є вираз `expr`, в якому буде виконуватися заміна. Далі ідуть пари аргументів, кожна з яких складається з імені змінної і значення, яке замість неї слід підставити в початковий вираз. Наприклад, `subst(x^2-y^2,x,0.5,y,0.3)`.

Існує велика кількість інших операторів. З деякими ми познайомимо вас пізніше.

Продемонструємо використання операторів на прикладах. Розглянемо тестову модель і поставимо до неї кілька питань. Хоча більшість з них штучні, вони продемонструють техніку застосування деяких операторів.

Приклад 1. Розв'яжемо рівняння Пуассона в одиничному квадраті $[0,1]\times[0,1]$

$$\Delta u = 16\cdot\left(x^2+y^2\right),$$

з граничними умовами Діріхле:

$$u(x,0)=x^4,\ u(0,y)=y^4,\ u(x,1)=\left(1+x^2\right)^2,\ u(1,y)=\left(1+y^2\right)^2.$$

Після стандартного розв'язання буде поставлено кілька додаткових питань. Щоб на них відповісти, ми створимо змінні моделі, при обчисленні яких будуть використовуватися оператори.

Створіть модель послідовністю команд:

File ->New->Model Wizard->2D->

Mathematics – Classical PDEs - Poisson's Equation(poeq) -> Stationary

В ПВ кореневого вузла **Untitled.mph (root)** введіть:

Unit system - < > = None

Створіть функцію $f(x,y)$, яка буде удавати праву частину рівняння Пуассона.

Home -> **Definitions** -> Functions -> Local -> Analytic

В ПВ створеного вузла **Component 1 – Definitions – Analytic 1** введіть:

Function name = `f`

Definition – **Expression**= `16*(x^2+y^2)`

Definition – **Arguments**= `x,y`

Для контролю побудуйте графік функції, клацнувши мишею по кнопці Plot.

Геометрія моделі складається з одиничного квадрата. Створіть його.

Geometry -> **Pimitives** -> Square

В ПВ створеного вузла **Square 1** залиште значення параметрів за замовчуванням і закінчіть побудову геометрії командою

Geometry – Square 1 (c1) : LMB > Build All Objects
На створеному квадраті побудуйте СкЕ сітку з параметрами за замовчуванням.
Component 1 – Mesh 1 : LMB > Build All
Налаштуйте параметри математичної моделі.
Poisson's Equation (poeq) - Poisson's Equation 1:

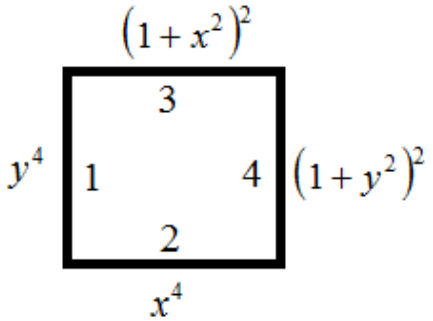
`Diffusion Coefficient` – **c** = `-1`

`Source Term` – **f** = `f(x,y)`

Задайте граничні умови. Оскільки вони на різних ділянках границі різні, то для кожної треба створити свій вузол `Dirichlet Boundary Condition`. Для цього 4 рази виконайте інструкцію:

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Потім у ПВ створених вузлів **Dirichlet Boundary Condition X** (де X=1,2,3,4) по черзі задайте граничні умови, які показано на наступному рисунку.



Наприклад, для лівої сторони квадрата потрібно виконати інструкції:
Dirichlet Boundary Condition 1:

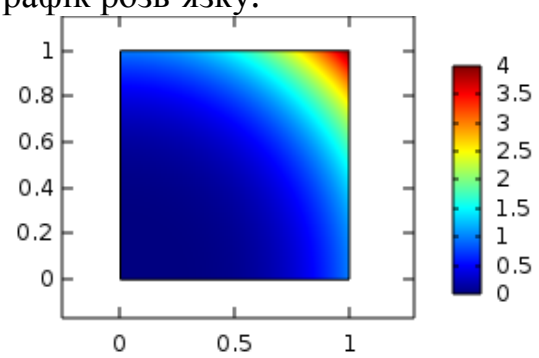
`Boundary Selection` – **Selection** = `1`

`Dirichlet Boundary Condition` – **r** = `y^4`

Після введення граничних умов на всіх чотирьох сторнах квадрата виконайте обчислення.

`Home` -> Compute

В гілці **Results** з'явиться двовимірна графічна група **2D Plot Group 1** з вузлом **Surface 1**, що містить графік розв'язку.



Знайдемо значення розв'язку в окремих точках, наприклад, в одній внутрішній точці, в одній точці на контурі і в одній вершині. Для цього створимо кілька змінних моделі, які будуть містити потрібні значення.

Створимо вузол локальних змінних командою

`Home` -> **`Definitions`** – Variables – Local Variables

Заповніть перші 3 рядка таблиці так, як показано нижче (стовпець **Description** ми замінили більш докладним поясненням).

Component 1 – Definitions – Variables 1:

Name	Expression	Пояснення
up0	at2(0.5,0.3,u)	Значення розв'язку у внутрішній точці $u(0.5,0.3)$
up1	at1(1,0.5,u)	Значення в точці границі $u(1,0.5)$
up2	at0(1,1,u)	Значення в вершині $u(1,1)$

Обновіть обчислення для того, щоб нові змінні стали працездатними.

Study 1: RMB > Update Solution

Надрукуйте значення змінних up0, up1, up2, використовуючи вузли Global Evaluation.

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

Derived Values - Global Evaluation 1:

Expression – **Expression** = comp1.up0

LMB > Evaluate

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

Derived Values - Global Evaluation 2:

Expression – **Expression** = comp1.up1

LMB > Evaluate – Table 1

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

Derived Values - Global Evaluation 3:

Expression – **Expression** = comp1.up2

LMB > Evaluate – Table 1

В таблиці Table 1 будуть надруковані наступні значення змінних.

comp1.up0	comp1.up1	comp1.up2
0.11559	1.5625	4.0000

Припустимо, що нам потрібна змінна, яка містить суму значень розв'язку u в деяких точках, наприклад, розташованих на горизонтальному перерізі квадрата $y=0.5$ в точках $x=0.2,0.4,0.6,0.8$, тобто нас цікавить сума $\sum_{k=1}^{4} u(k/5,0.5)$. Суму можна обчислювати за допомогою оператора sum(...), а значення розв'язку u у внутрішніх точках геометрії обчислюється оператором at2(...).

Створимо змінну моделі us, додавши в таблицю змінних наступний рядок.

Name	Expression	Пояснення
...	...	...
us	sum(at2(k/5,0.5,u),k,1,4)	$\sum_{k=1}^{4} u(k/5,0.5)$

Обновіть розв'язок.

Study 1: RMB > Update Solution

Надрукуйте значення змінної us.

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

Derived Values - Global Evaluation 4:

Expression – **Expression** = comp1.us

LMB > Evaluate – Table 1

В таблицю Table 1 буде додано стовпець us зі значенням 1.4164.

comp1.up0	comp1.up1	comp1.up2	comp1.us
0.11559	1.5625	4.0000	1.4164

Нехай нам потрібно значення інтеграла від розв'язку вздовж горизонтального перерізу квадрата $y = 0.5$, тобто нас цікавить інтеграл $\int_0^1 u(x,0.5)\,d\,x$. Його можна обчислити за допомогою оператора `integrate(...)`.

Створимо змінну моделі `ui`, яка містить потрібне значення, додавши в таблицю змінних наступний рядок.

Name	Expression	Пояснення
...	...	...
ui	integrate(at2(s,0.5,u),s,0,1)	$\int_0^1 u(x,0.5)\,d\,x$

Обновіть розв'язок.

Study 1: RMB > Update Solution

Надрукуйте значення змінної `ui`.

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

Derived Values - Global Evaluation 5:

`Expression` – **Expression** = `comp1.ui`

LMB > Evaluate – Table 1

comp1.up0	comp1.up1	comp1.up2	comp1.us	ui
0.11559	1.5625	4.0000	1.4164	0.42916

Продемонструємо використання оператора підстановки `subst(expr,...)`. Створимо 4 змінні моделі. Змінні `p1` і `p3` будуть містити вирази, в яких буде виконуватися заміна, а `p2` і `p4` – безпосередньо підстановки. Також створимо змінну $p5 = u\left(\sqrt{x}, \sqrt{y}\right)$ з «функціональною» заміною просторових координат.

Додайте в таблицю змінних наступні рядки.

Name	Expression	Пояснення
...	...	...
p1	x^2-y^2	
p2	subst(p1,x,0.5,y,0.8)	$p1(0.5,0.8)$
p3	x^0.25	$\sqrt[4]{x}$
p4	subst(p3,x,u)	$\sqrt[4]{u}$
p5	at2(sqrt(x),sqrt(y),u)	$u\left(\sqrt{x}, \sqrt{y}\right)$

Обновіть розв'язок.

Study 1: RMB > Update Solution

Для обчислення змінної `p2` вчиніть як і раніше, створивши вузол `Global Evaluation 6` (зробіть це самостійно).

comp1.up0	comp1.up1	comp1.up2	comp1.us	ui	p2
0.11559	1.5625	4.0000	1.4164	0.42916	-0.39000

Для перевірки працездатності змінної `p4` побудуємо її графік. Створіть двовимірну графічну групу з вузлом **Surface 1**, що містить графік функціонального забарвлення цієї змінної.

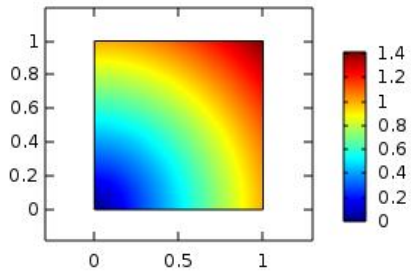
`Results` -> 2D Plot Group

`2D Plot Group 2` **-> Add Plot** – Surface

2D Plot Group 2 – Surface 1:

Expression – **Expression** = `p4`

LMB > Plot



Аналогічно перевіримо працездатність змінної `p5`, побудувавши її графік.

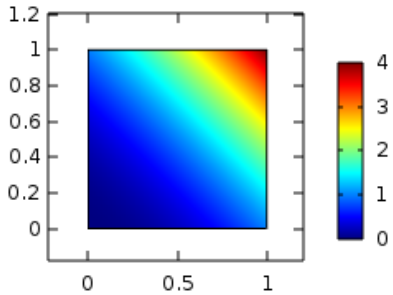
`Results` -> 2D Plot Group

`2D Plot Group 3` **-> Add Plot** – Surface

2D Plot Group 3 – Surface 1:

Expression – **Expression** = `p5`

LMB > Plot



Створимо ще 2 змінні моделі, додавши в таблицю наступні рядки.

Name	**Expression**	**Пояснення**
...	...	...
`p6`	`at2(x,0.75,u)`	$u(x,0.75)$
`p7`	`at2(x,0.7,d(u,x))/2`	$\frac{1}{2}\frac{\partial u}{\partial x}(x,0.7)$

Обновіть розв'язок.

Study 1: RMB > Update Solution

Для перевірки працездатності змінної `p6` побудуємо її графік, зважаючи на те, що вона залежить від однієї змінної x і отримана на горизонтальному перерізі квадрата $y = 0.75$. Створіть джерело даних, виконавши команду

`Results` -> More Data Sets – Grid 1D

Джерела даних типу `Grid` використовуються для обчислення функцій та змінних тоді, коли немає потреби прив'язуватися до СкЕ сітки. У ДД `Grid 1D` розрахунковою областю є інтервал.

Data Sets – Grid 1D 1:

Data – **Source**= `Data set`

Data – **Data set** = `Study 1/Solution 1`

Інші параметри в ПВ створеного вузла **Data Sets – Grid 1D 1** залиште незмінними. Зокрема в полі **Parameter Bounds – Name** залишиться ім'я координати *x* вздовж відрізка [0,1].

Створіть одновимірну графічну групу, і в ній лінійний графік змінної p6.

Results -> 1D Plot Group

1D Plot Group 4 :

Data – Data set = Grid 1D 1

Legends – Show legends = True

Legends – Position = Upper left

1D Plot Group 4-> Line Graph

1D Plot Group 4 - Line Graph 1 :

y-Axis Data – Expression = p6

x-Axis Data – Parameter, Expression = Expression,x (*два поля*)

Legends – Show legends = True

LMB > Plot (*крива* 1 *на наступному рисунку*)

Перевірте працездатність змінної p7 (половина похідної вздовж горизонтального перерізу квадрата $y = 0.7$), розуміючи, що вона також залежить від однієї змінної *x*. Для цього додайте в графічну групу **1D Plot Group 4** лінійний графік змінної p7.

1D Plot Group 4-> Line Graph

1D Plot Group 4 - Line Graph 2 :

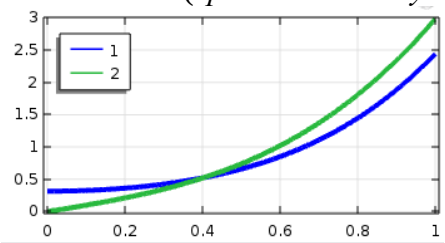
y-Axis Data – Expression = p7

x-Axis Data – Parameter, Expression = Expression, x

Legends – Show legends, Legends = True, Manual

Legends – Legends = 2 (*введення в таблиці легенд*)

LMB > Plot (*крива* 2 *на наступному рисунку*)



Остаточна таблиця змінних зображена на наступному рисунку.

Variables

Name	Expression
up0	at2(0.5,0.3,u)
up1	at1(1,0.5,u)
up2	at0(1,1,u)
us	sum(at2(k/5,0.5,u),k,1,4)
ui	integrate(at2(s,0.5,u),s,0,1)
p1	x^2-y^2
p2	subst(p1,x,0.5,y,0.8)
p3	x^0.25
p4	subst(p3,x,u)
p5	at2(sqrt(x),sqrt(y),u)
p6	at2(x,0.75,u)
p7	at2(x,0.7,d(u,x))/2

Зауваження. Окільки розв'язок задачі відомий $u=\left(x^2+y^2\right)^2$, то ви можете перевірити всі обчислення, створивши глобальну функцію з цим виразом, і виконати з нею всі попередні розрахунки.

■

Перевірку працездатності операторів, пов'язаних зі змінною часу, ми виконаємо в моделі коливання кругової мембрани, яка закріплена на контурі, має параболічну початкову форму і нульову початкову швидкістю. Вона розглядалася в пункті `1.6.3` і була збережена в файл `vibrmembrane.mph`. Відкрийте цей файл і перезбережіть з іншим ім'ям, бо ми будемо вносити в нього багато змін.

Приклад 2. Відкрийте файл `vibrmembrane.mph`. Нагадаємо, що в ньому побудована одновимірна осесиметрична (`1D Axisimmetric`) динамічна (`Time Dependent`) модель. Координата вздовж відрізка, який представляє геометрію, позначається літерою `r`. Це вказано в ПВ вузла **Component 1 (comp 1)** в полях таблиці **`Spatial coordinates`**.

Spatial coordinates

First (r)	Second (phi)	Third (z)
r	phi	z

Більшість операторів, які будуть розглядатися в цьому прикладі, можна використовувати лише на етапі обробки результатів обчислень, тобто в виразах панелей властивостей вузлів гілки **Results**. Можна створити змінні моделі, які містять такі оператори, але тоді їх можна буде використовувати лише на етапі постобробки. Тому ми відразу будемо використовувати оператори у вузлах гілки **Results**.

Побудуємо графік розв'язку в момент часу $t=3$, використовуючи оператор `at(...)`. Виконайте наступні інструкції.

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

2D Plot Group 4: `2D Plot Group 4->` Surface

2D Plot Group 4 – Surface 1:

`Data` – **Data set** = `Revolution 1D 1`

`Expression` – **Expression** = `at(3,u)`

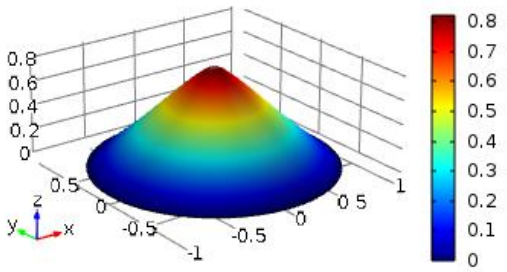
RMB> Height Expression

2D Plot Group 4 – Surface 1 – Height Expression 1:

`Axis` – **Scale factor** = `True, 1`

LMB>Plot

Буде побудований наступний профіль мембрани для моменту часу `t=3`.



Зверніть увагу, що в операторі `at(3,u)` перший аргумент позначає момент часу, в який розглядається розв'язок `u`. Схожим чином працює оператор `with(nummoment,expr)`, але першим аргументом в динамічних задачах він приймає номер моменту часу, а не саме значення часу.

Побудуємо графік розв'язку на 52-му часовому кроці, використовуючи оператор `with (...)`. Виконайте наступні інструкції.

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

Results - 2D Plot Group 5:

`Data` – **Data set** = `Revolution 1D 1`

`2D Plot Group 5->` Surface

2D Plot Group 5 – Surface 1:

`Expression` – **Expression** = `with(52,u)`

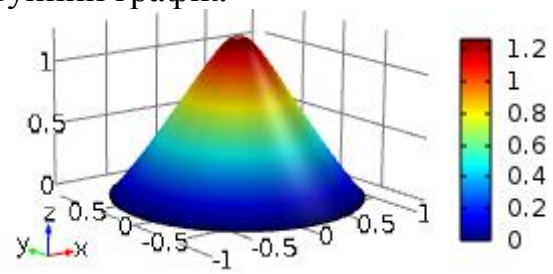
RMB > Height Expression

2D Plot Group 5 – Surface 1 – Height Expression 1:

`Axis` – **Scale factor** = `True, 1`

LMB>Plot

Буде побудований наступний графік.



Щоб дізнатися значення цього 52-го кроку (тобто моменту часу) теж можна застосувати оператор `with(...)`.

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

Derived Values - Global Evaluation 1:

`Expression` – **Expression** = `with(52,t)`

LMB > Evaluate

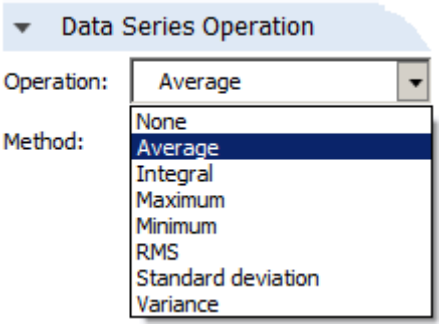
Буде створена таблиця `Table 1` з двома стовпцями: моментами часу і обчисленим значенням `5.1` (для кожного моменту часу).

Time (s)	with(52,t)
0.0000	5.1000
0.10000	5.1000
0.20000	5.1000
0.30000	5.1000
0.40000	5.1000
. . . . .	
9.8000	5.1000
9.9000	5.1000
10.000	5.1000

Тобто 52-му кроку відповідає `t=5.1`.

В `Time Dependent` дослідженні вузол **Global Evaluation** обчислює вираз, заданий в полі **Expression**, для кожного моменту часу, якщо в полі **`Data Series Operation`** – **Operation** обрано значення `None` (значення за замовчуванням). Коли обрати

`Data Series Operation` – **Operation**=`Average`,



то обчислюватиметься і друкуватиметься лише одне значення – середнє по всім моментам часу. Оскільки в нашому випадку `t=5.1` однакове для всіх моментів, то середнє співпадатиме з цим значенням. При цьому в полі **Time selection** повинно стояти `All` або обрано не менше двох моментів часу.

Отже, щоб надрукувати таблицю з однією коміркою (відповіддю `t=5.1`) можна вказати друк середнього значення.

Derived Values - Global Evaluation 1:

`Expression` – **Expression** = `with(52,t)`

`Data Series Operation` – **Operation** = `Average`

LMB > Evaluate – New Table

В інформаційній зоні буде створено таблицю **Table 2** і в ній надруковано шукане значення `5.1`.

Average: with(52,t)
5.1000

Судячи з графіків, на кожному часовому кроці максимальне зміщення досягається в центрі мембрани. Знайдемо момент часу, в який це зміщення максимальне. В системі існує оператор `attimemax(t1,t2,expr1,expr2)`. Він обчислює 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`. Вираз `at1(0,u)` обчислює зміщення центру мембрани, тобто $u(0,t)$. Нагадаємо, що ми маємо одновимірну осесиметричну (`1D Axisimmetric`) модель. Тому обчислення значення розв'язку `u` у внутрішній точці геометрії потрібно обчислювати за допомогою одновимірного оператора `at1`. Вираз `attimemax(0,10,at1(0,u),t)` обчислить значення t_0 $(0 \le t_0 \le 10)$, в який $u(0,t)$ досягає максимуму. Надрукуємо його.

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

Derived Values - Global Evaluation 2:

`Expression` – **Expression** = `attimemax(0,10,at1(0,u),t)`

`Data Series Operation` – **Operation** = `Average`

LMB > Evaluate – Table 2

В таблицю `Table 2` буде додано комірку зі значеннями `5.1117`.

Average: with(52,t)	Average: attimemax(0,10,at1(0,u),t)
5.1000	5.1117

У оператора `attimemax(...)` є опція `'nointerp'`, яка при пошуку максимуму відключає інтерполяцію між часовими кроками. Замініть у вузлі **Global Evaluation 2** вираз і виконайте обчислення.

Expression – **Expression** = `attimemax(0,10,at1(0,u),t, 'nointerp')`

Data Series Operation – **Operation** = `Average`

LMB > Evaluate – Table 2

В таблицю `Table 2` буде додано комірку зі значеннями `5.100`

Average: with(52,t)	Average: attimemax(0,10,at1(0,u),t)	Average: attimemax(0,10,at1(0,u),t, 'nointerp')
5.1000	5.1117	5.1000

Побудувати поверхню мембрани в момент t_0 максимального зміщення центру можна за допомогою оператора `at(t0,u)`, для якого значення t_0 слід взяти з оператора `attimemax(0,10,at1(0,u),t)`. Наступна суперпозиція операторів `at(attimemax(0,10,at1(0,u),t),u)` не працює. Форму мембрани з максимальним зміщенням центру можна було б побудувати по формулі `at(5.1117,u)`, використовуючи в першому аргументі знайдене вище значення часу t_0. Але доцільніше використати вираз `attimemax(0,10,at1(0,u),u)`. Тоді буде знайдено час t_0, в який досягається максимальне зміщення точки `r=0`, і в цей момент буде обчислюватися змінна `u` у всіх точках `r`.

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

2D Plot Group 6:

Data – **Data set** = `Revolution 1D 1`

`2D Plot Group 6`-> Surface

2D Plot Group 6 – Surface 1:

Expression – **Expression** = `attimemax(0,10,at1(0,u),u)`

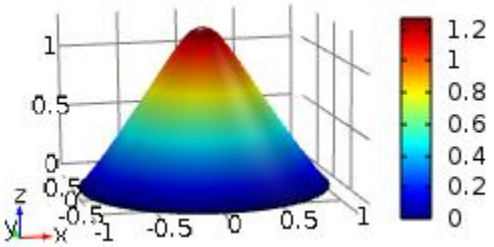
RMB > Height Expression

2D Plot Group 6 – Surface 1 – Height Expression 1:

Axis – **Scale factor** = `True, 1`

LMB>Plot

Буде створено наступний графік, який практично співпадає з побудованим вище на 52-му часовому кроці.



В подальших вправах цього прикладу зручно використовувати одновимірні графіки, які будуватимуться на радіальному перерізі мембрани. Для них потрібно одновимірне джерело даних, але воно (`Cut Line 2D 1`) вже існує в нашій моделі.

Намалюємо графік розв'язку $u(r,t)$ при `t=3` вздовж радіального перерізу.

`Results` -> 1D Plot Group

1D Plot Group 7 :

Data – **Data set** = `Cut Line 2D 1`

`1D Plot Group 7` -> Line Graph

1D Plot Group 7 - Line Graph 1 :

y-Axis Data – **Expression** = `at(3,u)`

x-Axis Data – **Parameter, Expression** = `Expression, r`

LMB > Plot (*верхня крива на наступному рисунку*)

Побудуємо графік похідної $\frac{\partial u}{\partial r}(r,t)$ вздовж радіального перерізу при `t=3` .

`1D Plot Group 7` -> Line Graph

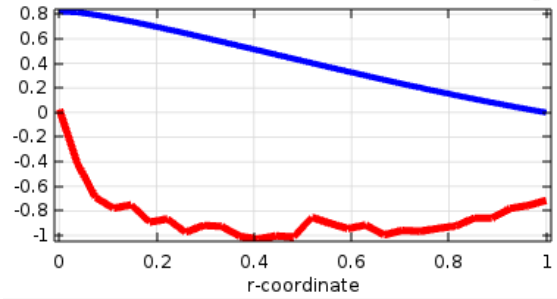
1D Plot Group 7 - Line Graph 2 :

y-Axis Data – **Expression** = `d(at(3,u),r)`

Той же графік буде побудований виразом `d(with(31,u),r)` або `at(3,ur)`

x-Axis Data – **Parameter, Expression** = `Expression, r`

LMB > Plot (*нижня крива на наступному рисунку*)



В системі існують оператори `timemax(t1,t2,expr)` і `timemin(t1,t2,expr)`, які знаходять максимум і мінімум виразу `expr` на вказаному діапазоні часу `t1≤t≤t2`. Якщо вираз `expr` залежить крім змінної часу `t` ще від просторових змінних, то максимум і мінімум знаходиться в кожній точці окремо і повертається просторовозалежний вираз. Наприклад вираз (в умовних позначеннях) `v(r)=timemax(0,10,u(r,t))` повертає максимуми по часу (`0≤t≤10`) в кожній точці `r` розрахункового інтервалу `0≤r≤1`, тобто функцію `r`. Вираз `v(r)` може не співпадати з профілем радіального перерізу мембрани ні в який момент часу. Пояснимо це.

На відрізку `0≤r≤1` побудуємо графіки виразів `at(5.1117,u)` і `timemax(0,10,u)`. Перший з них зображає радіальний переріз мембрани в момент часу `t=5.1117`, коли центральна точка досягає максимального зміщення, а другий – графік максимальних зміщень точок перерізу.

`Results` -> 1D Plot Group

1D Plot Group 8 :

Data – **Data set** = `Cut Line 2D 1`

`1D Plot Group 8` -> Line Graph

1D Plot Group 8 - Line Graph 1 :

y-Axis Data – **Expression** = `at(5.1117,u)`

x-Axis Data – **Parameter, Expression** = `Expression, r`

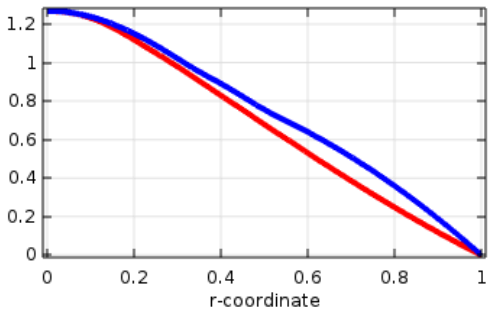
RMB > Duplicate

1D Plot Group 8 - Line Graph 2 :

y-Axis Data – **Expression** = `timemax(0,10,u)`

(той же графік буде побудований виразом `timemax(0,10,at1(r,u))`)

LMB > Plot



Як бачите криві не співпадають. Пояснимо це, побудувавши максимальне зміщення деяких точок мембрани. Виконайте наступні інструкції.

Line Graph 2 : RMB > Duplicate

Line Graph 3 :

y-Axis Data – Expression = `timemax(0,10,at1(0.2,u))`

Цей вираз обчислює максимальне зміщення `u` в точці `r=0.2`, а наступна інструкція зображає його у вигляді точки.

x-Axis Data – Expression = `0.2`

Coloring and Style – Width = `8`

Line Graph 3 : RMB > Duplicate

Line Graph 4 :

y-Axis Data – Expression = `timemax(0,10,at1(0.4,u))`

x-Axis Data – Expression = `0.4`

Line Graph 4 : RMB > Duplicate

Line Graph 5 :

y-Axis Data – Expression = `timemax(0,10,at1(0.6,u))`

x-Axis Data – Expression = `0.6`

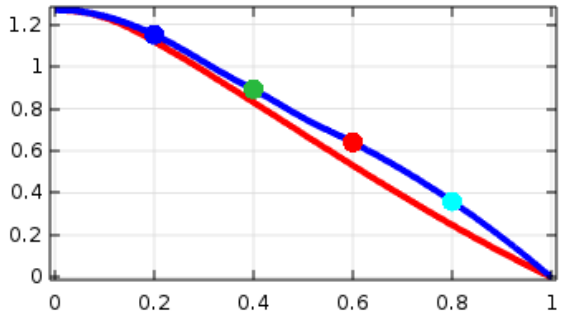
Line Graph 5 : RMB > Duplicate

Line Graph 6 :

y-Axis Data – Expression = `timemax(0,10,at1(0.8,u))`

x-Axis Data – Expression = `0.8`

LMB > Plot



Очевидно, що у кожної точки радіального перерізу є максимальні зміщення, але моменти часу, в які вони його досягають, не співпадають. Тобто максимальне зміщення точок `r=0.2,0.4,0.6,0.8` досягається в різні моменти часу. Крива виразу `timemax(0,10,u)` проходить крізь вказані точки і зображає графік максимальних зміщень точок радіального перерізу мембрани, і не зобов'язана співпадати з графіком перерізу в якийсь фіксований момент часу.

Зауваження. Максимальне зміщення деяких точок можна зобразити швидше, якщо графік виразу `timemax(0,10,u)` побудувати на ДД `Cut Point`, яке являтиме ці точки. Інший спосіб полягає в обчислені зазначеного виразу у вузлах `Point Evaluation`.

Продовжимо ілюструвати роботу операторів.

Оператор `at1(0,u)` обчислює розв'язок в просторовій точці `r=0` і повертає зміщення цієї точки як функцію часу $u(0,t)$. Побудуємо графік цієї функції, тобто графік зміщення центральної точки мембрани в залежності від часу.

`Results` -> 1D Plot Group

1D Plot Group 9 :

`Data` – **Data set** = `Study 1/Solution 1`

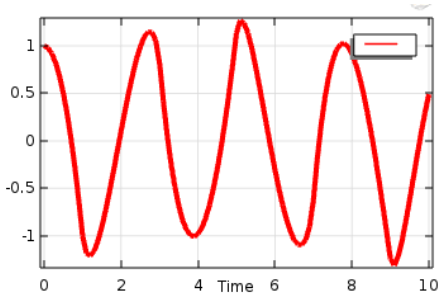
`1D Plot Group 9` -> Global

1D Plot Group 9 - Global 1 :

`y-Axis Data` – **Expression** = `at1(0,u)` (*введення в таблиці*)

`x-Axis Data` – **Parameter** = `Time`

LMB > Plot



Вираз `timemin(0,10,at1(0,u),'nointerp')` обчислює мінімальне зміщення (максимальне вниз) центру мембрани за час $0 \le t \le 10$, не виконуючи інтерполяцію по часу.

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

Derived Values - Global Evaluation 3:

`Expression` – **Expression** = `timemin(0,10,at1(0,u),'nointerp')`

`Data Series Operation` – **Operation** = `Average`

LMB > Evaluate – New Table

В інформаційній зоні створюється таблиця `Table 3`, в якій буде надруковано шукане значення `-1.3109` мінімального зміщення центру.

Average: timemin(0,10,at1(0,u),'nointerp')
-1.3109

Вираз `timeint(0,10,at1(0,u))` обчислює в точці `r=0` інтеграл по часу, тобто $\int_0^{10} u(0,t)\,dt$.

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

Derived Values - Global Evaluation 4:

`Expression` – **Expression** = `timeint(0,10,at1(0,u))`

`Data Series Operation` – **Operation** = `Average`

LMB > Evaluate – Table 3

В таблицю `Table 3` буде додано комірку зі значеннями `-0.38995` вказаного інтеграла.

Average: timemin(0,10,at1(0,u),'nointerp')	Average: timeint(0,10,at1(0,u))
-1.3109	-0.38995

Зверніть увагу на те, що обчислення тривають кілька секунд, бо перед усередненням інтеграл обчислюється у всі моменти часу. Вони будуть тривати швидше, якщо обмежитися двома моментами часу (при одному моменті не спрацьовуватиме опція **Operation** = `Average`).

Вираз `timeavg(0,10,at1(0,u))` обчислює середнє за часом значення зміщення в точці `r=0`, тобто $\frac{1}{10}\int_0^{10} u(0,t)\,dt$.

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

Derived Values - Global Evaluation 5:

`Data` – **Time selection** = `Manual`

`Data` – **Times (s)(1-101)** = `1,2`

`Expression` – **Expression** = `timeavg(0,10,at1(0,u))`

`Data Series Operation` – **Operation** = `Average`

LMB > Evaluate – Table 3

Average: timemin(0,10,at1(0,u),'nointerp')	Average: timeint(0,10,at1(0,u))	Average: timeavg(0,10,at1(0,u))
-1.3109	-0.38995	-0.038995

Вираз `at(3,at1(0.5,u))` обчислює зміщення точки `r=0.5` в момент часу `t=3`.

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

Derived Values - Global Evaluation 6:

`Data` – **Time selection** = `Manual`

`Data` – **Times (s)(1-101)** = `1,2`

`Expression` – **Expression** = `at(3,at1(0.5,u))`

`Data Series Operation` – **Operation** = `Average`

LMB > Evaluate – Table 3

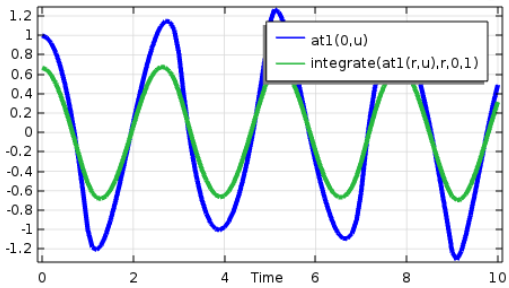
Average: timemin(0,10,at1(0,u),'nointerp')	Average: timeint(0,10,at1(0,u))	Average: timeavg(0,10,at1(0,u))	Average: at(3,at1(0.5,u))
-1.3109	-0.38995	-0.038995	0.41929

Вираз `integrate(at1(r,u),r,0,1)` обчислює інтеграл $\int_0^1 u(r,t)\,dr$, який є функцією часу. Побудуйте його графік, додавши в таблицю виразів вузла **1D Plot Group 9 - Global 1** новий рядок :

`y-Axis Data` – **Expression** = `integrate(at1(r,u),r,0,1)`

LMB > Plot

Самостійно налаштуйте легенду графіка.



Вираз `timeavg(0,0.5,integrate(at1(r,u),r,0,1))` обчислює середнє за часом `0≤t≤0.5` попереднього інтеграла, тобто $\frac{1}{0.5}\int_0^{0.5}\left(\int_0^1 u(r,t)\,d\,r\right)d\,t$.

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

Derived Values - Global Evaluation 7:

`Data` – **Time selection** = `Manual`

`Data` – **Times (s)(1-101)** = `1,2`

`Expression` – **Expression** =

`timeavg(0,0.5,integrate(at1(r,u),r,0,1))`

`Data Series Operation` – **Operation** = `Average`

LMB > Evaluate – Table 3

Average: timemin(0,1	Average: timeint(0	Average: timeavg(	Average: at(3,at1(0	Average: timeavg(0,0.5,integrate(at1(r,u),r,0,1))
-1.3109	-0.38995	-0.038995	0.41929	0.52202

Майте на увазі, що останнє обчислення подвійного інтеграла триває досить довго, і збільшення часового інтервалу `0≤t≤0.5` впливає на це суттєво.

Якщо вираз `expr` в операторі `v=timeavg(t1,t2,expr)` залежить від просторових координат, наприклад, `expr=u(r,t)`, то середнє за часом обчислюється в кожній точці `r` розрахункового інтервалу, і вираз `v` буде функцією `r`. Перевірка цього виконується аналогічно до того, що ми робили з оператором `timemax(0,10,u)` – будували криву і окремі точки. Але тут для побудови точкового графіка ми використаємо точкове джерело даних.

`Results ->` **`Data Set- > More Data Sets ->`** Cut Point 1D

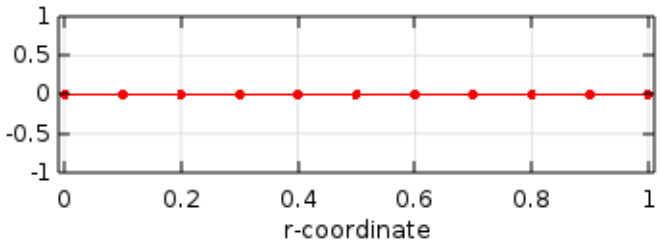
Data Sets - Cut Point 1D 1:

`Data` – **Data set** = `Study 1/Solution 1`

`Point Data` –**Entry method** = `Coordanates`

`Point Data` – **r** = `range(0,0.1,1)`

LMB > Plot



Спочатку побудуємо криву середніх зміщень точок радіального перерізу за інтервал часу `0≤t≤3`.

`Results ->` 1D Plot Group

1D Plot Group 10 :

`1D Plot Group 10 ->` Line Graph

1D Plot Group 10 - Line Graph 1 :

`Data` – **Data set** = `Cut Line 2D 1`

`y-Axis Data` – **Expression** = `timeavg(0,3,u)`

`x-Axis Data` – **Parameter, Expression** = `Expression, r`

`Coloring and Style` – **Color** = `Red`

Потім побудуємо середні зміщення точок, які належать точковому ДД, за той же інтервал часу 0≤t≤3.

1D Plot Group 10 -> Point Graph

1D Plot Group 10 - Point Graph 1 :

Data – **Data set** = Cut Point 1D 1

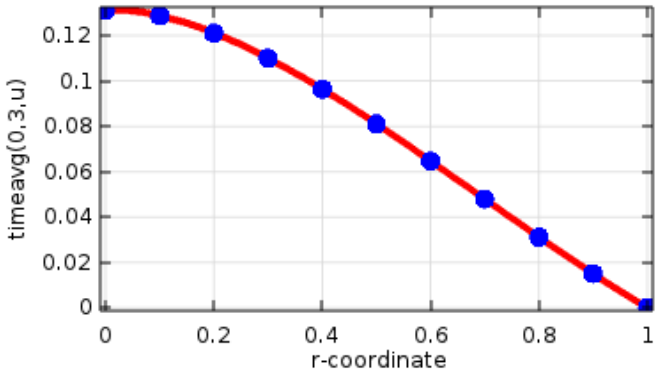
y-Axis Data – **Expression** = timeavg(0,3,u)

Тут вираз timeavg(0,3,u) обчислює середнє за часом 0≤t≤3 зміщення точок, визначених в точковому джерелі даних.

x-Axis Data – **Parameter, Expression** = Expression, r

Coloring and Style – **Color, Width** = Blue, 7 (*два поля*)

LMB > Plot



Отже останній графік зображає середні зміщення точок радіального перерізу за час 0≤t≤3.

Нагадаємо, що звичайний точковий графік можна побудувати, використовуючи вузол Point Graph. Для прикладу побудуємо графік зміщення тих самих точок перерізу мембрани в момент часу t=0.4.

Results -> 1D Plot Group

1D Plot Group 11 :

Data – **Data set** = Cut Point 1D 1

Якщо в полі **Data** – **Times (s)** ввести конкретний момент часу, то в дочірньому вузлі Point Graph буде побудовано положення точок в обраний момент.

Data – **Time selection** = From list

Data – **Times (s)** =0.4

1D Plot Group 11 -> Point Graph

1D Plot Group 11 - Point Graph 1 :

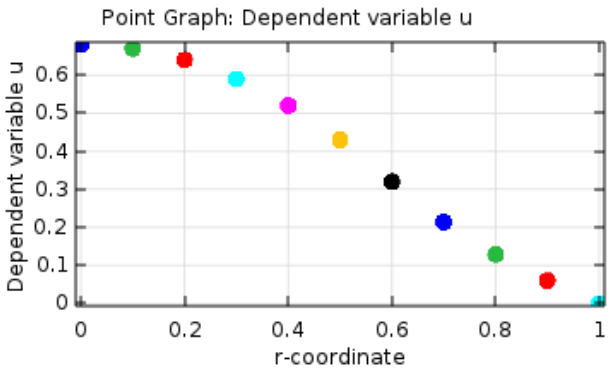
Data – **Data set** = From parent

y-Axis Data – **Expression** = u

x-Axis Data – **Parameter, Expression** = Expression, r

Coloring and Style – **Width** = 6

LMB > Plot



Суттєвим моментом при побудові останніх двох графіків є значення поля **`x-Axis Data`** – **Expression** =`r`.

■

Важливими елементами COMSOL Multiphysics є списки значень. Поля, біля яких стоїть кнопка , вимагають введення списків. Їх можна вводити по-різному:

- використовуючи кнопку списку ;
- використовуючи введення чисел через проміжок або через кому, наприклад, `10, 15, 23, 29.7, 30;`
- використовуючи числа з проміжками-розділювачами всередині фігурних дужок, наприклад, `{0 0.25 0.5 0.75 1};`
- використовуючи список в фігурних дужках в якості аргументу функції, наприклад, `cos({0 pi/6 pi/4 pi/3 pi/2});`
- для введення рівномірно розподілених значень використовують функцію `range(start,step,stop)`, де аргументи `start` і `stop` представляють початкове і кінцеве значення елементів списку, а `step` – інтервал між елементами.

Зазначені способи можна поєднувати. Наведемо кілька прикладів.

1. Запис
   `range(0,0.2,1) 1.5  2.0.`
   обробляється як список `0 0.2 0.4 0.6 0.8 1.0 1.5 2.0.`
2. Запис (дві функції `range` через проміжок)
   `range(0,0.2,1.0) range(2,0.1,2.5)`
   обробляється як об'єднаний список
   `0  0.2  0.4  0.6  0.8  1.0  2.0  2.1  2.2  2.3  2.4  2.5.`
3. Запис `2*range(0,0.2,1.0)`
   призводить до створення списку `0  0.4  0.8  1.2  1.6  2.0.`
4. Запис `2^range(0,1,5)` призводить до створення списку `1 2 4 8 16 32.`

В системі COMSOL Multiphysics важливим елементом синтаксису виразів і функцій є одиниці вимірювання фізичних величин. Більшість функцій користувача і вбудованих математичних функцій використовують безрозмірні аргументи і повертають безрозмірні значення. Тому виклик функцій з аргументом, який має розмірність, в ряді контекстів буде приводити до помилки. Наприклад, це може статися в виразі `sin(t)`, де `t` – змінна часу.

Справа в тому, що змінна `t` зазвичай має розмірність секунди, якщо в моделі використовується система одиниць СІ (SI). Щоб використовувати змінну часу `t` в якості безрозмірного аргументу її слід доповнити розмірністю `[1/s]`. Тоді розмірність `[s]` (секунда) змінної `t` множиться на розмірність `[1/s]` і величина `t[1/s]` стає безрозмірною. В результаті, виклик функції `sin(t[1/s])` буде коректним. При некоректному використанні розмірності COMSOL виділяє вираз помаранчевим кольором.

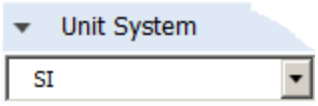
Множення розмірностей має пріоритет над математичними операціями. Наприклад, вираз `1/2[m]` обчислюється як `1/(2[m])` і результатом буде величина `0.5[1/m] = 0.5[m^-1]`. Тоді як `(1/2)[m]` і `1/2*1[m]` представляють величину `0.5[m]=50[sm]`. Аналогічно, якщо `L=2[m]`, то величина `L[1/s]` представляє швидкість `2[m/s]`.

Множення розмірностей можна використовувати при додаванні для отримання сумісних по типу доданків. Наприклад, якщо змінна `rho` представляє густину, то допустимим є вираз `10[kg]+rho[m^3]`. Крім того, множення використовується для з'єднання декількох одиниць, наприклад, значення `4.5[N][m]` еквівалентно `4.5[N*m]=4.5[J]` (`[J]` – джоуль).

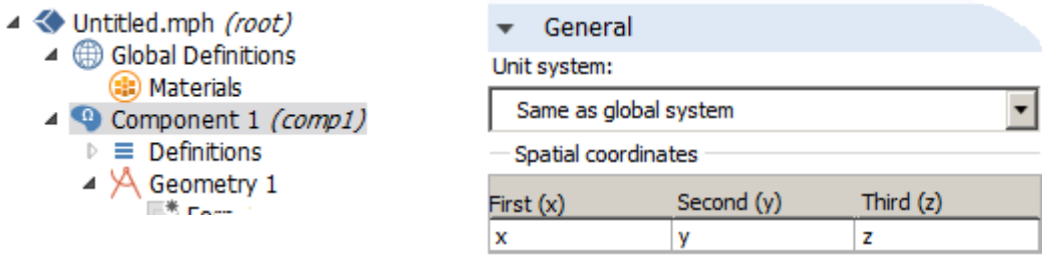
При додаванні двох величин однакової розмірності (по типу), наприклад довжини, але заданих в різних одиницях, система COMSOL перетворює результат в основну систему одиниць. Наприклад, вираз `0.5[m]+25[sm]` перетвориться до значення `0.75[m]`, якщо СІ (SI) є основною системою одиниць моделі.

До будь-якої величини можна додавати безрозмірну константу. Тоді система вважає, що вона (константа) має такий же тип розмірності, як перший операнд, і при цьому її розмірність відповідає основній системі одиниць моделі.

Сама система одиниць моделі визначається в ПВ кореневого вузла. На початку роботи з файлом цей вузол має ім’я **Untitled.mph (root)**. В його ПВ в списку **`Unit system`** можна обрати (або відключити) бажану систему одиниць.



Кожний компонент моделі теж має власну систему одиниць, яка визначається в ПВ вузла **Component** в списку **Unit system** розділу **`General`**. Найчастіше нас задовольняє вибір `Same as global system`, тобто система одиниць компонента така сама, яка обрана для всієї моделі. Тут же задаються імена змінних для просторових координат.



1.8. Довідкова система і документація

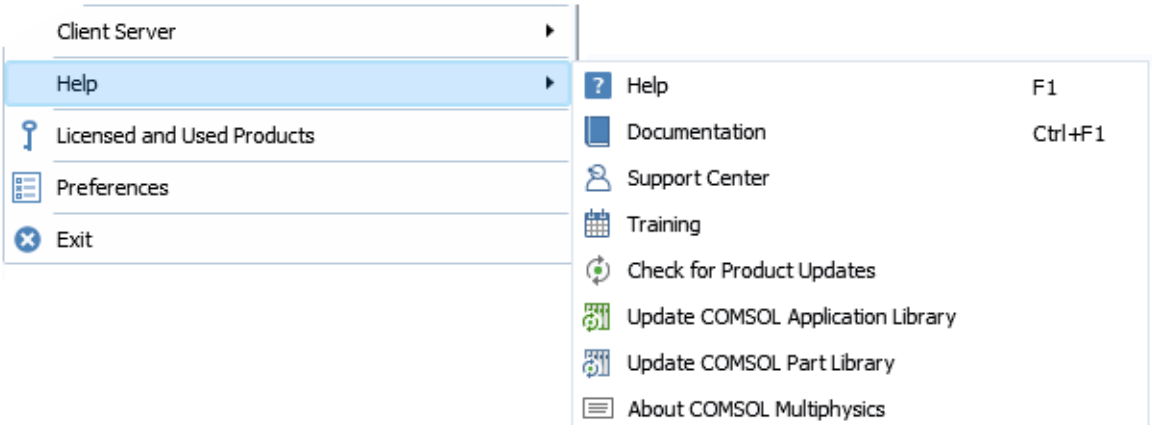
Система COMSOL Multiphysics постачається з великою кількістю електронної документації. Вона встановлюється разом з COMSOL в двох форматах - PDF і HTML. Її ви можете знайти у встановлювальній папці системи COMSOL в каталозі `..\doc`.

Довідкова система відкривається при натисканні мишею на кнопку Help, що розташована в правому верхньому куті вікна програми COMSOL Multiphysics, або натисканням на клавішу `F1` клавіатури.

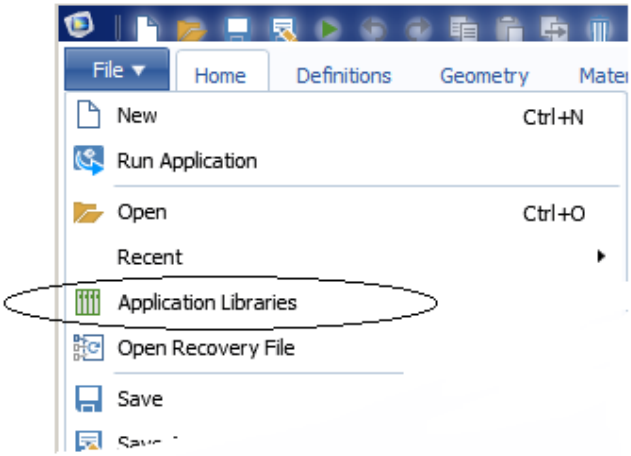
В результаті в правій частині вікна програми відкривається панель довідки з темою, пов'язаною з активним вузлом дерева моделі.

Комбінація клавіш `Ctrl-F1` відкриває окреме довідкове вікно. В ньому міститься список всієї документації COMSOL Multiphysics в форматах HTML і PDF. Це вікно працює як самостійний Browser.

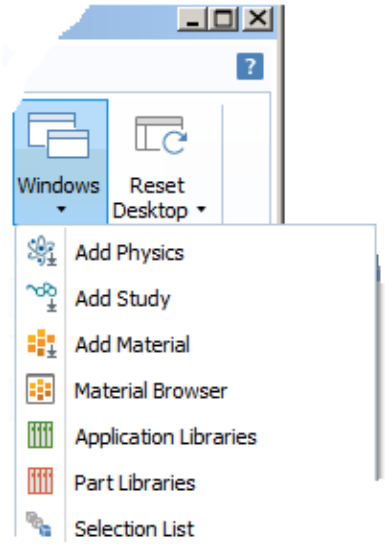
Крім того, в меню `File` є пункт `Help`, який містить ті ж команди роботи з контекстною довідкою (`F1`) і з документацією (`Ctrl-F1`).



Також в меню `File` є пункт Application Libraries, який відкриває бібліотеку готових моделей/прикладів з різних областей фізики і техніки.



Ця команда є також на закладці `Home` стрічки інструментальних панелей в випадному списку `Windows`.



За цією командою відкривається панель `Application Libraries`. Вибір в списку приклада приводить до завантаження `mph` файлу, який ви можете аналізувати, змінювати і виконувати.

Команда Part Libraries відкриває бібліотеку деталей, тобто готових геометричних об'єктів, які при завантаженні в гілку **Geometry** створюють відповідні вузли. Ви можете не креслити свої 3D об'єкти, перерізи, канали і т.д., а використовувати готову геометрію.

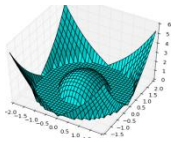
На сайті розробників https://www.comsol.com/models є галерея моделей. Вона містить документацію (зазвичай в форматі PDF) по великій кількості прикладів.

1.9. Задачі для самостійного розв'язання.

Задача 1. Побудуйте графік функції $z = x^2 - y^2$ (поверхню) над квадратною областю $[-2,2]\times[-2,2]$ з круговим отвором $x^2 + y^2 \le 1$.

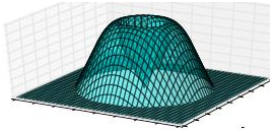
Задача 2. Побудуйте графік функції

$$z = \begin{cases} 0 \quad , & 1 \le x^2 + y^2 \le 4 \\ \sqrt{(1 - x^2 - y^2)(4 - x^2 - y^2)}, & \text{інакше} \end{cases}$$



Задача 3. Побудуйте графік функції

$$f(x,y) = \begin{cases} \sqrt{x^2 + y^2} & , x^2 + y^2 \le 1 \\ 1 + \sqrt{x^2 + y^2} - \frac{1}{2}(x^2 + y^2), & 1 < x^2 + y^2 < 4 \\ 3 - \sqrt{x^2 + y^2}\ , & 4 \le x^2 + y^2 < 9 \\ 0 & , \ x^2 + y^2 \ge 9 \end{cases}$$

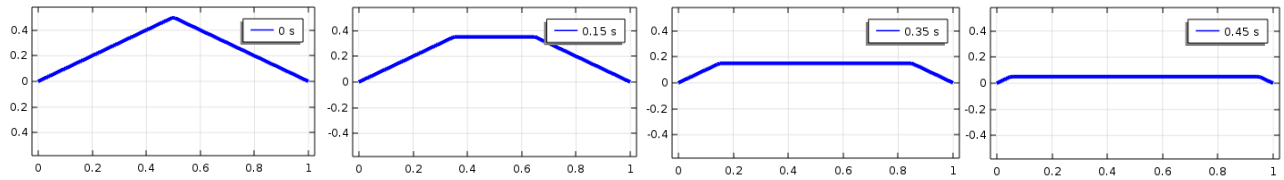


Задача 4. Накреслити тетраедр з вершинами в точках $A(0,0,0)$, $B(2,1,0)$, $C(1,2,0)$, $D(1.3,1.1,2)$. Обчислити об'єм тетраедра, площу грані ABC і повну площу поверхні.

Задача 5. Накреслити вертикальний циліндр радіусом 1 і висотою 5, в якому на висоті 2 зроблено горизонтальний циліндричний отвір радіусом 0.5. Обчислити об'єм створеної фігури.

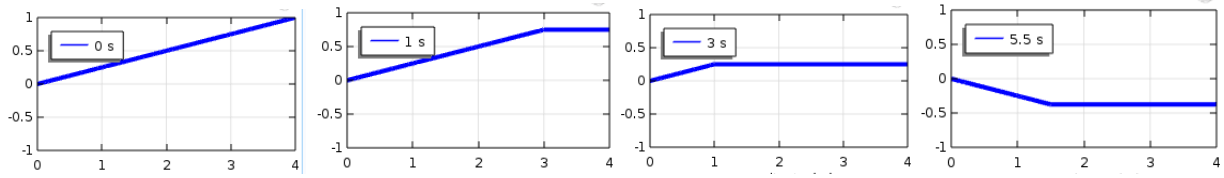
Задача 6. Дослідити коливання скінченної струни довжини L=1 з закріпленими кінцями, початковим зміщенням $u(x,0)=0.5-|x-0.5|$, і нульовою початковою швидкістю $u'_t(x,0)=0$. Покладіть a=1 (швидкість поширення хвиль).

Побудуйте модель, в якій створіть графіки форми струни в різні моменти часу. На наступному рисунку показана струна в початковий і кілька наступних моментів часу.



Задача 7. Дослідити коливання скінченної струни довжини L з одним закріпленим кінцем і іншим вільним. В такій постановці граничні умови мають вигляд $u(0,t)=0, u'_x(L,t)=0$. Покладіть a=1 (швидкість поширення хвиль), L=4, $u(x,0)=x/4$ (початкове зміщення), $u'_t(x,0)=0$ (початкова швидкість).

На наступному рисунку наведені графіки розв'язку `u(x,t)` в моменти часу `t=0,1,3,5.5`.

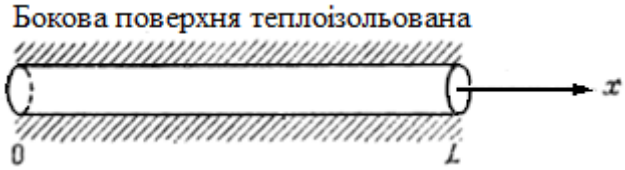


Задача 8. Дослідити вільні осесиметричні коливання кругової мембрани радіусу R=1 (див. параграф 1.6.3), яка закріплена на контурі, має початкову форму $u(r,0)=1-r$ $(0\le r\le 1)$ і нульову початкову швидкість $u'_t(r,0)=0$. В задачі покладіть $a=1$ (швидкість поширення хвиль). Побудуйте анімацію руху лінії перерізу поверхні мембрани вертикальною площиною.

Задача 9. Обчислити і зобразити стаціонарний прогин балки.

Диференціальне рівняння осі зігнутої балки, що знаходиться під дією зовнішніх (поперечних) сил, має вигляд $EJ\,u''''_{xxxx}=q(x)$, де q – зовнішня сила (розрахована на одиницю довжини), E і J – модуль пружності матеріалу балки і момент інерції її перерізу. Ось X збігається з віссю балки. Функція `u(x)` являє вертикальне зміщення точок осі балки, що мали до навантаження горизонтальну координату `x`. Для вільно опертої на краях балки граничні умови мають вигляд $u(0)=0, u(L)=0$, $u''(0)=0, u''(L)=0$, де `L` – довжина балки. В задачі покладіть `L=1`, `E=1`, `J=1`, `q(x)=-1`.

Задача 10. Дослідити процес поширення тепла в тонкому стрижні бокова поверхня якого теплоізольована (тепло може поширюватися тільки вздовж осі).



Задача поширення тепла математично формулюється так: знайти обмежену функцію $u(x,t)$ $(t>0,\ 0\le x\le L)$, яка задовольняє рівнянню теплопровідності

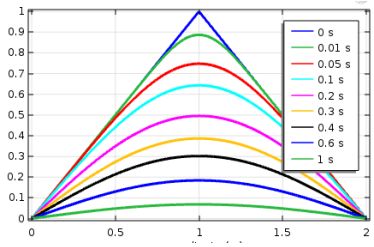
$$\frac{\partial u}{\partial t}=a^2\frac{\partial^2 u}{\partial x^2}\quad (t>0,\ 0<x<L),$$

початковій умові

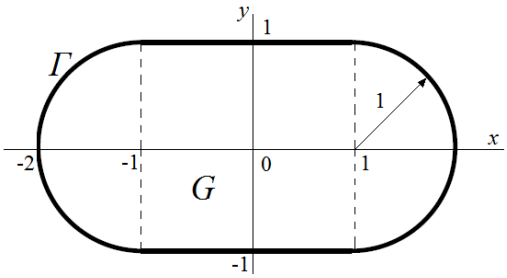
$$u(x,0)=\varphi(x)\ \ (0<x<L),$$

і граничним умовам $u(0,t)=u_0$, $u(L,t)=u_1$, де $u(x,t)$ температура перерізу x стрижня в момент часу t (стрижень тонкий, це означає, що температура точок будь-якого поперечного перерізу x однакова).

Виконати дослідження при L=2, a=1, нульовій граничній температурі $(u(0,t)=0,\ u(L,t)=0)$, і початковій температурі $u(x,0)=\varphi(x)=1-|1-x|$.



Задача 11. Розв'яжіть рівняння Пуассона $u''_{xx}+u''_{yy}=-1$ в області овалу G, який показано на наступному рисунку. Функція $u(x,y)$ задовольняє нульовим граничним умовам Дірихле $u_\Gamma=0$ на його контурі Γ .



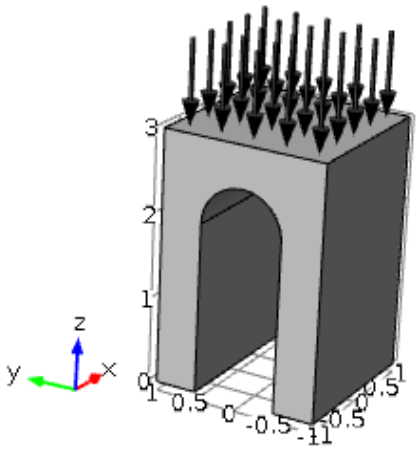
Задача 12. Розв'яжіть рівняння Пуассона $u''_{xx}+u''_{yy}=-4$ в одиничному колі G з однорідними граничними умовам $u_{\partial G}=0$. Побудуйте графік функції $u(x,y)$.

Точним розв'язком є функція $u(x,y)=1-x^2-y^2$. Знайдіть абсолютну похибку чисельного розв'язку, та побудуйте графік різниці двох розв'язків.

Задача 13. Дослідити процес охолодження квадратної пластини $[0,1]\times[0,1]$, контур якої підтримується при нульовій температурі, а початкова температура дорівнює одиниці $(u(x,y,0)=1)$. Виконати дослідження при одиничному значенні температуропровідності.

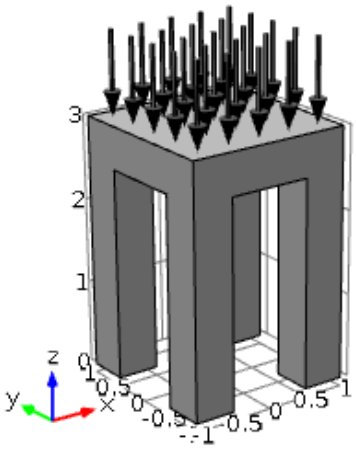
Побудувати графіки: залежності температури від часу в центрі пластини; залежності середньої температури пластини від часу; залежності середньої температури на діагоналі пластини від часу. Побудувати зонди, які відслідковують середню температуру пластини і температуру в фіксованій точці.

Задача 14. Розв'язати задачу деформування тривимірного тіла, яке показано на наступному рисунку. Воно являє бетонний блок $-1 \le x, y, \le 1,\ \ 0 \le z \le 3$ з отвором у вигляді арки, яка розташована вздовж осі X. Радіус арки 0.5 м. Інші параметри геометрії візьміть з рисунка. Основа тіла закріплена. Зверху діє сила $F = 20000\, н/м^2$, спрямована вертикально вниз і рівномірно розподілена по верхній грані.



Побудувати графіки: деформований поверхневий графік напруження σ_z; поверхневий графік напруження σ_z зі схованими фронтальними гранями; зобразити напруження σ_z по двох горизонтальних перерізах, розташувавши їх графіки поруч; зобразити картину навантаження; побудувати деформований лінійний графік величини переміщення на ребрах тіла; показати точки максимуму і мінімуму σ_z на поверхні арки.

Задача 15. Розв'язати задачу деформування тривимірного тіла, яке показано на наступному рисунку. Воно являє бетонний блок $-1 \le x, y, \le 1,\ \ 0 \le z \le 3$ з прямокутними отворами завширшки 1 м і заввишки 2.5 м, які розташовані вздовж координатних осей X та Y. Інші параметри геометрії візьміть з рисунка. Основа тіла закріплена. Зверху діє сила $F = 20000\, н/м^2$, спрямована вертикально вниз і рівномірно розподілена по верхній грані.



Побудувати графіки: деформований поверхневий графік напруження σ_z; поверхневий графік напруження σ_z зі схованими фронтальними гранями; зобразити напруження σ_z по вертикальному діагональному перерізу; зобразити картину навантаження; побудувати деформований лінійний графік величини переміщення на ребрах тіла; показати точки максимуму і мінімуму σ_z на площині $z = 2.5$.

2. Візуалізація та обробка даних.

2.1. Основи графічного зображення даних.

Обговорюючи в попередній главі способи побудови графіків, ми бачили, що гілка **Results - Data Sets** містить вузли, які репрезентують джерела даних (ДД) і використовуються при побудові графіків. Джерелами даних можуть бути розв'язки задач (Solution), одно-, дво- або тривимірні області, просторові криві і поверхні, скінченноелементна (СкЕ) сітка, межа області, січні площини або лінії. Це можуть бути набори даних, отримані деякими перетворенням з інших джерел даних.

При створенні графічного зображення функції, джерело даних фактично визначає зону зміни її аргументів. Джерела даних, такі як `Solution`, крім зони зміни, зазвичай містять розв'язок задачі. «Похідні» джерела даних дозволяють посилатися на дані базового джерела, наприклад, на такі його елементи, як контури границі, січні лінії або площини.

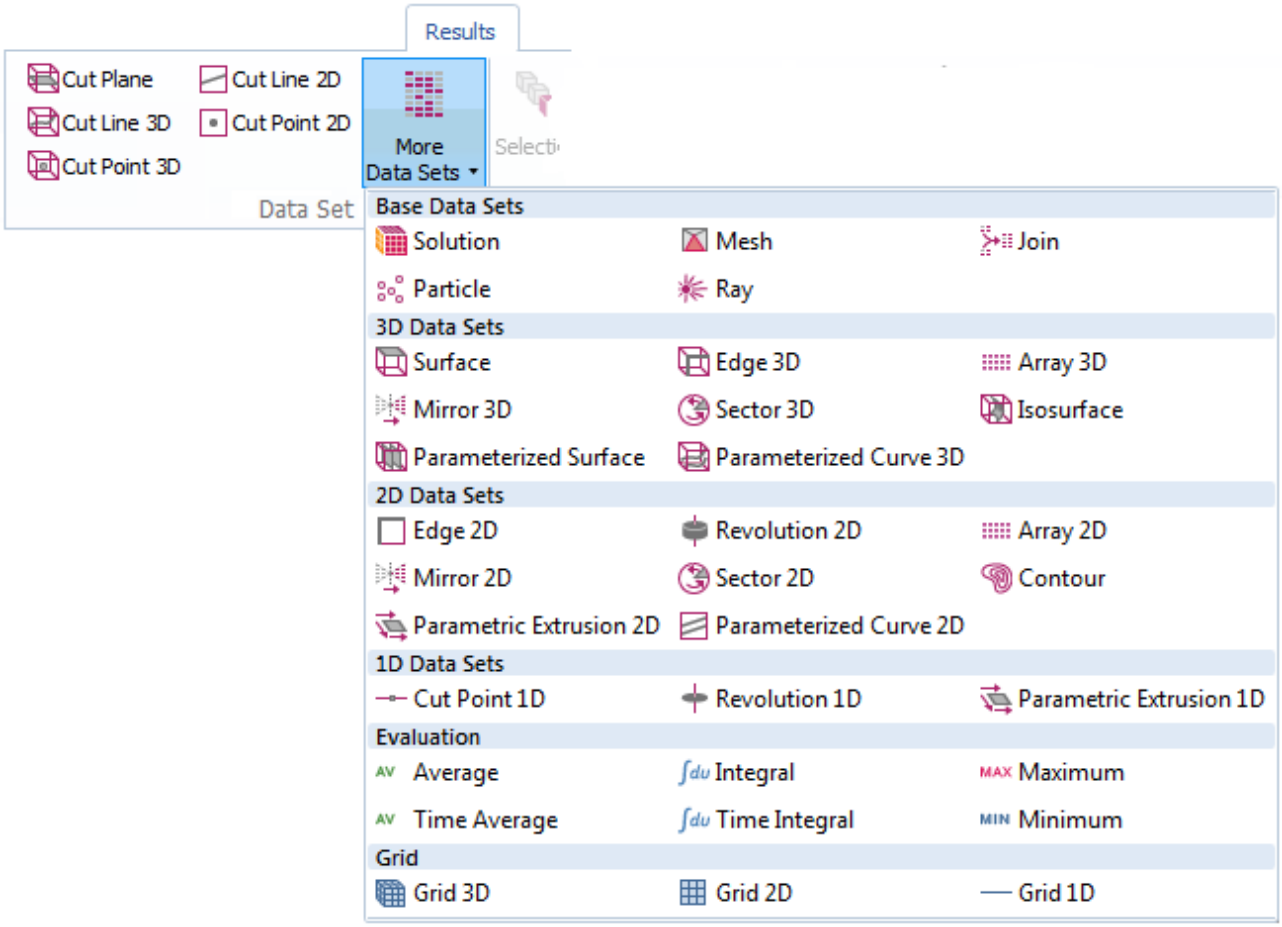
Крім графіків, джерела даних використовуються при обчисленні інтегралів, середніх значень і екстремумів виразів, оскільки цим операціями, крім досліджуваних функцій, треба вказувати зони (множини точок), на яких вони вживаються. Зазвичай такі операції виконуються з використанням вузлів гілки `Derived Values` дерева моделі.

Усі графіки в своєму полі **Data set** повинні посилатися на який-небудь елемент гілки **Results-Data Sets.** Розв'язок задачі (`Solution`), якщо він побудований, є значенням за замовчуванням для джерел даних більшості графіків.

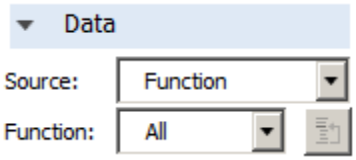
Назви різних типів ДД можна побачити на панелі Results інструментальної стрічки в розділі **Data Set** при розкритому списку **More Data Sets** (див. наступний рисунок).

Джерела даних типу Grid (решітка) використовуються для обчислення функцій в зонах, в яких немає СкЕ сітки, наприклад, при побудові графіків в дальній зоні. Для джерела типу Grid 1D розрахункової зоною є інтервал, а для джерел Grid 2D і Grid 3D розрахунковими областями є прямокутник і прямокутний паралелепіпед. Розміри і положення інтервалу, прямокутника і паралелепіпеда задаються у вікні властивостей (ВВ) відповідних вузлів Grid.

Джерела типу Mesh (сітка) призначені для аналізу розв'язків і побудови графіків функцій на СкЕ сітках, існуючих в моделі. Джерела цього типу використовують також для аналізу самої сітки.

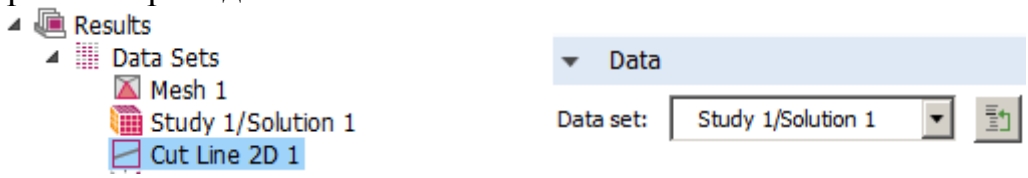


Вікна властивостей більшості вузлів, що репрезентують джерела даних, містять розділ **Data** з полями, які залежать від типу джерела. Так вікна властивостей джерел даних Grid в розділі **Data** мають список **Source**, в якому зазвичай робиться вибір Function, і список **Function**, в якому слід вибрати ім'я глобальної функції або All (всі функції).



Модель, в якій немає інших джерел даних окрім джерела Grid, повинна мати, принаймні, одну глобальну функцію, навіть якщо вона не використовується.

Джерела типу Cut Point представляють набори дискретно розташованих точок. В попередній главі ми використовували джерело Cut Point 2D для обчислення значень розв'язку рівняння Пуассона у внутрішніх точках кругової області. Джерела типу Cut Line зазвичай є лініями перерізу областей. Джерела Cut Plane представляють січні площини тривимірних областей і вживаються аналогічно джерелам Cut Line. Розділ **Data** ПВ вузлів Cut Line і Cut Plane містить список **Data Set**, з якого зазвичай вибирається інше джерело даних, через яке проходить січна.



Джерело Study 1/Solution 1 після завершення обчислень, містить інформацію про СкЕ сітку і значення шуканих величин в її вузлах. Якщо вузлам Cut Line і Cut Plane в списку **Data set** вказати Study 1/Solution 1, то вони «виберуть» з цього джерела потрібну інформацію і надаватимуть її графічним або обчислювальним вузлам, які їх використовуватимуть.

В наступних прикладах ми докладніше опишемо більшість типів джерел даних і пояснимо способи їх використання при побудові графіків. Оскільки ми не готові розв'язувати складні задачі, то деякі типи джерел даних і графіків залишаться поза нашою увагою.

2.1.1. Графіки функцій.

В цьому параграфі буде продемонстровано роботу з джерелами даних Grid, Cut Point 1D, Contour, та графіків Line Graph, Point Graph, Global, Line, Surface, Contour, Slice, Multislice, Isosurface, Scatter Volume, полярних графіків (Polar Plot Group) і графіків в дальній зоні (FarField).

Розглянемо використання ДД типу Grid.

Приклад 1. Створіть шаблон порожньої моделі Blank Model.

File -> New -> Blank Model

Щоб використовувати джерела даних типу Grid (решітка), коли немає інших джерел, модель повинна мати принаймні одну глобальну функцію, навіть якщо вона не буде вживатися.

Global Definitions : RMB > Functions – Analytic

В результаті виконання інструкції створюється вузол і відкривається панель властивостей (ПВ), в якій задається аналітичний вираз для обчислення функції.

Function name = f

Definition – Expression = 1

Definition – Arguments = x

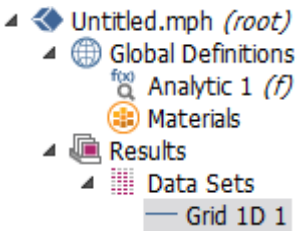
Поле **Function name** містить ім'я функції, в полі **Arguments** через кому перелічуються імена її аргументів, а в полі **Expression** вводиться математичний вираз. Зверніть увагу на те, що в ДМ трохи змінилася назва вузла

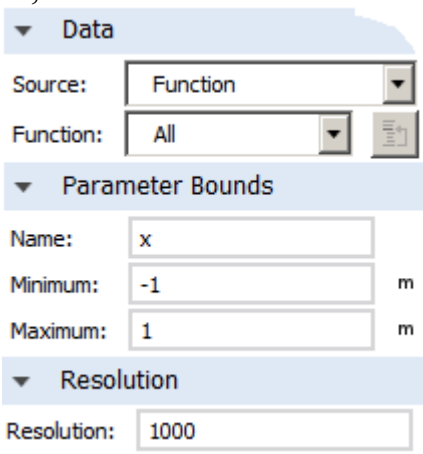
функції – він тепер має позначку **Analytic (f)**. Цю функцію–заглушку ми не будемо використовувати, проте вона потрібна для працездатності вузлів типу `Grid`.

Створіть одновимірне джерело даних типу `Grid` командою

Results - Data Sets: RMB> More Data Sets -> Grid 1D

і заповніть ПВ створеного вузла **Grid 1D 1** так, як показано нижче.





Кінці відрізка визначаються значеннями в полях **Minimum** і **Maximum**. Вміст поля **Resolution** залиште без змін. В ньому вказується кількість інтервалів розбиття відрізка.

Створіть групу одновимірних графіків. В гілку **Results** додайте гілку типу `1D Plot Group` і в ній створіть `Line Graph` вузол.

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

1D Plot Group 1:

`Data Set` – Data = `Grid 1D 1`

RMB > Line Graph

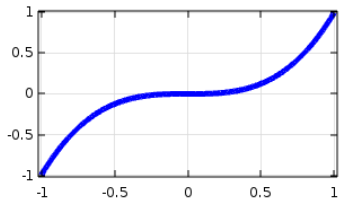
1D Plot Group 1 - Line Graph 1:

`y-Axis Data` – Expression = `x^3`

`x-Axis Data` – Parameter, Expression = `Expression, x` (*два поля*)

В розділі **`Coloring and Style`** задайте стиль, колір і товщину лінії.

LMB > Plot



Як результат, графік кубічної параболи побудовано на відрізку [-1,1], який репрезентується джерелом даних `Grid 1D 1`.

Покажемо, як на графіку можна відобразити окремі точки. Спочатку треба створити «точкове» джерело даних.

`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 = `Coordanates`

`Point Data` – x = `range(-1,0.25,1)`

Потім в одновимірній графічній групі потрібно створити `Point Graph` вузол і в якості його джерела даних використати створений вузол **Cut Point 1D 1**. Для цього активуйте вузол **1D Plot Group 1**, а потім виконайте наступні інструкції:

`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** = `x^3`

`x-Axis Data` – **Parameter, Expression** = `Expression,x`

В розділі **`Coloring and Style`** задайте стиль, колір і розмір точок.

LMB > Plot (*наступний рисунок ліворуч*)

Тут в вираз `x^3` замість аргументу `x` підставлялися абсциси точок ДД **Cut Point 1D 1**.

Коли на графік треба додати одну точку (або кілька зі спільною абсцисою), то для цього можна використати вузол `Global`. Він призначений для побудов графіків функцій, які залежать від часу або якогось непросторового параметра моделі. Але якщо в якості виразів використовуються сталі значення, то будуються точки.

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

1D Plot Group 1 - Global 1:

`y-Axis Data` – **Expression** = `0.5` (*перший рядок таблиці*)

`y-Axis Data` – **Expression** =`-0.5` (*другий рядок таблиці*)

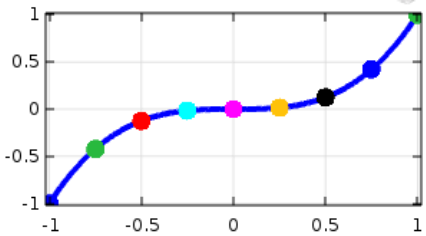
y-Axis Data

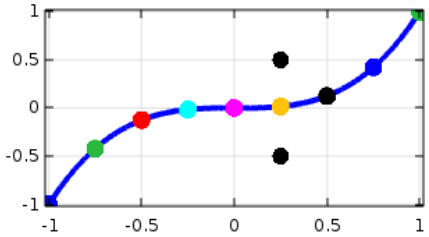
Expression	Unit	Description
0.5	1	
-0.5	1	

`x-Axis Data` – **Parameter, Expression** = `Expression, 0.25`

В розділі **`Coloring and Style`** задайте стиль, колір і розмір точок.

LMB > Plot (*наступний рисунок праворуч*)





Тут для двох точок з ординатами `0.5` і `-0.5` задана спільна абсциса `0.25`.

Розглянемо використання двовимірного джерела даних `Grid 2D`.

`Results` -> Data Set - More Data Sets -> Grid 2D

Results - Data Sets - Grid 2D 2 :

`Data` – **Source, Function** = `Function, All` (*два поля*)

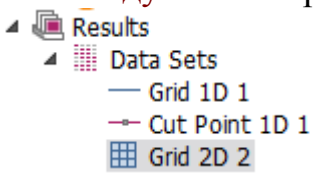
`Parameemter Bounds` –

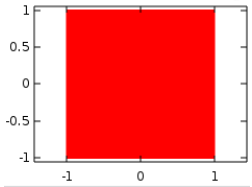
– **First parameter** – **Minimum, Maximum** = `-1,1` (*два поля*)

– **Second parameter** – **Minimum, Maximum**= `-1,1` (*два поля*)

В розділі **`Resolution`** вказуються кількості точок дискретизації прямокутної області по кожній координаті; залишимо їх незмінними.

Виконайте команду Plot і отримайте зображення області джерела даних.





Використовуючи джерело даних `Grid 2D`, можна створити кілька різних графічних зображень функції двох змінних.

Побудуємо графік функції двох змінних у вигляді поверхні.

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

2D Plot Group 2:

`Data` – Data set = `Grid 2D 2`

RMB > Surface

2D Plot Group 2 - Surface 1 :

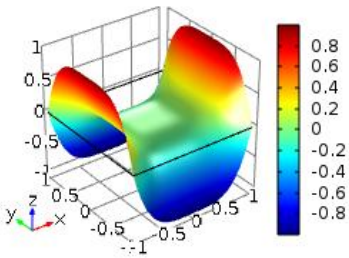
`Expression` – Expression = `x^4-y^4`

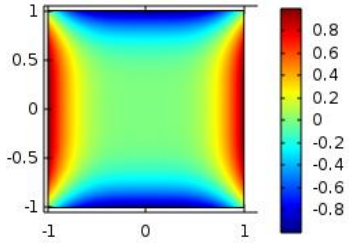
2D Plot Group 2 – Surface 1 : RMB > Height Expression

LMB > Plot (*наступний рисунок в центрі*)

Поверхня зображена на наступному рисунку всередині і побудована над прямокутною областю. Джерело даних **Grid 2D 2** представляє цей прямокутник.

Тепер виділіть вузол **Height Expression 1** і натисніть клавішу `F3` (або з випадного меню виконайте команду Disable). Цим ви деактивуєте вузол **Height Expression 1**, і графічне вікно гілки **2D Plot Group 2** набуде вигляду, зображеного праворуч. Деактивація «вимикає» вузол і скасовує будь-який його вплив на модель. Для повторної активації вузла натисніть клавішу `F4`.





Іноді бажано побудувати графік функції двох змінних не над областю, а над її контуром. Це можна зробити, використовуючи `Line` вузол в двовимірній графічній групі. Він знаходитиме всі одновимірні елементи, створені в двовимірній геометрії (в нашому прикладі на межі ДД `Grid 2D`), і на них буде будувати графік виразу. Кожна точка кривої в 2D області має координати `x`,`y`, і вони будуть використовуватися при побудові графіків функції двох змінних на цій кривій.

Деактивуйте вузол **Surface 1**, натиснувши клавішу `F3`. Цим прибирається зображення з графічного вікна.

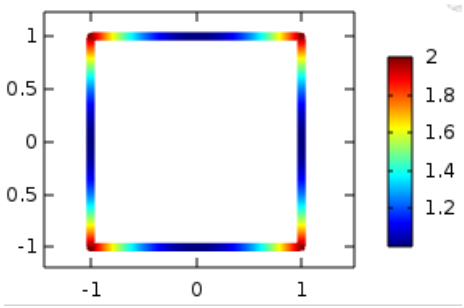
Додайте в гілку вузол `Line` графіка.

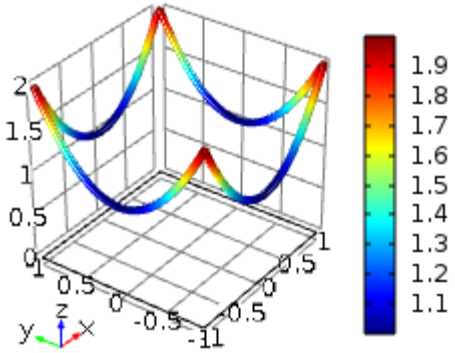
`2D Plot Group 2` -> Add Plot - Line

2D Plot Group 2 - Line 1 :

`Data` – Data set = `From parent`

Expression – **Expression** = x^2+y^2
Coloring and Style – **Line type** =Tube
Coloring and Style – **Tube radius expression** =0.04
Coloring and Style – **Radius scale factor** =True,1
LMB > Plot (*наступний рисунок ліворуч*)
RMB > Height Expression
LMB > Plot (*наступний рисунок праворуч*)

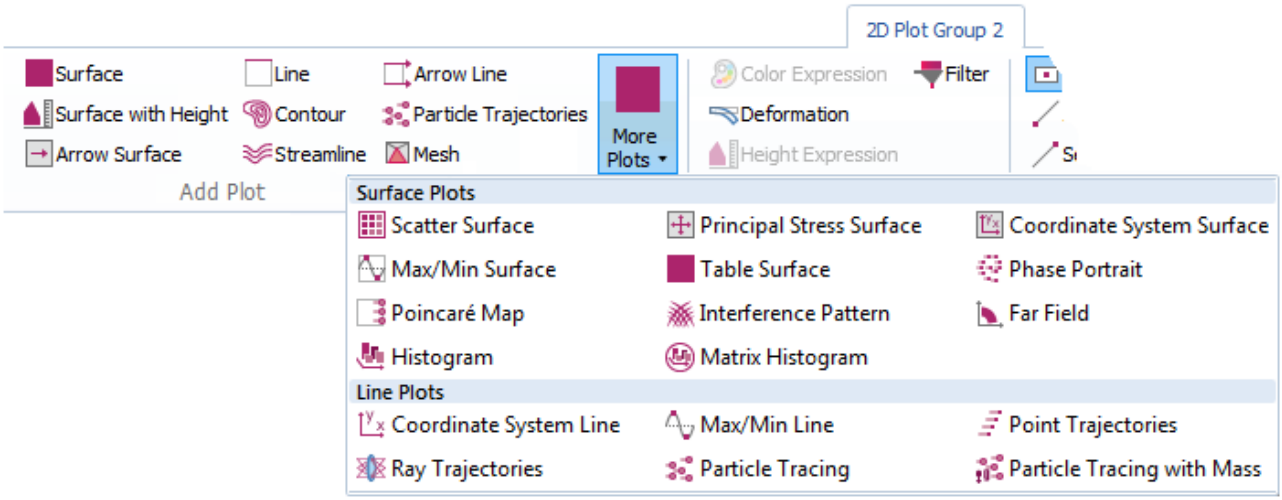




Перед демонструванням інших способів графічного зображення функції двох змінних спочатку деактивуйте останній графік.

2D Plot Group 2 – Line 1 : RMB > Disable

При активному вузлі **2D Plot Group 2** на стрічці буде активна закладка 2D Plot Group 2, що містить, зокрема, команди створення графіків різних типів. Назви всіх доступних графіків можна побачити в розділі **Add Plot** при розкритому списку **More Plots**.



Не всі графіки можуть використовувати джерело даних Grid 2D. Ті, що розташовані в групі **Line Plots**, призначені, головним чином, для побудов на межах двовимірних областей. Частина використовується для зображення векторних полів і деяких «особливих» графіків. Для зображення скалярної функції двох змінних користь становить контурний графік.

В гілці **2D Plot Group 2** створіть графічний вузол типу Contour. Для цього на закладці 2D Plot Group 2 виконайте однойменну команду і задайте властивості графіка (нагадаємо, що вузли **Surface 1** і **Line 1** деактивовано).

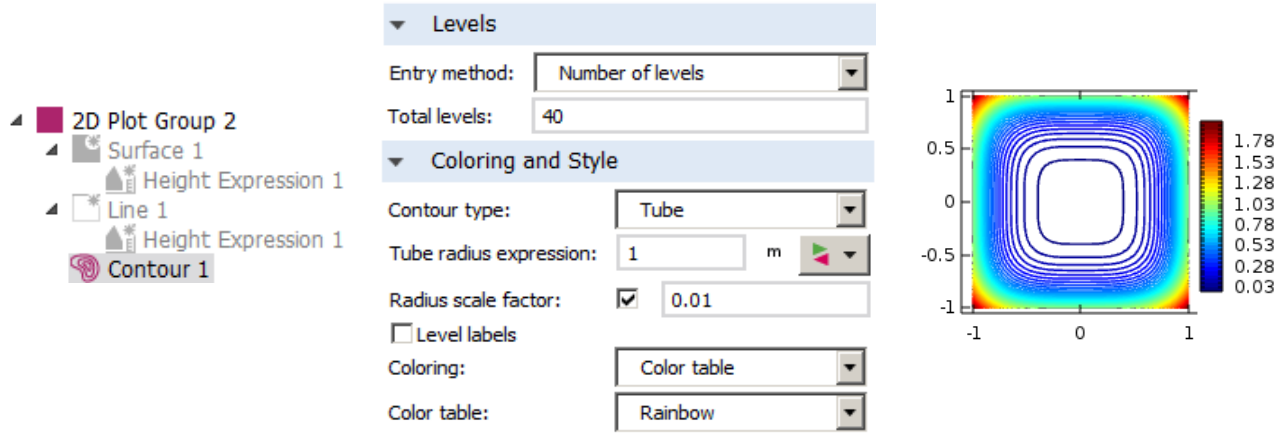
2D Plot Group 2 –> Add Plot - Contour

2D Plot Group 2 - Contour 1 :

Data – **Data set** = `From parent`

Expression – **Expression** = `x^4+y^4`

Решту полів налаштуйте так, як показано на наступному рисунку. Виконання команди Plot призведе до побудови контурного графіка, зображеного праворуч.



У вузол **Contour 1** можна додати вузли, які керують зображенням контурних ліній. Спочатку додайте підвузол `Height Expression`.

2D Plot Group 1 – Contour 1 : RMB > Height Expression

Ви отримаєте зображення контурних ліній у вигляді поверхні (див. наступний рисунок ліворуч).

Деактивуйте вузол **Height Expression 1** і додайте підвузол типу `Color Expression`.

2D Plot Group 2 – Contour 1 – Height Expression 1 : RMB > Disable

2D Plot Group 2 – Contour 1 : RMB > Color Expression

Він накладає додаткове функціональне фарбування на контурний графік. Нове забарвлення відповідає виразу, який задається в ПВ вузла **Color Expression 1** в полі **Expression**. Введіть в цьому полі вираз `sqrt(abs(x^2-y^2))`.

2D Plot Group 2 – Contour 1 – Color Expression 1 :

Expression – **Expression** = `sqrt(abs(x^2-y^2))`

LMB > Plot (*наступний рисунок в центрі*)

Буде побудовано графік, який зображено на наступному рисунку всередині. Забарвлення ліній, яке було раніше, ігнорується, і точки контурних ліній розфарбовуються відповідно до нової функціональної схеми

Деактивуйте вузол **Color Expression 1** і додайте підвузол `Deformation`. З його допомогою можна точки базового графіка змістити в площині джерела даних, на вектор, координати якого задаються в розділі **Expression**.

2D Plot Group 2 – Contour 1 – Color Expression 1 : RMB > Disable

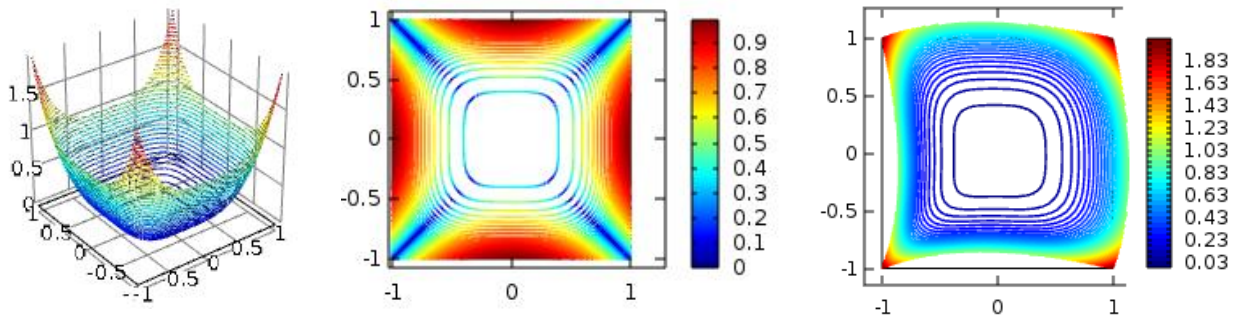
2D Plot Group 2 – Contour 1 : RMB > Deformation

2D Plot Group 2 – Contour 1 – Deformation 1 :

Expression – **x component** = `x^2-y^2`

Expression – **y component** =`-x^2+y^2`

LMB > Plot (*наступний рисунок праворуч*)



Вузол типу `Contour` можна використовувати для побудови кривої, заданої неявним рівнянням. Деактивуйте вузол **Contour 1** і створіть новий вузол **Contour 2**.

2D Plot Group 2 – Contour 1 : RMB > Disable

`2D Plot Group 2` -> Add Plot - Contour

2D Plot Group 2 - Contour 2 :

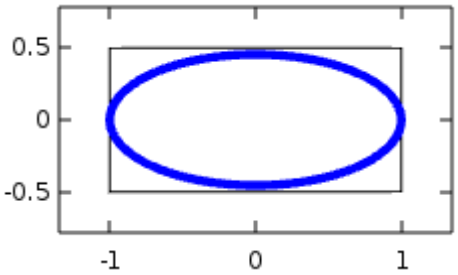
`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `x^2+5*y^2-1`

`Levels` – **Entry method** = `Levels`

`Levels` – **Levels** = `0`

Налаштуйте в розділі `Coloring and Style` тип, товщину і колір кривої. Виконання команди Plot призведе до побудови еліпса.



Тут налаштування вузла **Contour 2** відповідали графіку неявно заданої кривої $x^2+5y^2-1=0$. Ліва частина рівняння була задана в полі **Expression**, а права – в полі **Levels**.

Контурні лінії самі можуть виступати в якості джерела даних для інших графіків.

`Results` -> Data Set - More Data Sets -> Contour

Results - Data Sets – Contour 1 :

`Data` – **Data set** = `Grid 2D 2`

`Expression` – **Expression** = `x^2+y^2`

`Levels` – **Entry method** = `Number of levels`

`Levels` – **Total levels** = `10`

LMB > Plot (*наступний рисунок ліворуч*)

Деактивуйте поточне зображення в графічній групі **2D Plot Group 2** і додайте `Line` графік, джерелом даних якого буде створений **Contour 1**.

2D Plot Group 2 – Contour 2 : RMB > Disable

`2D Plot Group 2` -> Add Plot - Line

2D Plot Group 2 - Line 2 :

Data – **Data set** = Contour 1

Expression – **Expression** = x^2-y^2

Coloring and Style – **Line type** =Tube

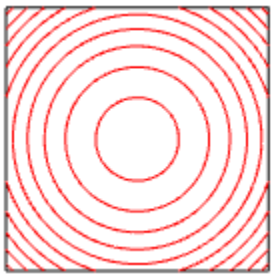
Coloring and Style – **Tube radius expression** =0.03

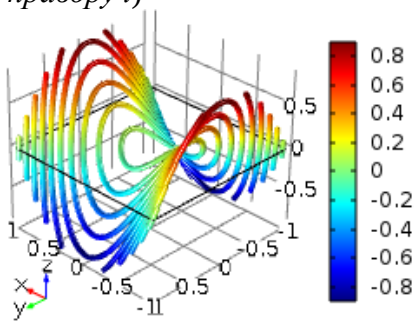
Coloring and Style – **Radius scale factor** =True,1

2D Plot Group 2 – Line 2 : RMB > Height Expression

Line 2 – Height Expression 1 : Axis – **Scale factor** = True, 1 (*два поля*)

LMB > Plot (*наступний рисунок праворуч*)





Тут над лініями, побудованими у вузлі ДД **Contour 1**, будується графік виразу, заданого у графічному вузлі **Line 2**.

Перейдемо до розглядання тривимірних графіків, почавши зі створення тривимірного джерела даних Grid.

Results -> Data Set - More Data Sets -> Grid 3D

Results - Data Sets - Grid 3D 3 :

Data – **Source, Function** = Function, All (*два поля*)

Paramemter Bounds –

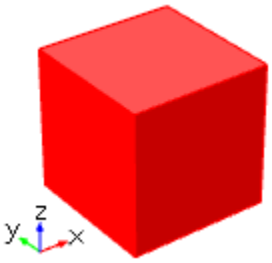
– **First parameter** – **Minimum, Maximum** = -1,1 (*два поля*)

– **Second parameter** – **Minimum, Maximum** = -1,1 (*два поля*)

– **Third parameter** – **Minimum, Maximum** = -1,1 (*два поля*)

В розділі **Resolution** вказується кількість точок дискретизації кубічної області; в нашому прикладі в полях **x resolution**, **y resolution** і **z resolution** залиште значення за замовчуванням 30.

Виконайте команду Plot і отримайте зображення області джерела даних.



Використовуючи джерело даних Grid 3D, можна побудувати кілька різних графічних зображень функції трьох змінних.

Створіть групу тривимірних графіків, і в ній Surface вузол. У 3D випадку він використовується для побудови функціонального забарвлення на поверхні тривимірного джерела даних.

Results -> Plot Group - 3D Plot Group

3D Plot Group 3 : Data – **Data set** = Grid 3D 3

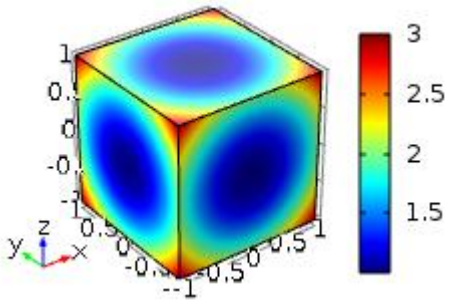
RMB > Surface

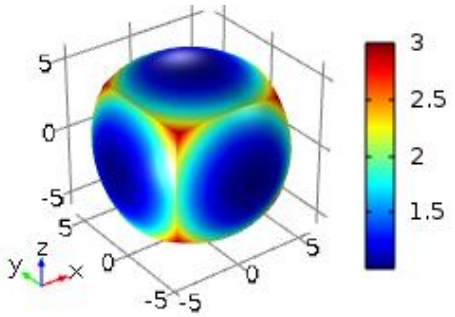
3D Plot Group 3 - Surface 1 :

Data – **Data set** = `From parent`

Expression – **Expression** = `x^2+y^2+z^2`

Виконайте команду Plot, і отримайте графічне зображення виразу у вигляді функціонального забарвлення на поверхні куба, яке показано на наступному рисунку ліворуч. Забарвлення куба залежить від числового значення введеного виразу в точках його граней.





Так само, як і в двовимірному випадку, вузол **Surface 1** може містити підвузол `Deformation`.

3D Plot Group 3 – Surface 1 : RMB > Deformation

3D Plot Group 3 – Surface 1 – Deformation 1 :

В розділі **Expression** вводяться вирази, що представляють компоненти вектора зсуву точок графіка.

Expression – **x component**=`5*x/sqrt(x^2+y^2+z^2)`

– **y component**=`5*y/sqrt(x^2+y^2+z^2)`

– **z component**=`5*z/sqrt(x^2+y^2+z^2)`

В розділі **Scale**, якщо включити відповідний прапорець, в полі **Scale factor** задається масштабний множник для величини зсуву.

Scale – **Scale factor** = `True, 1` (*два поля*)

LMB > Plot (*попередній рисунок праворуч*)

Графічний вузол `Surface` гілки `3D Plot Group` може використовувати джерело даних типу `Grid 2D`. Перевіримо це.

Створіть новий вузол **3D Plot Group 4** і в ньому підвузол **Surface 1**.

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

3D Plot Group 4 :

Plot Settings – **Plot data set edges** = `false`

RMB > Surface

3D Plot Group 4 - Surface 1 :

Data – **Data set** = `Grid 2D 2`

Expression – **Expression** = `0`

Тобто ми, начебто, збираємося зобразити площину `z=0`. Вузлу **Surface 1** додайте підвузол **Deformation 1**. З його допомогою можна задати вектор (векторне поле) зміщення точок базової поверхні.

3D Plot Group 4 – Surface 1 : RMB > Deformation

3D Plot Group 4 – Surface 1 – Deformation 1 :

В розділі **Expression** вводяться координати вектора зміщення.

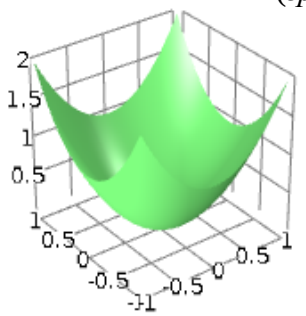
Expression – **x component**=0

– **y component**=0

– **z component**= x^2+y^2

Scale – **Scale factor** = True, 1 (*два поля*)

LMB > Plot (*графік параболоїда*)



Деформувати необов'язково площину z=0. В полі **Expression** вузла **Surface 1** можна ввести будь-який вираз двох змінних *x* і *y*. До речі, імена змінних *x* і *y* визначені в полях **Name** розділу **Parameter Bounds** вузла **Grid 2D 2**. В математичних виразах графічних вузлів, що спираються на це джерело даних, в якості просторових координат ви повинні використовувати тільки ці імена. Аналогічно, у вузлах, що спираються на джерело даних **Grid 3D 3**, слід використовувати імена просторових змінних, визначених в його ПВ.

Продовжимо роботу з вузлами 3D Plot Group. Створіть нову тривимірну графічну групу з підвузлом типу Contour. У тривимірному випадку він використовується для відображення на поверхні тривимірного джерела даних ліній сталого значення функції трьох змінних.

Results –> Plot Group - 3D Plot Group

3D Plot Group 5 : RMB > Contour

3D Plot Group 5 - Contour 1 :

Data – **Data set** = From parent

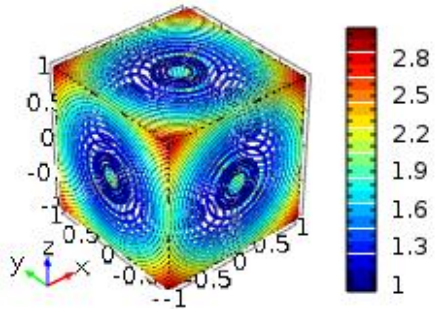
Expression – **Expression** = x^2+y^2+z^2

Levels – **Entry method** = Levels

Levels – **Levels** = 1 1.025 1.05 1.075 range(1.1,0.1,3)

Список, заданий в полі **Levels**, відповідає правим частинам рівняння $x^2 + y^2 + z^2 = C$. Стала *C* по черзі набуває значення з цього списку, і для кожного значення будується крива на поверхні ДД.

Виконайте команду Plot і отримайте наступний графік



Створіть нову тривимірну графічну групу з підвузлом типу `Slice`. Він використовується для зображення функції трьох змінних у вигляді функціонального забарвлення декількох паралельних плоских шарів, розташованих всередині тривимірного джерела даних.

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

3D Plot Group 6 : RMB > Slice

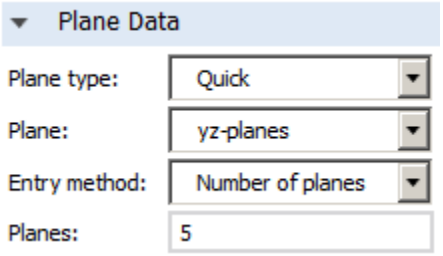
3D Plot Group 6 - Slice 1 :

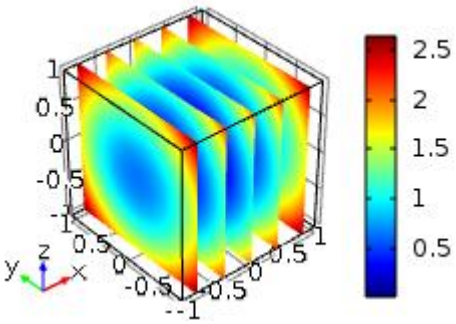
`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `x^2+y^2+z^2`

Розділ **`Plane Data`** використовується для управління шарами. Заповніть його так, як показано на наступному рисунку ліворуч. Вибір значення `yz-planes` в списку **Plane** визначає орієнтацію шарів: вони будуть паралельні площині `yz`.

Виконайте команду Plot і отримайте наступний графік.





Створіть нову тривимірну графічну групу з підвузлом типу `Multislice`. Він використовується для графічного зображення функції трьох змінних у вигляді функціональної забарвлення трьох сімей плоских шарів, розташованих в межах тривимірного джерела даних.

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

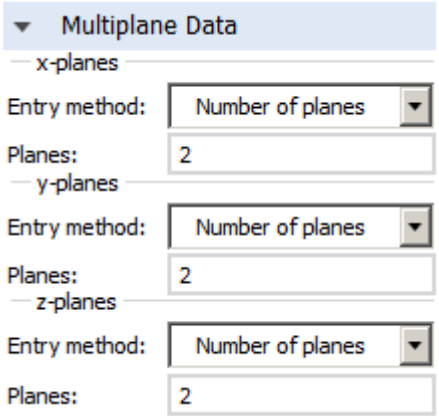
3D Plot Group 7 : RMB > More Plots - Multislice

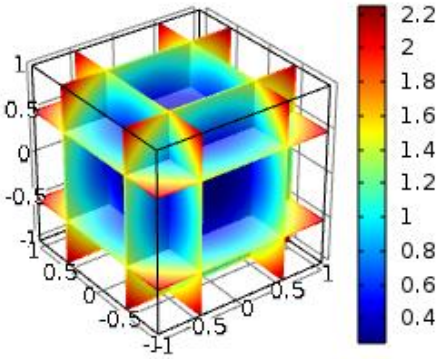
3D Plot Group 7 - Multislice 1 :

`Expression` – **Expression** = `x^2+y^2+z^2`

Розділ **`Multiplane Data`** використовується для управління шарами. Заповніть його так, як показано на наступному рисунку ліворуч.

Виконайте команду Plot і отримайте наступний графік.





Створіть нову тривимірну графічну групу з вузлом `Isosurface`. Він використовується для графічного зображення поверхонь сталого значення

$F(x,y,z)=Const$ функції трьох змінних. В ПВ такого вузла в полі **Expression** задається вираз $F(x,y,z)$, а в розділі **Levels** вводяться значення сталих, яким він буде дорівнювати. Якщо ввести одне значення, наприклад, 1, то буде побудована поверхня за рівнянням $F(x,y,z)=1$.

Results –> Plot Group - 3D Plot Group

3D Plot Group 8 : RMB > Isosurface

3D Plot Group 8 - Isosurface 1 :

Expression – **Expression** = x^2+y^2+z^2

Levels – **Entry method** = Levels

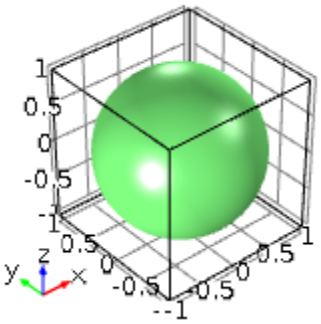
Levels – **Levels** = 1

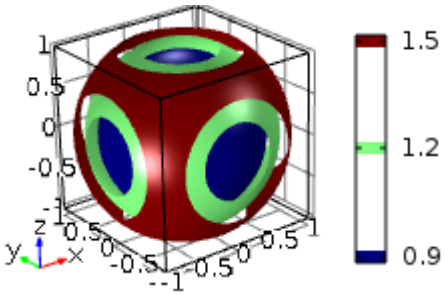
LMB > Plot (*наступний рисунок ліворуч*)

Якщо в полі **Levels** ввести кілька значень, наприклад, 0.9,1.2,1.5, то буде побудовано кілька поверхонь рівня.

Levels – **Levels** = 0.9,1.2,1.5

LMB > Plot (*наступний рисунок праворуч*)





Познайомимося з графіком типу Scatter Volume (графік розсіювання). Він складається з множини точок (або невеликих кульок), розсіяних в області джерела даних і розфарбованих у відповідності зі значенням скалярної функції трьох змінних.

Створіть нову тривимірну графічну групу з вузлом типу Scatter Volume.

Results –> Plot Group - 3D Plot Group

3D Plot Group 9 : RMB > More Plots - Scatter Volume

3D Plot Group 9 - Scatter Volume 1 :

Expression – **x component**=x
– **y component**=y
– **z component**=z

Evaluation Points – **x grid points - points** =10
– **y grid points - points** =10
– **z grid points - points** =10

Поля розділу **Expression** визначають вирази, в напрямку яких виконується розміщення точок. Зазвичай це x,y,z, але можуть бути і інші вирази. Згадайте, що джерело даних **Grid 3D 3**, яке використовується вузлом **3D Plot Group 10**, являє куб $[-1,1]\times[-1,1]\times[-1,1]$. Тоді *x*-координати розсіяних точок будуть визначатися десятьма значеннями виразу, який введено в полі

x component. Те ж саме стосується двох інших координат. В нашому випадку точки будуть рівномірно розподілені по кубу.

В розділі **Radius** в полі **Expression** можна ввести функцію, яка буде визначати радіуси кульок-точок.

Radius – **Expression** =1

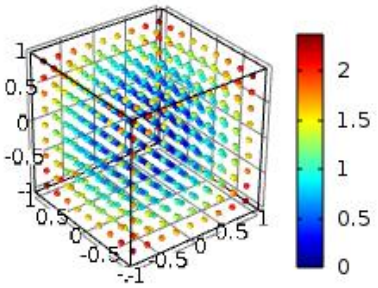
Radius – **Radius scale factor** = True, 0.03 (*два поля*)

Введення в полі **Radius** – **Expression** одиниці каже про те, що радіуси кульок будуть незмінними.

Поле **Color** – **Expression** використовується при фарбуванні кульок-точок.

Color – **Expression** = x^2+y^2+z^2

Виконайте команду Plot і отримайте наступний графік.



Деактивуйте вузол **3D Plot Group 10 - Scatter Volume 1** і додайте інший підвузол типу Scatter Volume.

3D Plot Group 9 - Scatter Volume 1 : RMB > Disable

3D Plot Group 9 –> Add Plot - More Plots - Scatter Volume

3D Plot Group 9 - Scatter Volume 2 :

Data – **Data set** = From parent

Expression – **x component**=x

– **y component**=y

– **z component** = sqrt((x^2+y^2<1)*(1-x^2-y^2))

Evaluation Points – **x grid points - points** =30

– **y grid points - points** =30

– **z grid points - points** =1

Radius – **Expression** =1

– **Radius scale factor** = True, 0.03 (*два поля*)

Color – **Expression** =1

Coloring and Style – **Coloring, Color** = Uniform,Blue (*два поля*)

LMB > Plot (*наступний рисунок ліворуч*)

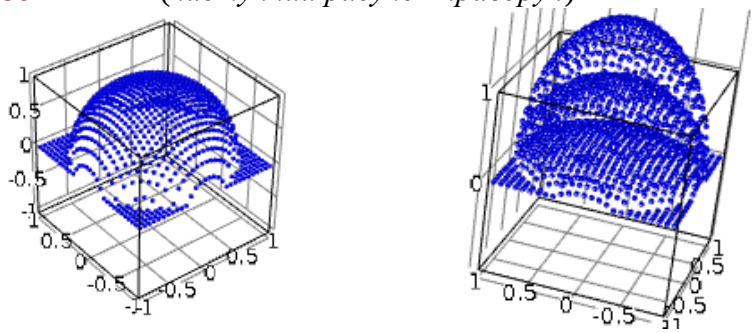
Зверніть увагу на те, що значення в полі **Evaluation Points** – **z grid points – points** ніяк не впливає на зображення. Справа в тому, що це число задає кількість точок розбиття діапазону зміни z координати в виразі **Expression** – **z component**, який в нашому прикладі не залежить від z. Якщо, наприклад, в цьому полі до виразу додати множник 1+z, то вплив з'явиться. Внесіть в ПВ вузла **Scatter Volume 2** наступні зміни.

Expression – **z component** =

sqrt((x^2+y^2<1)*(1-x^2-y^2))*(1+z)

Evaluation Points – **z grid points - points** =3

LMB > Plot (*наступний рисунок праворуч*)



Зауваження. Аналогічно застосовуються Scatter Surface графіки, які використовуються в двовимірних графічних групах.

■

Зауваження. Скоріш за все на вашому комп'ютері колір фону тривимірних графіків буде блакитний, а не білий, як у нас. Цей колір визначається значеннями за замовчуванням параметра системи Dcs.canvascolor, який дорівнює 255,255,120,255,255,204. Тут перші три числа визначають RGB колір верхньої частини графічного полотна, а останні три числа задають колір його нижньої частини. Щоб полотно було білим, всі шість чисел слід задати рівними 255. Для цього на робочому столі Windows відкрийте вікно властивостей ярлика COMSOL Multiphysics і на закладці Ярлик відредагуйте в полі Об'єкт командний рядок наступним чином (без перенесення рядка): "C:\Program Files\COMSOL\COMSOL51\Multiphysics\bin\win64\comsol.exe" -Dcs.canvascolor=255,255,255,255,255,255. При наступному запуску COMSOL Multiphysics фон графіків буде білим.

Полярною системою координат називається двовимірна система координат, в якій положення точки на площині визначається двома числами - полярним кутом θ і полярним радіусом r. Рівняння кривих в полярних координатах записують у вигляді $r = r(\theta)$.

В COMSOL для побудови полярних графіків використовується Polar Plot графічна група, в яку звичайним чином додаються вузли лінійних графіків. Особливістю цієї групи є те, що центральна точка полярного графіка відповідає мінімальному значенню радіуса (яке обчислюється автоматично). Зокрема припустимі від'ємні значення, тоді нульовому радіусу може відповідати коло всередині графічної зони. Навпаки, центральна точка графіка може відповідати додатному значенню мінімального радіуса. Якщо мінімальне значення радіуса не нуль, то графік може ввести вас в оману. Пояснимо це на прикладах.

Приклад 2. Графік кривої $r = r(\theta)$ в полярних координатах.

Створіть шаблон порожньої моделі Blank Model.

File -> New -> Blank Model

Для роботи з решітками, тобто з ДД Grid, знадобиться функція-заглушка. Створіть її.

Home –> Definitions – Functions –> Global – Analytic

Function name = f

Definition – **Expression, Arguments** = 1, theta (*два поля*)

Для використання на полярному графіку незалежного аргументу θ знадобиться одновимірне джерело даних Grid 1D. Створіть його.

Results –> Data Set - More Data Sets –> Grid 1D

Data Sets - Grid 1D :

Data – **Source, Function** = Function, All (*два поля*)

Paramemter Bounds – **Name** = theta (*ім'я полярної змінної*)

Paramemter Bounds – **Minimum, Maximum** = 0,2*pi (*два поля*)

Створіть Polar Plot графічну групу і в ній вузол Line Graph лінійного графіка.

Results –> Plot Group - Polar Plot Group

Polar Plot Group 1:

Data Set – **Data** = Grid 1D 1

RMB > Line Graph

Polar Plot Group 1 - Line Graph 1:

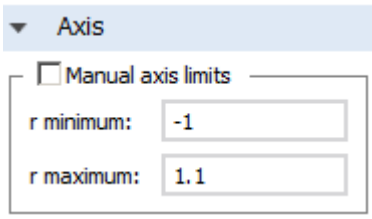
r-Axis Data – **Expression** = cos(theta)

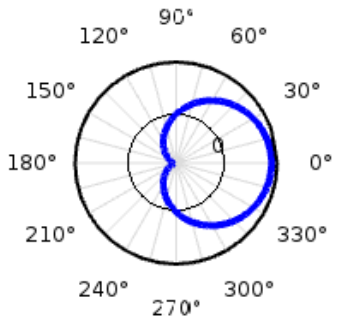
θ-Angle Data – **Parameter** = Expression

θ-Angle Data – **Expression** = theta

В розділі **Coloring and Style** задайте стиль, колір і товщину лінії.

LMB > Plot (*наступний рисунок праворуч*)





Ми звикли, що рівняння $r = \cos\theta$ в полярній системі координат репрезентує коло (зсунуте праворуч від полюса). Якщо придивитися до графіка, то побачимо, що всередині з позначкою 0 намальовано коло. Воно розділяє зони від'ємних і додатних значень радіуса r. Дійсно, в розділі **Axis** вузла графічної групи **Polar Plot Group 1** вказано мінімальне значення −1 (попередній рисунок ліворуч). Саме воно відповідає центральній точці графіки.

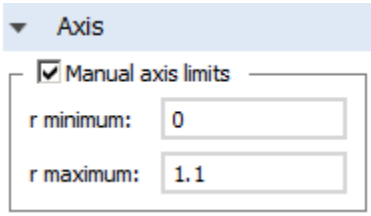
Якщо відкинути точки графіка з від'ємними полярними радіусами, то вийде звичайний полярний графік кривої $r = \cos\theta$ (коло). Для цього в полі **Axis** – **r minimum** треба встановити нуль.

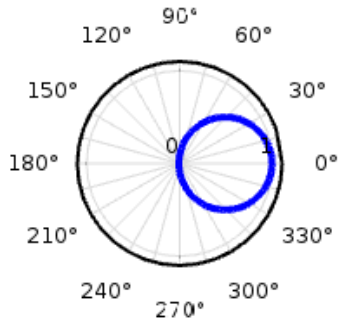
Polar Plot Group 1:

Axis – **Manual axis limits**=True

Axis – **r minimum**=0

LMB > Plot (*наступний рисунок праворуч*)





За вказаною особливістю потрібно стежити, інакше і графіки з позитивними полярними радіусами приводитимуть до непорозуміння. Для пояснення побудуємо в новій полярній графічній групі графік кривої $r = 1 + \cos(10\,\theta)/10$.

`Results` -> Plot Group - Polar Plot Group

Polar Plot Group 2 : **`Data Set`** – **Data** = `Grid 1D 1`

RMB > Line Graph

Polar Plot Group 2 - Line Graph 1:

`r-Axis Data` – **Expression** = `1+cos(10*theta)/10`

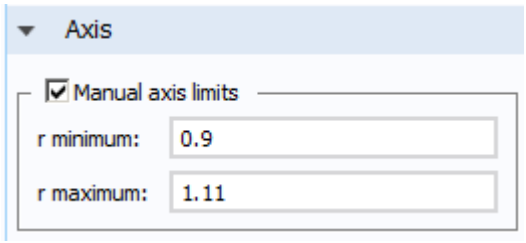
`θ-Angle Data` – **Parameter** = `Expression`

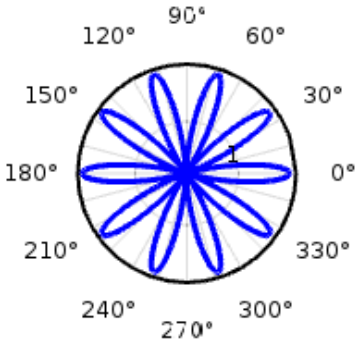
`θ-Angle Data` – **Expression** = `theta`

В розділі **`Coloring and Style`** задайте стиль, колір і товщину лінії.

LMB > Plot (*наступний рисунок праворуч*)

Ліворуч графіка показано розділ **`Axis`** ПВ вузла **Polar Plot Group 2**. З нього зрозуміло, що центральній точці графіка відповідає $r = 0.9$.





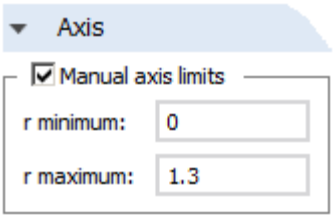
Щоб побачити правильний графік потрібно в графічній групі в полі **`Axis`** – **r minimum** встановити нуль. Для «краси» ми ще змінимо і максимальне значення.

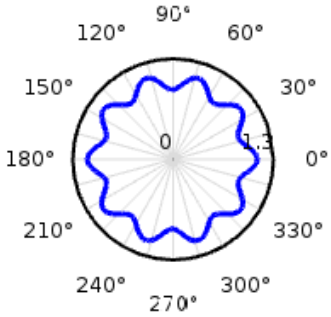
Polar Plot Group 2:

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

`Axis` – **r minimum, r maximum**=`0,1.3` (*два поля*)

LMB > Plot (*наступний рисунок праворуч*)





Майте на увазі, що клацнувши по кнопці Zoom Extents, ви зруйнуєте налаштування в розділі **Axis**.

Подивимося, як буде виглядати Point Graph графік в полярній групі. Для нього в якості джерела даних потрібно використати точкове джерело. Створіть його.

Results -> Data Set - More Data Sets -> Cut Point 1D

Data Sets - Cut Point 1D 1:

Data Set – Data = Grid 1D 1

Point Data – Entry method = Coordanates

Point Data – x = range(0,pi/4,2*pi)

Тепер побудуйте Point Graph графік в існуючій групі **Polar Plot Group 2** і налаштуйте його аналогічно до попереднього Line Graph графіка, обравши в якості ДД точкове джерело **Cut Point 1D 1**.

Polar Plot Group 2: RMB > Point Graph

Polar Plot Group 2 - Point Graph 1:

Data Set – Data = Cut Point 1D 1

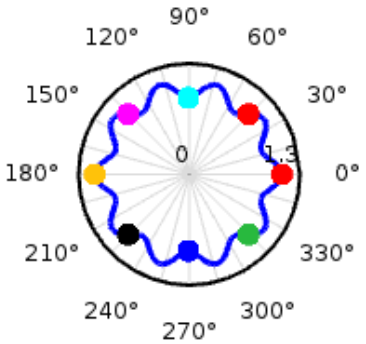
r-Axis Data – Expression = 1+cos(10*theta)/10

θ-Angle Data – Parameter = Expression

θ-Angle Data – Expression = theta

В розділі **Coloring and Style** задайте стиль, колір і розмір точок.

LMB > Plot



Тут вираз 1+cos(10*theta)/10 обчислюється в точках ДД **Cut Point 1D 1**, чиї координати розглядаються як полярні кути.

Якщо в цю графічну групу додати графічний вузол Global, то можна побудувати ще кілька точок по їх полярним координатам. Нагадаємо, що Global вузли призначені для побудов графіків виразів, які залежать від часу або якогось іншого непросторового параметра моделі. Коли в якості виразів використовуються сталі значення, то вони інтерпретуються як координати точок.

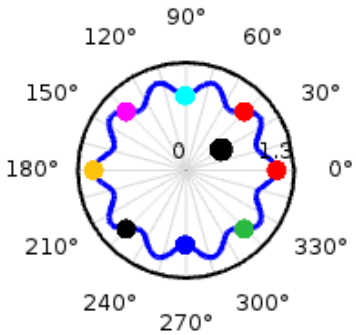
Polar Plot Group 2 -> Add Plot – Global

Polar Plot Group 2 - Global 1:

r-Axis Data – Expression = 0.5 (*введення в таблиці*)

θ-Angle Data – Parameter, Expression=Expression,pi/6 (*два поля*)

LMB > Plot



На цьому графіку внутрішня точка побудована по полярним координатам $r = 0.5$, $\theta = \pi / 6$.

■

Схожий зміст мають двовимірні графіки `Far Field` (графіки поля в дальній зоні). Зазвичай вони призначені для опису електромагнітних або акустичних полів вдалині від розрахункової зони, де немає СкЕ сітки. В відповідних інтерфейсах (електромагнітні, акустичні) реалізовані методики побудови глобальних виразів, які представляють якусь характеристику поля (амплітуду, енергію, тиск, тощо) в дальній зоні у вигляді виразу $C(r) \cdot U(\theta)$. Обираючи $C(r) = 1$ будується полярний графік виразу $U(\theta)$, який зображає відносне значення поля в напрямку кута θ. Часто такі графіки називають діаграмами спрямованості. Вони будуються шляхом деформування одиничного кола (або сфери в 3D випадку) в радіальному напрямку, тобто побудовою нормалізованих полярних графіків $r = U(\theta)$.

Відмінність двовимірних графіків в дальній зоні від полярних графіків COMSOL полягає в використанні просторових координат. Так в попередньому прикладі в полярних графіках ми використовували лише вирази $r(\theta)$, а в виразах графіків `Far Field` треба використовувати імена просторових координат `x`,`y`, які на одиничному колі пов'язані з полярним кутом співвідношеннями $x = \cos\theta$, $y = \sin\theta$. Таким чином, для створення графіка в дальній зоні потрібно мати глобально визначений вираз просторових координат $g(x, y)$, який при побудові буде трансформований в полярний графік $r = g(\cos\theta, \sin\theta) / \|g\| = \tilde{g}(\theta)$, де $\|g\|$ - необов'язковий нормуючий множник.

Приклад 3. Графік виразу $\frac{1}{16}\left[(x-y)^8 + (x+y)^8\right]$ в дальній зоні.

Створіть шаблон порожньої моделі `Blank Model`.

File -> New -> Blank Model

Додайте в модель двовимірний компонент.

`Home` -> Model -> Add Component – 2D

Геометрія компонента складатиметься з одного квадрата.

`Geometry` -> **`Primitives`** – Square

Geometry 1 – Square 1:

`Size` – **Side length** = `2`

`Position` – **Base** = `Center`

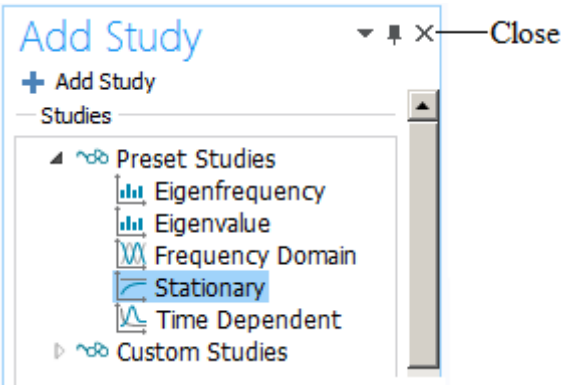
Position – **x, y** = 0, 0

На квадраті генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

В виразах Far Field графічних вузлів застосовуються лише обчислені змінні. Це призводить к необхідності мати в моделі вузол дослідження Study.

Клацніть правою кнопкою миші по самому верхньому вузлу дерева моделі (**Untitled.mph**) і в випадному меню виберіть команду Add Study, а потім в панелі Add Study, яка з'явиться збоку, виберіть Stationary тип дослідження, і закрийте панель. В принципі можна вибрати будь-який інший тип дослідження.



Вибір у вікні Add Study виконується в стандартний спосіб (в нашому випадку подвійним клацанням миші по рядку Stationary або однократним клацанням по кнопці +Add Study вгорі вікна). Потім вікно Add Study закривається хрестиком (X) в правому верхньому куті.

Таку послідовність дій ми кратко записуватимемо наступним чином.

Untitled.mph (root): RMB > Add Study = Add Study ->
Preset Studies - Stationary ->X

В результаті в ДМ з'являється вузол **Study 1** і кілька його підвузлів. Тепер можна виконати обчислення.

Study 1 : LMB > Compute

Ніяких реальних обчислень в нашій моделі виконуватися не буде, але в гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**. Воно зберігатиме інформацію про СкЕ сітку і «скомпільовані» змінні.

Створіть полярну графічну групу і в ній вузол графіка в дальній зоні.

Results -> Plot Group - Polar Plot Group

Polar Plot Group 1:

Data Set – **Data** = Study 1/Solution 1

Polar Plot Group 1 -> Add Plot - Far Field

Polar Plot Group 1 - Far Field 1:

Expression – **Expression** = ((x-y)^8+(x+y)^8)/16

Evaluation – **Angles - φ resolution** = 150 (*кількість точок на кривій*)

Coloring and Style – **Color, Width** = Blue, 3 (*два поля*)

LMB > Plot

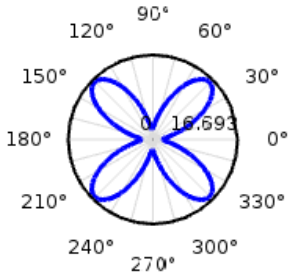
Клацніть мишею по кнопці Zoom Extents вгорі графічного вікна і потім налаштуйте діапазон зміни полярного радіуса.

Polar Plot Group 1:

`Axis` – Manual axis limits = `True`

`Axis` – r minimum = `0`

LMB > Plot



Оскільки на одиничному колі

$$\frac{1}{16}\left[(x-y)^8+(x+y)^8\right]=\frac{1}{16}\left[(\cos\theta-\sin\theta)^8+(\cos\theta+\sin\theta)^8\right]=$$

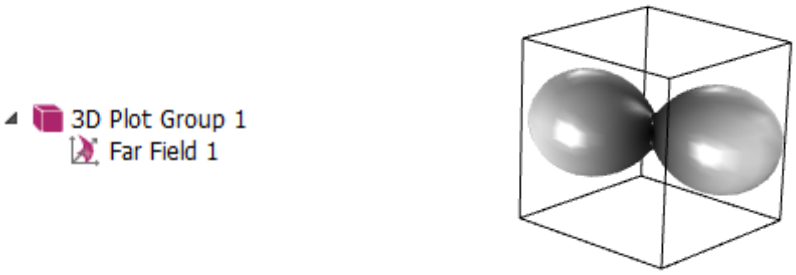
$$=\frac{1}{64}(35-28\cos 4\theta+\cos 8\theta),$$

то для порівняння ви можете самостійно побудувати полярний графік цього виразу в цій графічній групі, і отримати ту ж саму криву. Але для цього вам знадобиться створити одновимірне ДД `Grid 1D` з подовжнім параметром $0\le\theta\le 2\pi$.

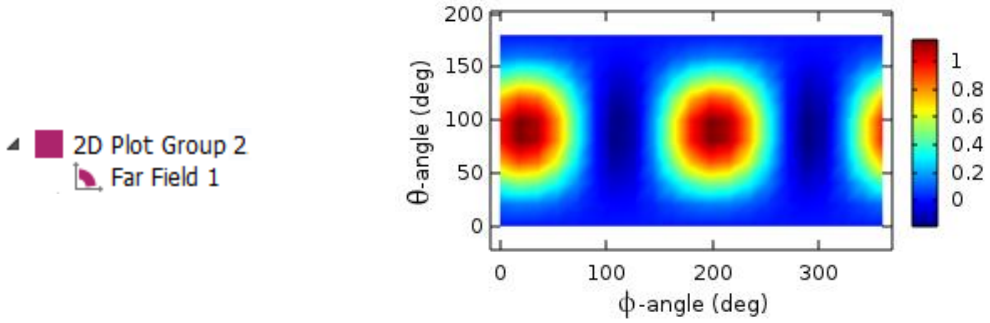
▪

Зауваження. Графіки `Far Field` можна будувати в `3D` і `2D axisymmetric` моделях.

Вправа. Побудуйте тривимірний `Far Field` графік (тобто той, що спирається на тривимірне ДД `Study 1/Solution 1`) виразу `x^2+x*y` в тривимірній графічній групі.



Побудуйте тривимірний `Far Field` графік того ж самого виразу `x^2+x*y` в двовимірній графічній групі.



2.1.2. Графіки параметрично заданих кривих і поверхонь.

В цьому параграфі продемонстровано використання джерел даних `Parameterized Curve 2D`, `Parameterized Curve 3D`, `Parameterized Surface` і `Revolution 1D`.

Приклад 1. Створіть шаблон порожньої моделі `Blank Model`.

File -> New -> Blank Model

Джерелам даних `Parameterized Curve` в якості базових потрібно мати інші джерела, якими в цьому прикладі будуть решітки `Grid`. Для їх функціювання потрібно мати в моделі хоча б одну функцію. Створіть функцію-заглушку.

Global Definitions : RMB > Functions – Analytic

Global Definitions - Analytic (an1):

Function name = `f`

`Definition` –**Expression, Arguments** = `1,x` (*два поля*)

Для побудови параметрично заданої кривої на площині знадобиться двовимірне джерело даних, яке буде репрезентувати цю площину.

Results - Data Sets: RMB > More Data Sets -> Grid 2D

Results - Data Sets - Grid 2D 1 :

`Data` – **Source, Function** = `Function, All` (*два поля*)

`Paramemter Bounds` –

– **First parameter** – **Minimum, Maximum** = `-1,1` (*два поля*)

– **Second parameter** – **Minimum, Maximum**= `-1,1` (*два поля*)

LMB > Plot

Тепер можна створювати джерело даних `Parameterized Curve 2D`.

`Results` -> Data Set - More Data Sets -> Parameterized Curve 2D

Data Sets - Parameterized Curve 2D 1 :

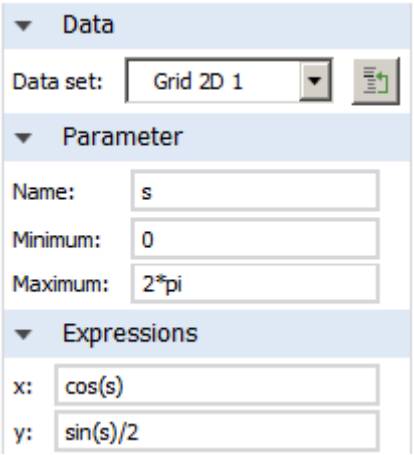
`Data` – **Data set** = `Grid 2D 1`

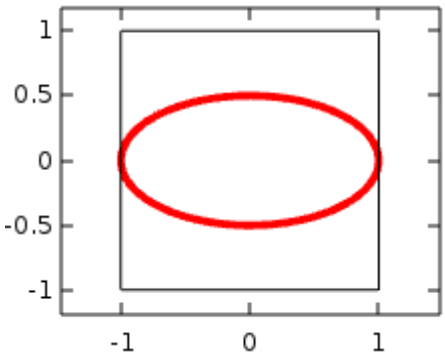
`Parameter` – **Name** = `s`

`Parameter` – **Minimum, Maximum** = `0,2*pi` (*два поля*)

`Expression` – **x, y** = `cos(s), sin(s)/2` (*два поля*)

Виконайте команду Plot і отримайте зображення джерела даних у вигляді еліпса.





Це джерело даних (плоска параметрична крива) можна використовувати для різних цілей, зокрема, для побудови більш представницького графіка еліпса.

Results –> Plot Group - 2D Plot Group

2D Plot Group 1 :

Data – **Data set** = Parameterized Curve 2D 1

2D Plot Group 1–> Add Plot – Line

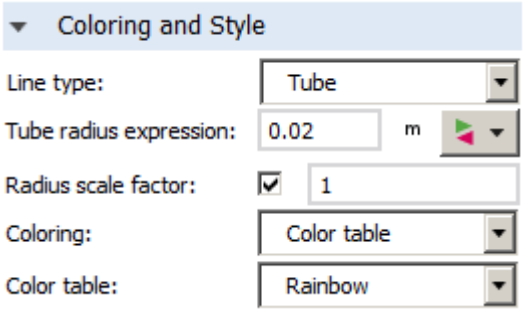
Вузли типу Line використовуються для графічного зображення скалярних функцій уздовж різноманітних ліній: параметричних кривих, границь 2D областей або ребер 3D областей.

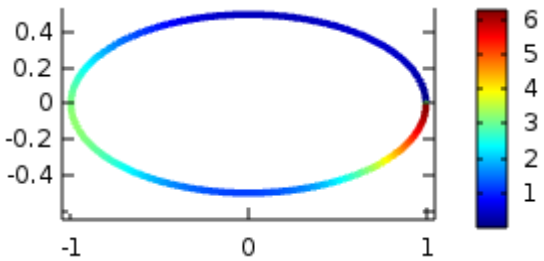
2D Plot Group 1 - Line 1 :

Data – **Data set** = From parent

Expression – **Expression** = (x^2+y^2)*s

Вираз в полі **Expression** може містити імена просторових координат x і y, а також ім'я параметра s, що використано у вузлі **Parameterized Curve 2D 1**. Вираз буде визначати колір точок еліпса. В розділі **Coloring and Style** задайте налаштування так, як показано нижче. Виконайте команду Plot і отримайте зображення еліпса з функціональним забарвленням.





Параметризована крива, тобто ДД Parameterized Curve 2D, яке її репрезентує, може виступати в якості одновимірного джерела даних. Перевіримо це.

В гілку **Results** додайте вузол 1D Plot Group, в якому створіть підвузол Line Graph.

Results –> Plot Group - 1D Plot Group

1D Plot Group 2 : Data – **Data set** = Parameterized Curve 2D 1

1D Plot Group 2–> Add Plot – Line Graph

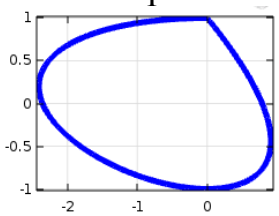
1D Plot Group 2 - Line Graph 1 :

Data – **Data set** = From parent

y-Axis Data – **Expression** =x

x-Axis Data – **Parameter, Expression** = Expression, y*s (*два поля*)

Виконайте команду Plot і отримайте зображення наступної кривої.



Кожна точка джерела даних **Parameterized Curve 2D 1** має власні координати x і y, а також значення параметра s. Використання їх в виразах розділів **y-Axis Data** і **x-Axis Data** дозволяє побудувати досить складні криві. Але якщо

згадати, що в точках ДД **Parameterized Curve 2D 1** $x = \cos s$, $y = 0.5 \sin s$, то зрозуміло, що побудована крива має параметричні рівняння $x = 0.5 s \cdot \sin s$, $y = \cos s$.

Щоб протестувати роботу з тривимірними параметричними кривими, спочатку створіть тривимірне ДД `Grid 3D`.

`Results` –> Data Set - More Data Sets –> Grid 3D

Results - Data Sets - Grid 3D 2 :

`Data` – **Source, Function** = `Function, All` (*два поля*)

`Paramemter Bounds` –

– **First parameter** – **Minimum, Maximum** = `-1,1` (*два поля*)

– **Second parameter** – **Minimum, Maximum** = `-1,1` (*два поля*)

– **Third parameter** – **Minimum, Maximum** = `-1,1` (*два поля*)

LMB > Plot

Тепер створіть джерело даних `Parameterized Curve 3D`.

`Results` –> Data Set - More Data Sets –> Parameterized Curve 3D

Data Sets - Parameterized Curve 3D 2 :

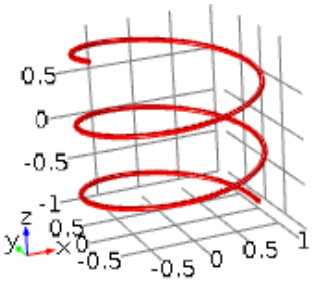
`Data` – **Data set** = `Grid 3D 2`

`Parameter` – **Name** = `s`

`Parameter` – **Minimum, Maximum** = `0,6*pi` (*два поля*)

`Expression` – **x, y,z** = `cos(s), sin(s),s/8-1` (*три поля*)

Виконайте команду Plot і отримайте зображення джерела даних у вигляді гвинтової лінії.



Це джерело даних можна використовувати для різних цілей, зокрема, для побудови на «поверхні» спіралі функціонального забарвлення.

Створіть тривимірну графічну гілку і в ній `Line` вузол

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

3D Plot Group 3 : `Data` – **Data set** = `None`

`3D Plot Group 3` –> Add Plot - Line

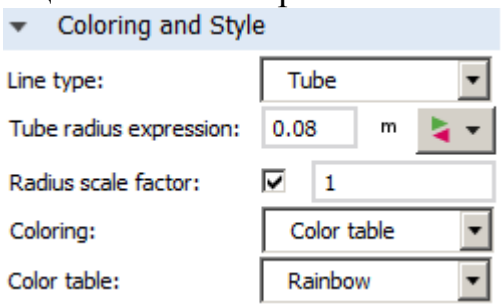
3D Plot Group 3 - Line 1 :

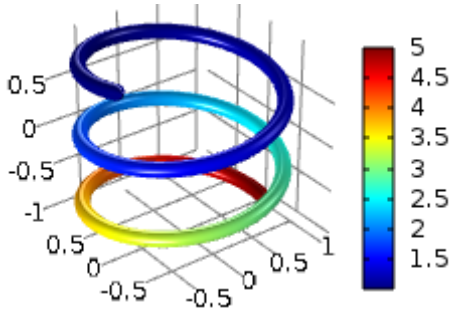
`Data` – **Data set** = `Parameterized Curve 3D 2`

`Expression` – **Expression** = `x^2+y^2+(z-1)^2`

Тут вираз в полі **Expression** може містити імена просторових координат `x`, `y` і `z`, а також ім'я параметра `s`, що задано в ПВ джерела `Parameterized Curve 3D 2`. Вираз буде визначати функціональне забарвлення спіралі.

В розділі **Coloring and Style** задайте налаштування такі, як показано нижче. Виконайте команду Plot і отримайте зображення спіралі з функціональної забарвленням.





Якщо в розділі **Expression** в полі **Expression** ввести константу, наприклад, 1, то ви отримаєте одноколірну криву.

Розглянемо роботу ДД `Parameterized Surface`.

`Results` -> Data Set - More Data Sets -> Parameterized Surface

Data Sets - Parameterized Surface 1 :

Data – **Data set** = `Grid 3D 2`

Parameter – **First parameter** – **Name** = `s1`

– **First parameter** – **Minimum, Maximum** = `-pi/2,pi/2`

– **Second parameter** – **Name** = `s2`

– **Second parameter** – **Minimum, Maximum** = `0,2*pi`

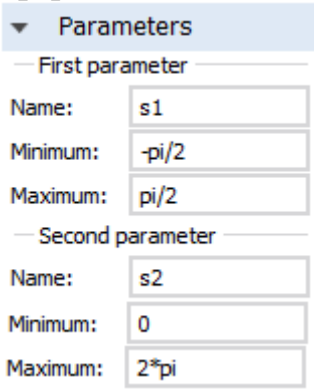
Expression – **x** = `cos(s1)*cos(s2)`

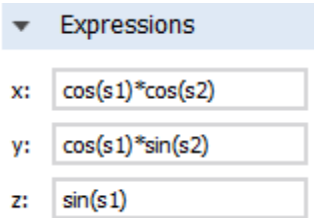
– **y** = `cos(s1)*sin(s2)`

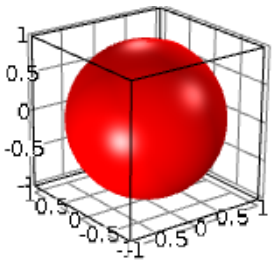
– **z** = `sin(s1)`

Тут параметри `s1` і `s2` виступають в якості криволінійних координат на поверхні.

Виконайте команду Plot і отримайте зображення області джерела даних в формі сфери.







Це джерело даних можна використовувати для різних цілей, зокрема, для побудови на його поверхні функціонального забарвлення.

Поверніться до гілки **3D Plot Group 3** (тобто активуйте цей вузол), і в ній створіть `Surface` вузол.

`3D Plot Group 3` -> Add Plot – Surface

Вузли такого типу використовуються для графічного представлення скалярних функцій на параметричних поверхнях, всередині 2D або на межі 3D областей.

3D Plot Group 3 - Surface 1 :

Data – **Data set** = Parameterized Surface 1

Expression – **Expression** = x^2+y^2+(z-1)^2

Вираз в полі **Expression** може містити імена просторових координат x, y і z, а також імена параметрів s1 і s2, які визначені в джерелі **Parameterized Surface 1**.

Виконайте команду Plot і отримайте зображення сфери з функціональним забарвленням. На рисунку повинні бути присутніми спіраль і сфера. Скалярна функція x^2+y^2+(z-1)^2, яка використовується нами для функціонального забарвлення кривої і поверхні однакова, проте кольори не узгоджені. Це видно по різним діапазонам значень на шкалах кольорів. Виправимо цей недолік.

3D Plot Group 3 - Line 1 :

Range – **Manual color range** = True

Range – **Minimum, Maximum** = 0,5 (*два поля*)

3D Plot Group 3 - Surface 1 :

Range – **Manual color range** = True

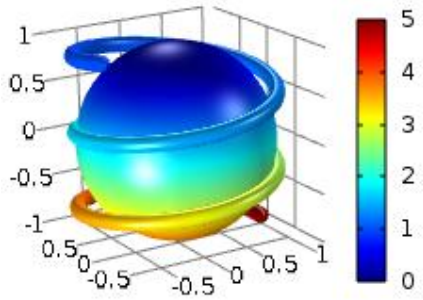
Range – **Minimum, Maximum** = 0,5 (*два поля*)

Щоб узгодити кольори в ПВ кривої і поверхні в розділах **Range** ми задали однакові мінімальне і максимальне значення. Minimum=0 це мінімальне значення з двох мінімумів, а Maximum =5 – це максимальне з двох максимумів.

Після виконання команди Plot функціональне забарвлення обох графіків буде узгоджене, і одну з палітр можна видалити, наприклад, з вузла **Surface 1** .

Coloring and Style – **Color legend** = False

Нижче наведено остаточний вигляд графіка.



Продовжимо знайомитися з новими типами джерел даних, які будемо розміщувати в тому ж файлі, що і раніше.

Розглянемо джерело даних типу Revolution 1D . Воно являє собою двовимірну область, яка утворюється обертанням одновимірного джерела даних (яке не повинно перетинати вісь обертання).

Спочатку створіть джерело даних Grid 1D.

Results -> Data Set - More Data Sets -> Grid 1D

Data Sets - Grid 1D 3 :

Data – **Source, Function** = Function, All (*два поля*)

Paramemter Bounds – **Name**= x

Paramemter Bounds – **Minimum, Maximum** = 1,2 (*два поля*)

Тепер створіть джерело даних типу Revolution 1D.

`Results` -> Data Set - More Data Sets -> Revolution 1D

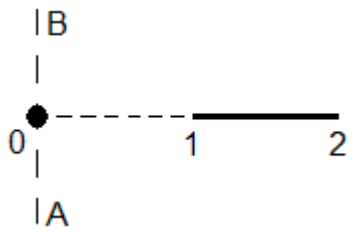
Results - Data Sets - Revolution 1D 1 :

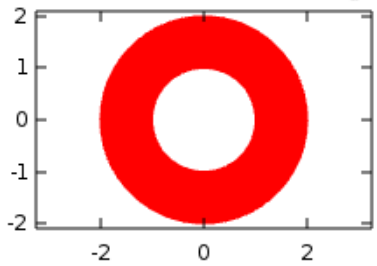
`Data` – **Data set** = `Grid 1D 3`

`Axis Data` – **`x`** = `0`

Останнє значення визначає точку на осі OX, навколо якої буде обертатися відрізок [1,2], заданий джерелом даних **Grid 1D 3**. Обертання буде виконуватися навколо осі AB (див. наступний рисунок ліворуч). В розділі **`Revolution Layers`** в полях **Start angle** і **Revolution angle** задаються параметри/кути обертання. В нашому прикладі залишимо в цих полях значення за замовчуванням `0` і `360` (градусів).

Виконайте команду Plot і отримайте вид зверху (з точки B на площину обертання) області джерела даних в формі кільця.





Це джерело даних можна використовувати для виконання функціонального забарвлення або побудови поверхні обертання.

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

2D Plot Group 4 :

`Data` – **Data set** = `Revolution 1D 1`

`2D Plot Group 4` -> Add Plot - Surface

2D Plot Group 4 - Surface 1 :

`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `(x-0.5)^2`

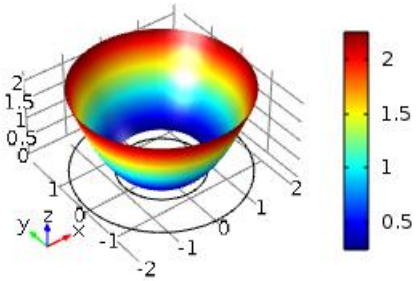
Тут вираз в полі **Expression** може залежати тільки від параметра `x`, ім'я якого задано в базовому джерелі даних **Grid 1D 3**. Вираз буде визначати функціональне забарвлення кільця.

У вузол **Surface 1** додайте підвузол `Height Expression`.

2D Plot Group 4 – Surface 1 : RMB > Height Expression

Surface 1 – Height Expression 1 : `Axis` – **Scale factor** = `True, 1` (*два поля*)

LMB > Plot



Таким чином, функція однієї змінної `x` (`x` - координата в одновимірному джерелі даних **Grid 1D 3**) використовується для побудови графіка вісесиметричної функції, тобто поверхні обертання.

Джерело даних Revolution 1D може спиратися на будь-яке інше одновимірне джерело даних.

Створіть нове джерело даних типу Revolution 1D.

Results -> Data Set - More Data Sets -> Revolution 1D

Data Sets - Revolution 1D 2 :

Data – **Data set** = Parameterized Curve 2D 1

Axis Data – **x** = 0 (*абсциса точки обертання*)

В розділі **Revolution Layers** в полях **Start angle** і **Revolution anle** залиште значення за замовчуванням 0 і 360. Виконайте команду Plot і отримайте зображення джерела даних в формі кола радіуса 2π. Це визначається налаштуваннями в розділі **Parameter** базового джерела даних **Parameterized Curve 2D 1**.

Деактивуйте вузол **2D Plot Group 4 - Surface 1**, а в графічну групу додайте вузол **Surface 2**.

2D Plot Group 4 - Surface 1 : RMB > Disable

2D Plot Group 4 -> Add Plot - Surface

2D Plot Group 4 - Surface 2 :

Data – **Data set** = Revolution 1D 2

Expression – **Expression** =2*x*y*s+2

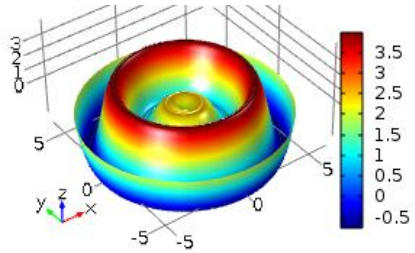
Нагадаємо, що вираз в полі **Expression** може містити імена координат x, y, а також ім'я параметра s. Ці імена використовуються в базовому джерелі **Parameterized Curve 2D 1** вузла **Revolution 1D 2**. Вираз буде визначати функціональне забарвлення джерела даних (кола) і/або його форму поверхні.

У вузол **Surface 2** додайте підвузол Height Expression.

2D Plot Group 4 – Surface 2 : RMB > Height Expression

Surface 2 – Height Expression 1 : Axis – **Scale factor** = True, 1 (*два поля*)

LMB > Plot



■

Джерелом даних може бути СкЕ сітка (Mesh), існуюча в моделі. Її можна використовувати таким же чином як і решітки Grid. Ми не будемо повторювати приклади побудови графіків різних типів, що спираються на джерело типу Mesh, а продовжимо розгляд нових типів джерел даних, які зазвичай використовують Mesh як базове джерело.

2.1.3. Використання сіткових джерел даних та їх трансформацій.

Найчастіше в моделях використовуються сіткові джерела даних, тобто області, в яких побудовано СкЕ сітку. Для побудови графіків їх можна використовувати

таким же чином, як і `Grid` джерела. Але над сітковими джерелами можна виконувати деякі перетворення: розмноження, віддзеркалення, поворот, перетин, тощо. В наступних прикладах продемонстровано способи використання таких трансформацій.

Приклад 1. Створіть шаблон порожньої моделі `Blank Model`.

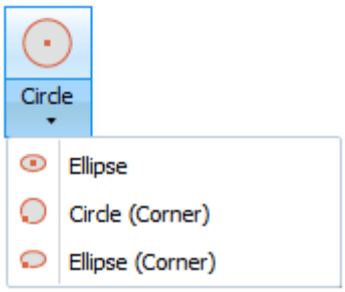
File -> New -> Blank Model

Клацніть правою кнопкою миші по верхньому вузлу дерева моделі (**Untitled.mph**) і в випадному меню виберіть команду Add Component - 2D. Це буде означати, що ми працюватимемо з двовимірною моделлю.

Untitled.mph : RMB > Add Component - 2D

З'являється вузол **Component 1** і кілька його підвузлів, з яких ми будемо використовувати вузли **Geometry 1** і **Mesh 1**. В дереві моделі виділіть вузол **Geometry 1**. Після цього графічне вікно зміниться і набуде вигляду, налаштованого для креслення плоских геометричних фігур. На стрічці перейдіть на закладку `Geometry`. На ній клацніть один раз мишею по кнопці-списку Circle і виберіть команду Ellipse.

`Geometry` -> Draw - Circle - **`Ellipse`**

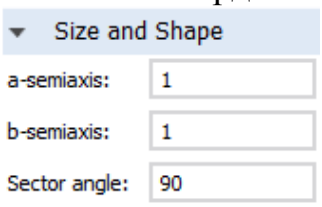


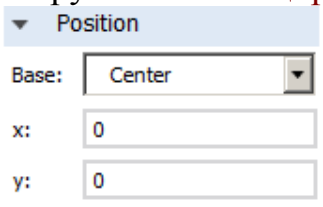
Наведіть курсор миші в область креслення, натисніть ліву кнопку, посуньте мишу і відпустіть кнопку. В результаті буде намальований еліпс. В дереві моделі в гілці **Geometry 1** з'явиться дочірній вузол **Ellipse 1**. Перейменуйте його. Для цього в ПВ в полі **Label** введіть `QuarterCircle`, перемістіть фокус в будь-яке місце з цього поля і в дереві моделі назва вузла зміниться.

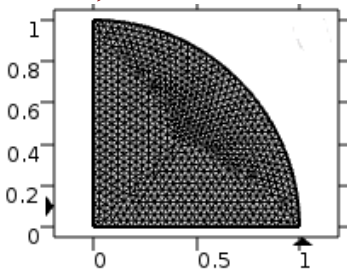
Geometry 1 – Ellipse 1 (e1) :

Label = `QuarterCircle`

Поки вузол **QuarterCircle (e1)** (колишня назва **Ellipse 1 (e1)**) активний праворуч від нього буде відкрита панель властивостей. В ній в розділі **`Size and Shape`** задаються півосі еліпса, а також кут (в градусах), який визначає, яку частину еліпса слід будувати (якщо 360, то повністю). В розділі **`Position`** можна змінити координати центру. Заповніть ці розділи так, як показано нижче.







Вгорі ПВ натисніть кнопку Build All Objects Build All Objects. В графічної зоні з'явиться зображення чверті кола.

Побудуйте СкЕ сітку.

Mesh 1 : `Mesh Settings` – Element size = `Finer`

LMB > Build All (*попередній рисунок праворуч*)

В гілку даних **Data Sets** дерева моделі додайте джерело даних, яке буде репрезентувати цю сітку.

Results - Data Sets : RMB > Mesh

Data Sets - Mesh 1 : `Mesh` – Mesh = `Mesh 1`

До речі, СкЕ сітку можна зобразити в графічному вузлі. Для цього достатньо в графічній групі створити вузол типу `Surface`, підключити його до ДД типу `Mesh` (або до ДД, яке його містить, наприклад, `Study 1/Solution 1`), і в ПВ вузла `Surface` в розділі **`Coloring and Style`** увімкнути прапорець **Wireframe**.

Двовимірне джерело даних `Mesh` можна використовувати аналогічно джерелу `Grid 2D`, проте є відмінності, а також додаткові можливості.

Побудуємо над створеною областю (чверть кола) графік функції двох змінних $z = \left(|x+y| - |x-y|\right)\cdot\left(1 - x^2 - y^2\right)$.

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

2D Plot Group 1 : `Data` – Data set = `Mesh 1`

`2D Plot Group 1` -> Add Plot - Surface

2D Plot Group 1 - Surface 1 :

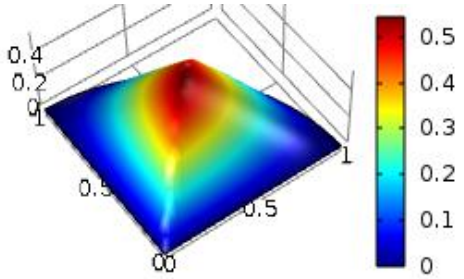
`Data` – Data set = `From parent`

`Expression` – Expression = `(abs(x+y)-abs(x-y))*(1-x^2-y^2)`

2D Plot Group 1 – Surface 1 : RMB > Height Expression

Surface 1 – Height Expression 1 : `Axis` – Scale factor = `True, 1` (*два поля*)

LMB > Plot



Поверхня побудована над розрахунковою областю, яка була створена в гілці **Geometry 1**, а форма поверхні задана виразом в полі **Expression**.

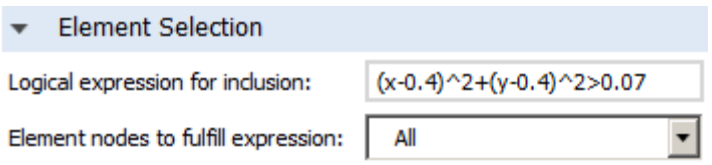
В графічних вузлах типу `Surface` іноді використовують підвузли типу `Filter`. Вони призначені для «фільтрування» зображення відповідно з логічним виразом, який задається в окремому полі. Додайте до вузла **Surface 1** підвузол `Filter` і налаштуйте його так, як показано нижче.

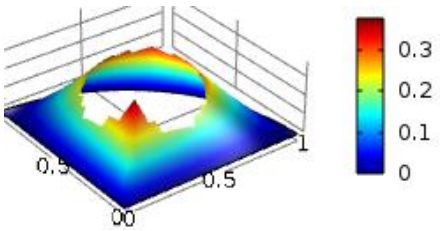
2D Plot Group 1 – Surface 1 : RMB > Filter

2D Plot Group 1 – Surface 1 - Filter 1 :

`Element Selection` – Logical expression for inclusion = `(x-0.4)^2+(y-0.4)^2>0.07`

LMB > Plot (*рисунок праворуч*)

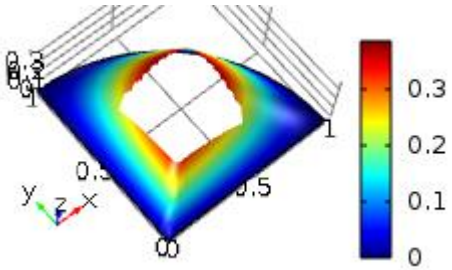




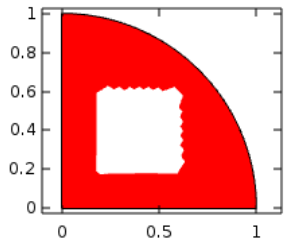
Логічна умова `(x-0.4)^2+(y-0.4)^2>0.07`, яка задана в полі **Logical expresion for inclusion**, представляє критерій, який виділяє на графіку ту його частину, яка повинна бути показана. Логічний вираз може містити оператори порівняння і логічні оператори. Спробуйте, наприклад, замінити введену умову на логічний вираз `!(abs(x-0.4)<0.2 && abs(y-0.4)<0.2)`, який приймає значення ІСТИНА зовні невеликого квадрата.

`Element Selection` – **Logical expression for inclusion** = `!(abs(x-0.4)<0.2 && abs(y-0.4)<0.2)`

LMB > Plot



Результат «фільтрації» плоскої області чверті кола показано на наступному рисунку.



Щоб отримати це зображення слід виконати наступні інструкції.

2D Plot Group 1 – Surface 1 – Height Expression 1 : RMB > Disable

2D Plot Group 1 – Surface 1 :

`Coloring and Style` – **Coloring** = `Uniform`

LMB > Plot (*попередній рисунок*)

Зауваження 1. Якщо джерелом даних графічного вузла є решітка `Grid`, то фільтр не працює.

Зауваження 2. Рвані края є наслідком того, що межа виділеної зони проходить по контурам скінченних елементів.

Поверніть налаштування вузла **2D Plot Group 1**.

2D Plot Group 1 – Surface 1 – Height Expression 1 : RMB > Enable

2D Plot Group 1 – Surface 1 :

`Coloring and Style` – **Coloring** = `Color table`

LMB > Plot

Вираз, який було використано в полі **Expression** вузла **Surface 1**, досить складний. Запишемо його у вигляді функції двох змінних.

Global Definitions : RMB > Functions – Analytic

Function name = g

Definition – **Expression** = (abs(x+y)-abs(x-y))*(1-x^2-y^2)

Definition – **Arguments** = x,y

Для самоперевірки побудуйте графік створеної функції, виконавши команду Plot.

Використання глобальних функцій в графічних вузлах, які спираються на розрахункові області, що містять СкЕ сітку, трохи відрізняється від їх використання разом з джерелами даних типу Grid 2D. Джерелам Grid з самого початку в полях **Data–Source**=Function вказувалося, що вони працюють разом з функціями. Але глобальні функції стають доступними в гілці обробки даних **Results** у вузлах, що спираються на сіткові вузли, тільки після виконання яких-небудь обчислень. Для цього в моделі потрібно мати вузол дослідження Study.

Додайте в модель Stationary тип дослідження.

Untitled.mph (root): RMB > Add Study = Add Study ->

Preset Studies - Stationary **->x**

В результаті в ДМ з'являється гілка **Study 1** і кілька її підвузлів. Натисніть клавішу F8, виконавши команду Compute. Ніяких реальних обчислень в моделі не виконуватиметься, але в гілці **Results - Data Sets** буде створено вузол (джерело даних) **Study 1/Solution 1**. Він буде зберігати інформацію про СкЕ сітку, а також про «скомпільовану» глобальну функцію.

Клацніть по вузлу **2D Plot Group 1** правою кнопкою миші і виберіть команду Duplicate. Цим в дереві моделі буде створена копія вузла **2D Plot Group 1** з новим ім'ям **2D Plot Group 2**. Змініть в ньому деякі налаштування.

2D Plot Group 1 : RMB > Duplicate

2D Plot Group 2 : **Data** – **Data set** = Study 1/Solution 1

2D Plot Group 1 – Surface 1 :

Data – **Data set** = From parent

Expression – **Expression** =g(x,y)

Виконайте команду Plot і отримайте той же графік, що і раніше, але тепер побудований з використанням глобальної функції g(x,y).

Таким чином, щоб використовувати глобальні функції в математичних виразах графічних вузлів, які спираються на сіткові джерела даних, воно (джерело даних, наприклад, Mesh) повинно бути «розміщено всередині» джерела даних Study 1/Solution 1. Після цього в полях **Data set** графічних вузлів, які будуть використовувати функції, треба обирати таке «обчислене» ДД.

Будувати графіки виразів, що залежать від двох змінних, можна не у всій області, а наприклад, по якій-небудь лінії її перетину. Для цього в гілці **Data Sets** треба створити інше джерело даних, яке буде репрезентувати лінію перетину. Нагадаємо, як таке ДД використовується.

Data Sets : RMB > Cut Line 2D

Data Sets - Cut Line 2D 1 :

Data – **Data set** = `Mesh 1`

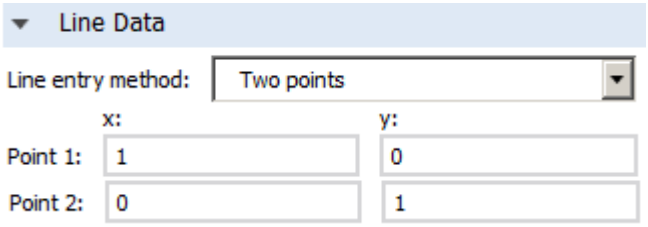
Line Data – **Line entry method** = `Two points` (*перетин по двох точках*)

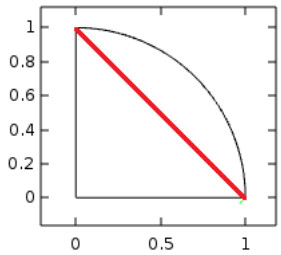
– **Point 1 : x, y**= `1,0` (*два поля*)

– **Point 2 : x, y**= `0,1` (*два поля*)

Цим було задано координати точок, через які проходить перетин.

Виконайте команду Plot, і отримайте графік лінії перерізу, тобто джерела даних **Cut Line 2D 1**.





Побудуйте на цьому перерізі графік функції. Для цього створіть групу одновимірних графіків і налаштуйте її.

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

1D Plot Group 3: Data Set – Data = `Cut Line 2D 1`

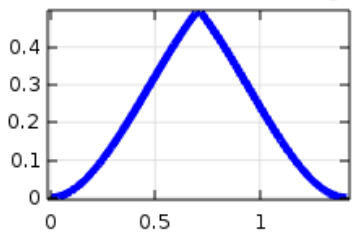
RMB > Line Graph

1D Plot Group 3 - Line Graph 1:

y-Axis Data – **Expression** = `(abs(x+y)-abs(x-y))*(1-x^2-y^2)`

В розділі **x-Axis Data** в списку **Parameter** залиште значення `Arc length`. В розділі **Coloring and Style** задайте власні стиль, колір і товщину лінії.

LMB > Plot



Зауваження. Якщо в полі **y-Axis Data**–**Expression** використати вираз `g(x,y)`

1D Plot Group 3 - Line Graph 1: y-Axis Data – Expression = `g(x,y)`

то буде згенеровано повідомлення про помилку. Це пов'язано з тим, що ДД **Cut Line 2D 1** спирається на ДД **Mesh 1**, а не **Study 1/Solution 1**. Якщо це виправити і ввести

Data Sets - Cut Line 2D 1 : Data – **Data set** = `Study 1/Solution 1`

то буде побудовано той же (попередній) графік.

Розглянемо трансформації, які можна виконувати з сітковими джерелами даних.

В гілку даних **Data Sets** дерева моделі добавимо джерело даних `Mirror 2D`. Воно призначено для поширення розв'язку (або виразу) з деякої зони на область, яка отримується об'єднанням розрахункової області з її віддзеркаленою відносно вісі симетрії копією. Віддзеркалюються також всі обчислені у вихідній області змінні і функції.

Results -> Data Set - More Data Sets -> Mirror 2D

Data Sets - Mirror 2D 1 :

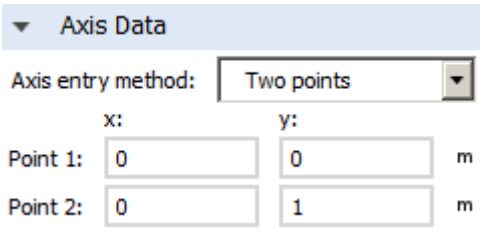
Data – **Data set** = Mesh 1

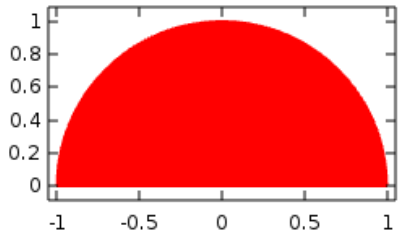
Axis Data – Point 1 – **x, y** = 0,0

Axis Data – Point 2 – **x, y** = 0,1

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 вихідної області та її дзеркального образу.





Тепер це джерело даних можна використовувати для виконання функціонального забарвлення або побудови над ним графіка функції двох змінних.

Results -> Plot Group - 2D Plot Group

2D Plot Group 4 :

Data – **Data set** = Mirror 2D 1

2D Plot Group 4 -> Add Plot - Surface

2D Plot Group 4 - Surface 1 :

Data – **Data set** = From parent

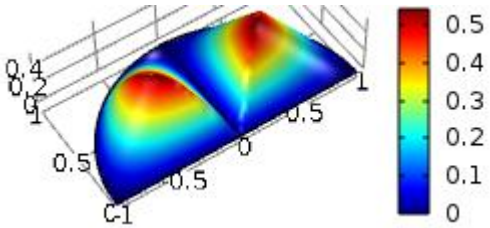
Expression – **Expression** = (abs(x+y)-abs(x-y))*(1-x^2-y^2)

Останній вираз в зоні **Mesh 1** (частина області **Mirror 2D 1**) буде застосовуватися явно, а в дзеркально відображеній частини буде використано його дзеркальне відображення.

2D Plot Group 4 – Surface 1 : RMB > Height Expression

Surface 1 – Height Expression 1 : Axis – **Scale factor** = True, 1 (*два поля*)

LMB > Plot



Розглянемо використання джерела даних Array 2D. Воно призначено для поширення розв’язку (або виразу) з деякої зони на область, яка отримується шляхом багаторазового розмноження за допомогою зсуву вихідної розрахункової області.

Results -> Data Set - More Data Sets -> Array 2D

Data Sets - Array 2D 1 :

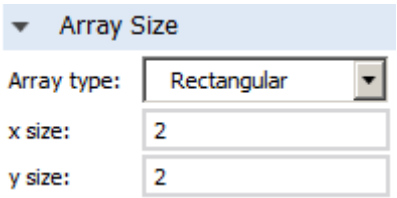
Data – **Data set** = Study 1/Solution 1

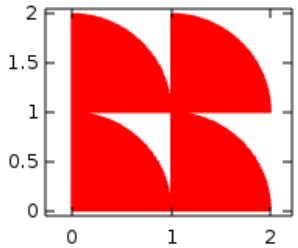
Нагадаємо, що ДД **Study 1/Solution 1** містить «всередині» джерело **Mesh 1** (нижче ми будемо використовувати функцію `g(x,y)`, а не вираз).

`Array Size` – **Array type** = `Rectangular` (*прямокутний масив*)

`Array Size` – **x size, y size** = `2,2` (*два поля*)

Останнім задається кількість рядків і стовпців в прямокутному масиві джерел. Він (масив джерел) утворюється зсувом охоплюючого базове джерело прямокутника (у нас це квадрат $[0,1]\times[0,1]$) на задану кількість разів вправо і вгору для утворення масиву областей, який розглядається як єдине джерело даних. Виконайте команду Plot і отримайте зображення області нового джерела даних.





Створене джерело даних можна використовувати для виконання функціонального забарвлення або побудови над ним графіків функцій.

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

2D Plot Group 5 : `Data` – **Data set** = `Array 2D 1`

`2D Plot Group 5` –> Add Plot - Surface

2D Plot Group 5 - Surface 1 :

`Data` – **Data set** = `From parent`

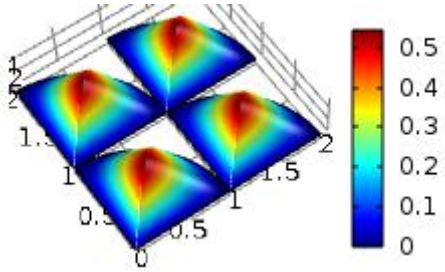
`Expression` – **Expression** =`g(x,y)`

Цей вираз в зоні **Mesh 1** (базова частина області **Array 2D 1**) буде використовуватися явно, а в зсунутих частинах буде використана його зрушена копія.

2D Plot Group 5 – Surface 1 : RMB > Height Expression

Surface 1 – Height Expression 1 : `Axis` – **Scale factor** = `True, 1` (*два поля*)

LMB > Plot



Розглянемо використання джерела даних `Sector 2D`. Воно призначено для поширення розв'язку (або виразу) з деякої зони на область, яка отримується шляхом багаторазового повороту вихідної розрахункової області.

`Results` –> Data Set - More Data Sets –> Sector 2D

Data Sets - Sector 2D 1 :

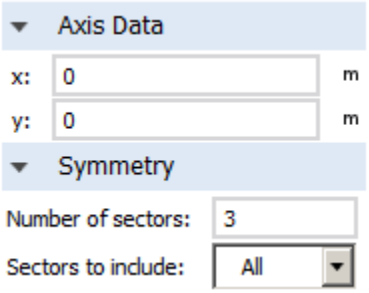
`Data` – **Data set** = `Study 1/Solution 1`

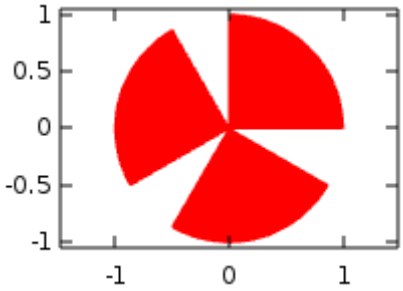
`Axis Data` – **x, y** = `0,0` (*точка повороту, два поля*)

Останнім задається точка (0,0) навколо якої буде виконуватися поворот при створені «кругового» масиву джерел даних.

Symmetry – **Number of sectors** = 3 (*кількість елементів в масиві ДД*)

Виконайте команду Plot і отримайте зображення нового джерела даних.





Побудуйте графік функції g(x,y) над новою областю.

Results -> Plot Group - 2D Plot Group

2D Plot Group 6 : Data – **Data set** = Sector 2D 1

2D Plot Group 6 -> Add Plot - Surface

2D Plot Group 6 - Surface 1 :

Data – **Data set** = From parent

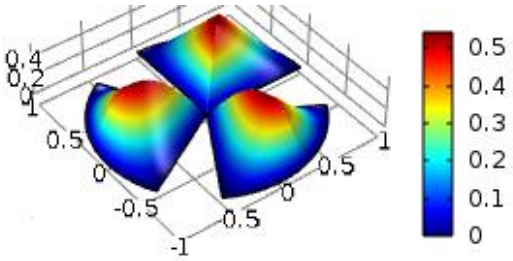
Expression – **Expression** =g(x,y)

Цей вираз в зоні **Mesh 1** (базова частина області **Sector 2D 1**) буде застосовуватися явно, а в знову створених частинах буде використана його повернена копія.

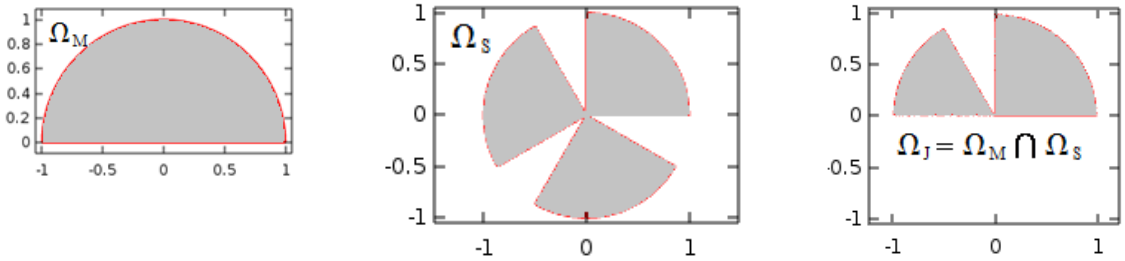
2D Plot Group 6 – Surface 1 : RMB > Height Expression

Surface 1 – Height Expression 1 : Axis – **Scale factor** = True, 1 (*два поля*)

LMB > Plot



Проаналізуємо роботу джерела даних Join. Воно призначено для комбінування розв'язків (або виразів), заданих в двох різних джерелах даних. Комбінування розв'язків/виразів виконується в спільній для обох джерел області. Наступна схема пояснює утворення джерела даних **Join 1** з джерел **Mirror 2D 1** (область Ω_M) і **Sector 2D 1** (область Ω_S). Результівне джерело даних репрезентує область $\Omega_J = \Omega_M \cap \Omega_S$. Тип операції комбінування виразів в області Ω_J задається в панелі властивостей джерела Join в полі **Combination** – **Method**.



`Results` -> Data Set - More Data Sets -> Join

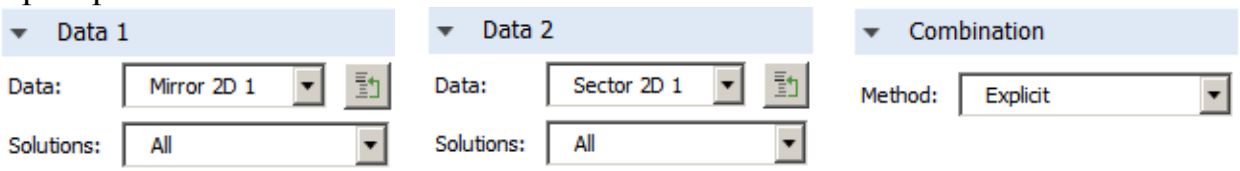
Data Sets - Join 1 :

`Data 1` – **Data** = `Mirror 2D 1` (*перше джерело даних*)

`Data 2` – **Data** = `Sector 2D 1` (*друге джерело даних*)

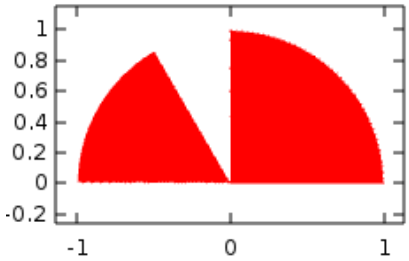
`Combination` – **Method** = `Explicit`

Остання опція означає, що для комбінування виразів будуть застосовуватися оператори `data1` і `data2`.



Нехай, наприклад, в першому джерелі обчислюється вираз `x^2+y^2`, а в другому - вираз `x*y`. Тоді в графічному вузлі, що буде спиратися на «об'єднане» джерело даних **Join 1**, можна написати, наприклад, вираз `data1(x^2+y^2)+ data2(x*y)`. Але зазвичай виразами, які використовуються в джерелах даних, є розв'язки задач. Якщо розв'язок в першому джерелі представлено змінною `u1`, а в другому - `u2`, то припустимо використати, наприклад, вираз `data1(u1)-data2(u2)`, який на спільній частині джерел буде репрезентувати різницю двох розв'язків.

Виконайте команду Plot і отримайте зображення нового джерела даних.



Побудуйте в результівній розрахунковій області **Join 1** який-небудь графік, а для цього створіть двовимірну графічну групу з `Surface` вузлом.

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

2D Plot Group 7 : `Data` – **Data set** = `Join 1`

`2D Plot Group 7` -> Add Plot - Surface

2D Plot Group 7 - Surface 1 :

`Data` – **Data set** = `From parent`

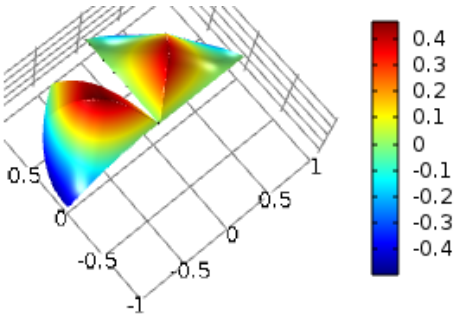
`Expression` – **Expression** =

`data1((abs(x+y)-abs(x-y))*(1-x^2-y^2))-data2(x*y)`

2D Plot Group 7 – Surface 1 : RMB > Height Expression

Surface 1 – Height Expression 1 : `Axis` – **Scale factor** = `True, 1` (*два поля*)

LMB > Plot



Способи застосування тривимірної сітки майже не відрізняються від двовимірної, окрім використання «об'ємних» `Volume` графіків.

Приклад 2. Створіть шаблон порожньої моделі `Blank Model`.

File -> New -> Blank Model

Додайте в модель тривимірний компонент.

`Home` -> Model -> Add Component – 3D

Геометрія компонента складатиметься з одного куба.

`Geometry` -> **`Primitives`** – Block

Geometry 1 – Block 1:

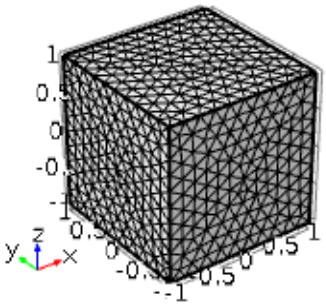
`Size` – **Width, Depth, Height** = `2,2,2` (*три поля*)

`Position` – **Base** = `Center`

`Position` – **x, y, z** = `0,0,0`

Генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All



В гілку даних **Data Sets** дерева моделі додайте джерело даних, яке буде репрезентувати цю сітку.

Results - Data Sets : RMB > Mesh

Data Sets - Mesh 1 : **`Mesh`** – **Mesh** = `Mesh 1`

СкЕ сітка створюється в об'ємі, але її можна використовувати і на поверхні. Побудуємо графік виразу $x^2 + y^2 + z^2$ на поверхні куба.

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

3D Plot Group 1 : **`Data`** – **Data set** = `Mesh 1`

3D Plot Group 1 : RMB > Surface

3D Plot Group 1 - Surface 1 :

`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `x^2+y^2+z^2`

Виконайте команду Plot, і отримайте графічне зображення виразу у вигляді функціонального забарвлення на поверхні куба, яке показано на наступному

рисунку ліворуч. Кольори залежать від числового значення введеного виразу в точках його граней.

Тепер створіть тривимірну графічну групу з підвузлом типу `Volume`. У тривимірному випадку він використовується для графічного зображення функції на поверхні тривимірного джерела даних, використовуючи значення функції у всьому його об'ємі.

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

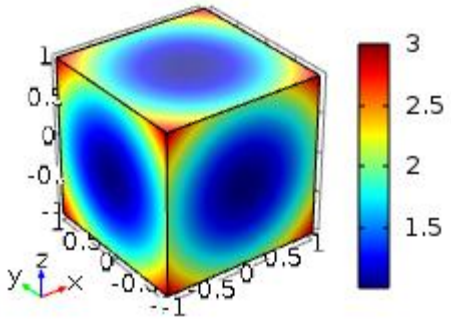
3D Plot Group 2 : `Data` – Data set = `Mesh 1`

RMB > Volume

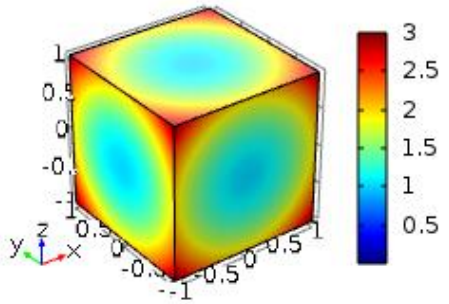
3D Plot Group 2 - Volume 1 : `Expression` – Expression = `x^2+y^2+z^2`

LMB > Plot (*наступний рисунок праворуч*)

Функціональне забарвлення відноситься до всього об'єму куба. Хоча фарбування ми бачимо на поверхні куба, однак шкала кольорів побудована за всіма значеннями виразу всередині області (а не тільки по поверхні, як у випадку вузла `Surface`).

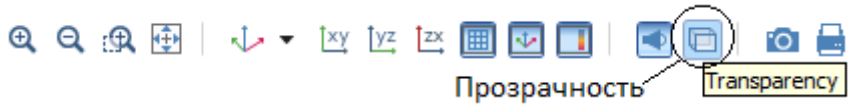


`Surface` графік



`Volume` графік

Зауваження. Над графічною зоною вузла **3D Plot Group 2** є кнопка Transparency (прозорість). Однак її увімкнення майже не впливає на наше сприйняття функціонального забарвлення «об'ємного» графіка.



Зверніть увагу, що на правому графіку темно синій колір, який відповідає нульовим значенням функції забарвлення, на поверхні куба відсутній. Це пов'язано з тим, що мінімальне (нульове) значення заданого виразу досягається в центрі куба. На око різниця між типами графіків `Surface` і `Volume` полягає лише в неоднаковому діапазоні шкали кольорів. Але це не зовсім так. У об'ємний графік ми можемо «зазирнути всередину» за допомогою підвузла `Filter`.

3D Plot Group 2 - Volume 1 : RMB > Filter

3D Plot Group 2 – Volume 1 - Filter 1 :

`Element Selection` – Logical expression for inclusion = `x+y+2*z<0.5`

LMB > Plot (*наступний рисунок ліворуч*)

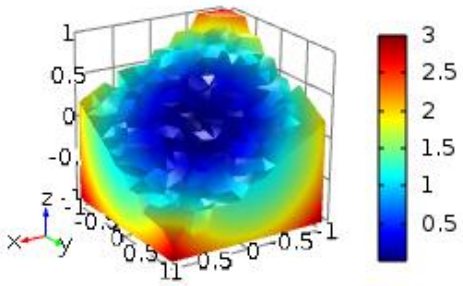
Якщо фільтр застосувати до поверхневого графіка, то буде зображено частину поверхні з порожнечею всередині (і іншою шкалою кольорів).

3D Plot Group 1 - Surface 1 : RMB > Filter

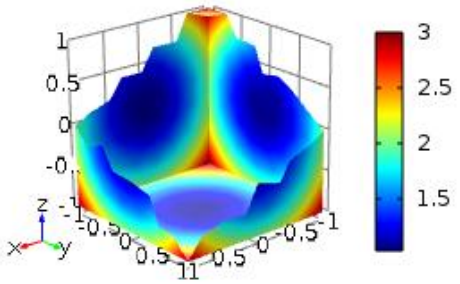
3D Plot Group 1 – Surface 1 - Filter 1 :

Element Selection–Logical expression for inclusion=x+y+2*z<0.5

LMB > Plot (*наступний рисунок праворуч*)



Volume графік з фільтром



Surface графік з фільтром

Є ще одна різниця між Surface і Volume графіками. На Volume графіки не діє приховання поверхневих елементів (а також реберних і точкових) при застосуванні підвузлів Hide Geometric Entities у View вузлах.

В системі існує тип графіків Mesh, який призначений для зображення і аналізу СкЕ сіток. Скінченний елемент вважається «добрим», якщо він близький по формі до правильного тетраедра (3D), або до правильного трикутника (2D). Вважається, що «якість правильних» елементів дорівнює 1, а якість вироджених елементів вважається 0. Всі інші елементи мають проміжну якість. Чим більше в СкЕ сітці елементів з «якістю» близькою до 1, тим краще. В Mesh графіках елементи сітки розфарбовуються відповідно до їх «якості».

При побудові Mesh графіки повинні спиратися на ДД типу Mesh, або ДД, яке їх «містить».

Створимо 3D графічну групу з Mesh графічним вузлом.

Results –> Plot Group - 3D Plot Group

3D Plot Group 3 : RMB > Mesh

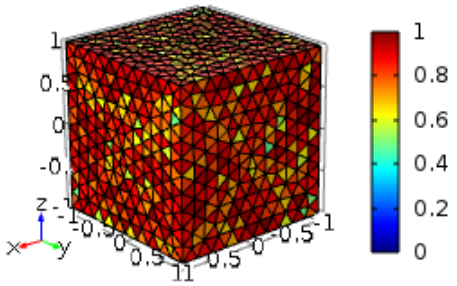
3D Plot Group 3 - Mesh 1 :

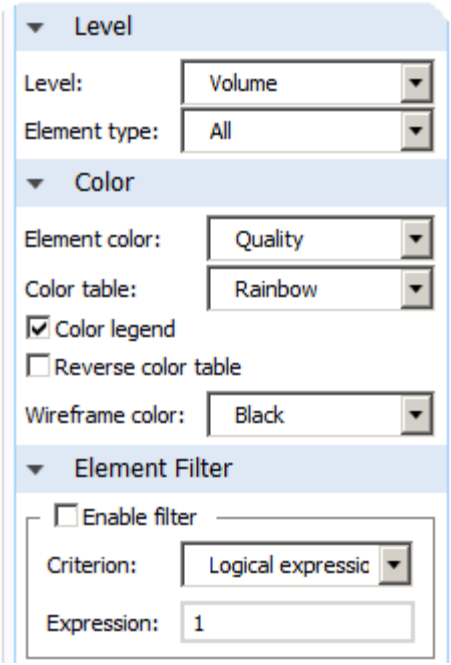
Data – **Data set** = Mesh 1

Level – **Level** = Volume (*фарбувати об'ємні елементи*)

LMB > Plot (*наступний рисунок ліворуч*)

Більшість налаштувань вузла Mesh ми залишили незмінними. Зокрема в списку **Color – Element color** вказано, що для фарбування сітки використовується функція «якості» (Quality).





Зауваження. У графічних вузлах `Volume` і `Mesh` є власні розділи «фільтрації» **`Element Filter`**.

Існує ще два типи джерел даних, використання яких можна тут продемонструвати. Це `Parametric Extrusion 1D` і `Parametric Extrusion 2D` (параметричні розширення). В цих джерелах при наявності в моделі будь-якого непросторового параметра, наприклад, часу, він (параметр) розглядається в якості просторової координати. Так ДД `Parametric Extrusion 2D` дозволяє будувати залиті контурні графіки в паралельних шарах при різних значеннях параметра.

В наступному прикладі ми продемонструємо використання джерела даних `Parametric Extrusion 2D`. Для цього нам знадобиться динамічна модель, щоб мати параметр часу `t`.

Приклад 3. Створіть 2D 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 контроль розмірностей фізичних величин.

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

Геометрія моделі складатиметься з чверті кола. Накресліть його.

`Geometry` -> **`Primitives`** – Circle

Component 1 - Geometry 1 - Circle 1:

`Size and Shape` – **Radius:** = `1`

`Size and Shape` – **Sector angle** = `90`

LMB > Build All Objects

Побудуйте СкЕ сітку.

Mesh 1 : `Mesh Settings` – **Element size** = `Finer`

LMB > Build All (*наступний рисунок ліворуч*)

Вузол **Component 1 (comp1) - Global ODEs and DAEs (ge) - Global Equations 1** зазвичай містить систему ЗДР. В нашому прикладі залишимо його таблицю рівнянь порожньою. В ПВ вузла **Step 1: Time Dependent** в полі **Times** задайте діапазон і інтервал зміни параметра часу `t`.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times** = `range(0,0.1,1)`

Виконайте обчислення.

Study 1 : LMB > Compute

Після звершення обчислень (яких насправді немає, тому що немає рівнянь) в дереві моделі з'явиться вузол **Results - Data Sets - Study 1/Solution 1**, тобто буде створено джерело даних `Study 1/Solution 1`.

В гілку даних **Data Sets** дерева моделі додайте джерело даних `Parametric Extrusion 2D`.

Results - Data Sets: RMB > More Data Sets –> Parametric Extrusion 2D

Data Sets - Parametric Extrusion 2D :

`Data` – **Data set** = `Study 1/Solution 1`

`Data` –**Time selection** = `All`

Виконайте команду Plot і отримайте зображення нового джерела даних, яке наведено на наступному рисунку в центрі На ньому кожний шар представлений четвертиною одиничного кола, і відповідає моментам часу `t=0,0.1,...,1`, які відкладені вздовж вертикальної координати.

Застосуйте це джерело для побудови функціональної забарвлення шарів, використовуючи функцію двох просторових координат `x,y` і часу `t`.

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

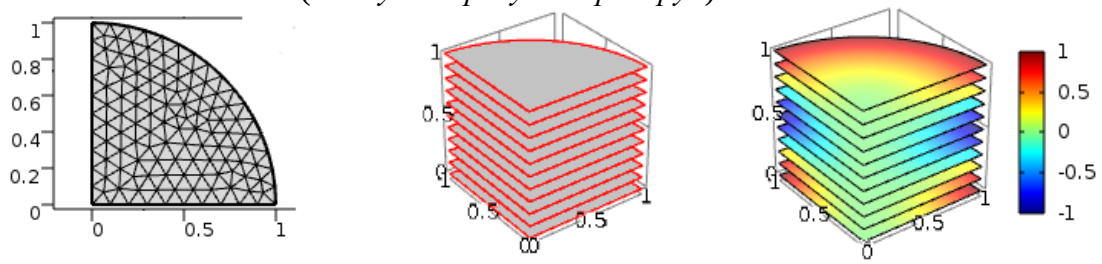
`Data` – **Data set** = `Parametric Extrusion 2D 1`

RMB > Surface

3D Plot Group 1 - Surface 1 :

`Expression` – **Expression** = `(x^2+y^2)*cos(2*pi*t)`

LMB > Plot (*наступний рисунок праворуч*)



Деактивуйте вузол **2D Plot Group 1 - Surface 1** і в гілку **2D Plot Group 1** додайте вузол `Volume`.

3D Plot Group 1 - Surface 1 : RMB > Disable

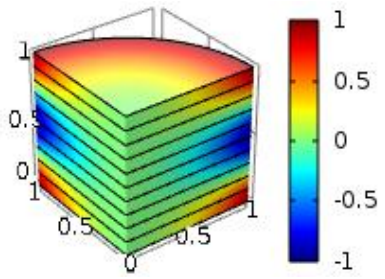
3D Plot Group 1 : RMB > Volume

3D Plot Group 1 - Volume 1 :

`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `(x^2+y^2)*cos(2*pi*t)`

LMB > Plot



Отримане зображення виглядає як тверде тіло, а не набір шарів з попереднього графіка.

Деактивуйте вузол **3D Plot Group 1 - Volume 1**, і у гілку **3D Plot Group 1** додайте `Slice` вузол.

3D Plot Group 1 - Volume 1 : RMB > Disable

3D Plot Group 1 : RMB > Slice

3D Plot Group 1 - Slice 1 :

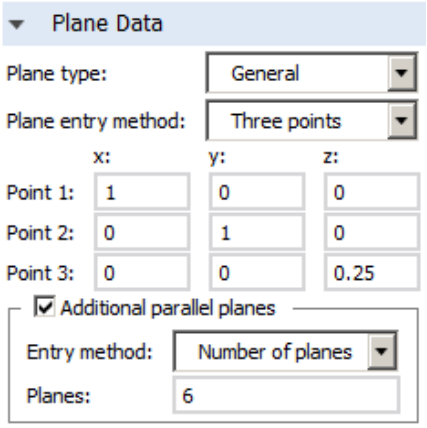
`Data` – **Data set** = `From parent`

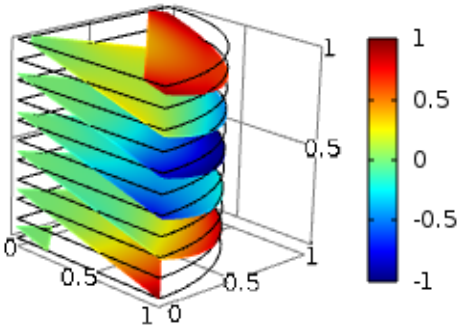
`Expression` – **Expression** = `(x^2+y^2)*cos(2*pi*t)`

`Plane Data` – **Plane type** = `General`

`Plane Data` – **Plane entry method** = `Three points`

Цим ви кажете, що орієнтація сім'ї площин буде визначатися трьома точками. Заповніть поля розділу **`Plane Data`** так, як показано нижче. Виконайте команду Plot і отримайте графік, зображений на наступному рисунку праворуч.





Деактивуйте вузол **2D Plot Group 1 - Slice 1**, і в гілку **2D Plot Group 1** додайте `Contour` вузол.

3D Plot Group 1 - Slice 1 : RMB > Disable

3D Plot Group 1 : RMB > Contour

3D Plot Group 1 - Contour 1 :

`Data` – **Data set** = `From parent`

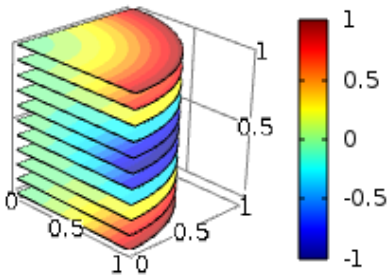
`Expression` – **Expression** = `(x^2+y^2)*cos(2*pi*t)`

`Levels` – **Entry method** = `Number of levels`

`Levels` – **Total levels** = `20`

`Coloring and Style` – **Contour type** = `Filled`

Виконайте команду Plot і отримайте наступний графік, на якому шари розфарбовані смугами.



В наступному прикладі демонструється використання джерела даних `Parametric Extrusion 1D`.

Приклад 4. Створіть 1D модель задачі з динамічним `Global ODEs and DAEs` інтерфейсом.

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

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Геометрія моделі буде складатися з відрізка [0,1].

`Geometry`->**`Primitives`**-> Interval

В ПВ створеного вузла **Interval 1** залиште значення за замовчуванням і завершіть конструювання геометрії натисканням на кнопку Build All Objects .

На створеному відрізку генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Таблицю рівнянь в ПВ вузла **Component 1 (comp1) - Global ODEs and DAEs (ge) - Global Equations 1** залиште порожньою.

В ПВ вузла **Step 1: Time Dependent** в полі **Times** задайте діапазон і інтервал зміни параметра часу `t`.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times** = `range(0,0.025,1)`

Виконайте обчислення.

Study 1 : LMB > Compute

Після звершення обчислень в дереві моделі з'являється вузол **Results - Data Sets - Study 1/Solution 1**, тобто створюється джерело даних `Study 1/Solution 1`.

В гілку даних **Data Sets** додайте джерело даних `Parametric Extrusion 1D`.

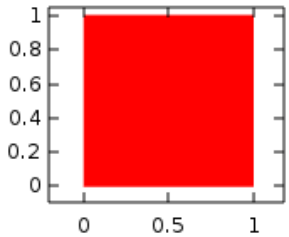
Results - Data Sets : RMB > More Data Sets -> Parametric Extrusion 1D

Data Sets - Parametric Extrusion 1D :

`Data` – **Data set** = `Study 1/Solution 1`

`Data` –**Time selection** = `All`

Виконайте команду Plot і отримайте зображення нового джерела даних у вигляді квадрата $[0,1]\times[0,1]$ на площині $X\times t$ (тут хрестик $\times$ позначає декартовий добуток).



Використайте це джерело даних для побудови функціонального забарвлення відрізків кожного часового шару.

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

2D Plot Group 1 :

`Data` – **Data set** = `Parametric Extrusion 1D 1`

`2D Plot Group 1` –> Add Plot - Line

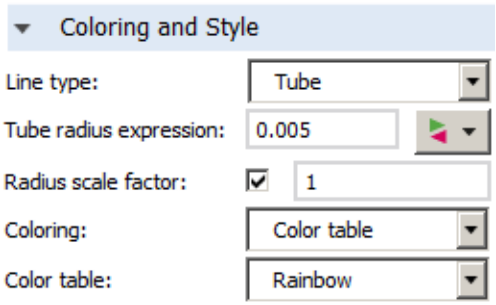
2D Plot Group 1 - Line 1 :

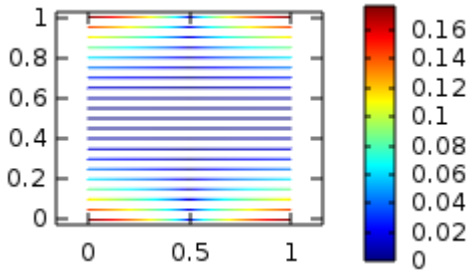
`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `sqrt(abs(x-0.5))*(t-0.5)^2`

В розділі **`Coloring and Style`** зробіть налаштування, які показані нижче.

LMB > Plot





Тут кожний часовий шар зображений горизонтальним відрізком і відповідає моментам часу `t` зі списку `range(0,0.025,1`, який заданий в ПВ вузла **Step 1: Time Dependent** в полі **Times**.

Деактивуйте вузол **2D Plot Group 1 - Line 1**, і у гілку **2D Plot Group 1** додайте `Surface` вузол.

2D Plot Group 1 - Line 1 : RMB > Disable

2D Plot Group 1 : RMB > Surface

2D Plot Group 1 - Surface 1 :

`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `sqrt(abs(x-0.5))*(t-0.5)^2`

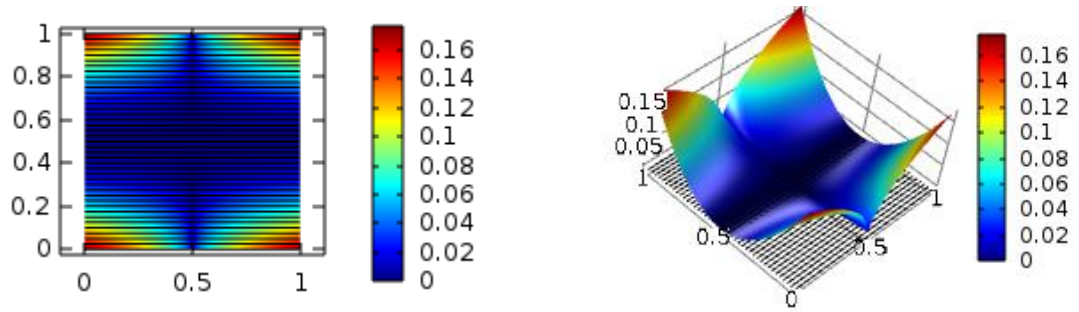
LMB > Plot (*наступний рисунок ліворуч*)

Якщо додати підвузол `Height Expression`, то вийде графік, показаний праворуч.

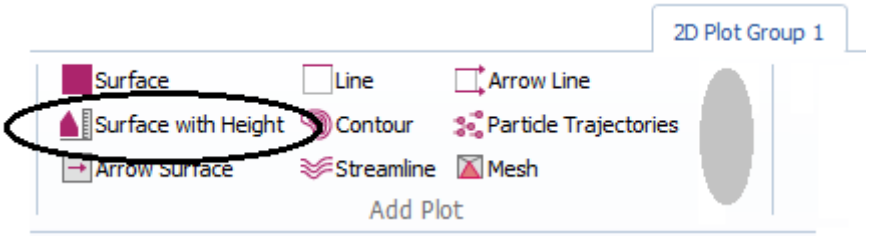
2D Plot Group 1 – Surface 1 : RMB > Height Expression

Surface 1 – Height Expression 1 : `Axis` – **Scale factor** = `True, 3` (*два поля*)

LMB > Plot (*наступний рисунок праворуч*)



Зауваження. Додавання `Surface` вузла з `Height Expression` підвузлом можна виконати одним кліком по кнопці Surface with Height на панелі `2D Plot Group 1`.



Звичайно, що після цього обидва створені вузли потребують налаштування.

В динамічних моделях можна будувати графіки типу `Point Graph`, які призначені для зображення в точці/точках розв'язку/виразу в залежності від параметра часу `t` або `sweep` параметра.

В гілку **Results** додайте гілку `1D Plot Group` і в ній створіть вузол `Point Graph`.

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

1D Plot Group 2 :

`Data` – **Data set** = `Study 1/Solution 1`

`1D Plot Group 2` –> Add Plot - Point Graph

1D Plot Group 2 - Point Graph 1 :

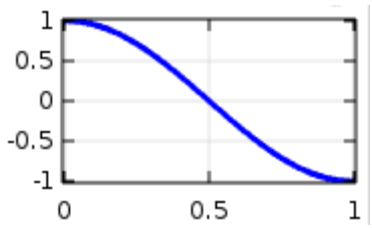
`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `x*cos(pi*t)`

В секції вибору натисніть кнопку Active, щоб можна було вводити індикатори точок, і клацніть в графічному вікні по правому кінцю відрізка. Індикатор точки буде додано в секцію вибору.

`Selection` – **Selection, <>** = `Manual, 2` (*правий кінець відрізка*)

Виконайте команду Plot і отримайте графік ділянки косинусоїди, де вздовж горизонтальної осі відкладено час.



Точка, в якій будується графік, може бути задана вузлом, існуючим в геометрії моделі, або ДД `Cut Point`.

■

Наведемо приклад використання джерела даних `Revolution 2D`. Нехай задана яка-небудь двовимірна область Ω (наприклад, двовимірне джерело даних `Mesh`) і розв'язок/вираз в ньому. За допомогою джерела `Revolution 2D` створюється тривимірна область, одержувана обертанням вихідної двовимірної області Ω навколо деякої осі. При цьому двовимірний розв'язок/вираз з Ω осесиметрично продовжується в створювану тривимірну зону.

Приклад 5. Створіть шаблон порожньої моделі `Blank Model`.

File -> New -> Blank Model

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Геометрія компонента складається з одного прямокутника.

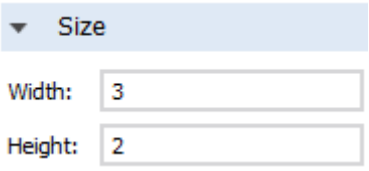
`Geometry` -> **`Primitives`** – Rectangle

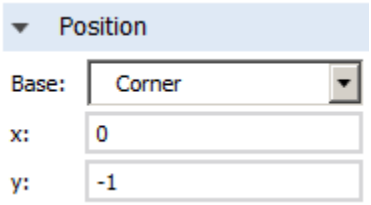
Component 1 - Geometry 1 - Rectangle1:

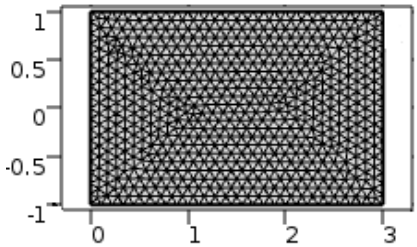
`Size` – **Width, Height** = `3,2` (*два поля*)

`Position` – **Base, x, y** = `Corner,0,-1` (*три поля*)

Тут в розділі **`Size`** задаються ширина і висота прямокутника. А в розділі **`Position`** вводяться координати його вершини.







LMB > Build All Objects

В графічній зоні з'явиться зображення прямокутника. Побудуйте на ньому СкЕ сітку.

Mesh 1 : **`Mesh Settings`** – **Element size** = `Finer`

LMB > Build All (*попередній рисунок праворуч*)

Тепер створіть джерела даних `Mesh` і `Revolution 2D`.

`Results` -> Data Set - More Data Sets -> Mesh

Data Sets - Mesh 1 : **`Mesh`** – **Mesh** = `Mesh 1`

`Results` -> Data Set - More Data Sets -> Revolution 2D

Data Sets - Revolution 2D 1 :

`Data` – **Data set** = `Mesh 1`

`Axis Data` – **Axis entry method** = `Two points`

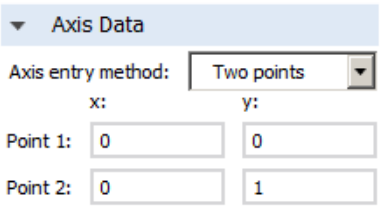
Цим визначається, що вісь обертання буде задаватися двома точками. Введіть координати цих точок `(0,0)` і `(0,1)`.

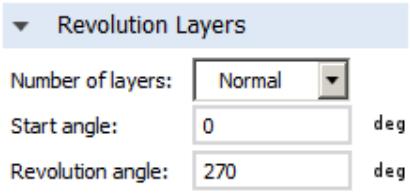
`Axis Data - Point 1` – **x, y** = `0,0` (*два поля*)

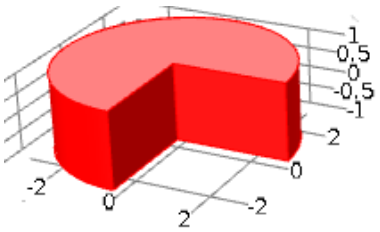
`Axis Data - Point 2` – **x, y** = `0,1` (*два поля*)

`Revolution Layers` – **Revolution angle** = `270` (*кут обертання*)

LMB > Plot (*наступний рисунок праворуч*)







Подивимось, як таке ДД можна використовувати. Створіть групу тривимірних графіків.

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

3D Plot Group 1 :

`Data` – **Data set** = `Revolution 2D 1`

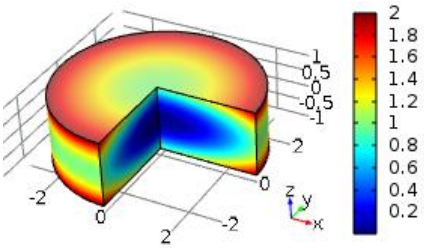
RMB > Volume

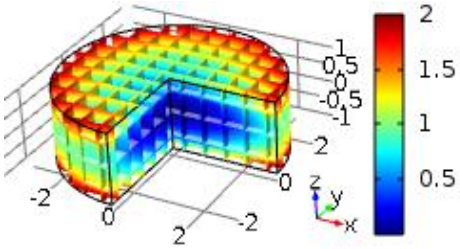
3D Plot Group 1 - Volume 1 :

`Expression` – **Expression** = `x^2/9+y^2`

Вираз `x^2/9+y^2` визначає функціональне забарвлення джерела даних `Revolution 2D 1`, тобто поверхні циліндричного сектора.

LMB > Plot (*наступний рисунок ліворуч*)





Зверніть увагу на те, що напрямок осі Y в площині прямокутника не збігається з напрямком осі Y тривимірної системи координат.

Замість `Volume` графіка ви можете побудувати графіки `Surface`, `Slice` або `Multislice`. Наприклад, у вузлі **3D Plot Group 1** деактивуйте вузол **Volume 1** і створіть `Multislice` вузол.

3D Plot Group 1 - Volume 1 : RMB > Disable

3D Plot Group 1 : RMB > More Plots - Multislice

3D Plot Group 1 - Multislice 1 :

`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `x^2/9+y^2`

В розділі **`Multiplane Data`** в списках **Entry method** для всіх трьох напрямків залиште значення за замовчуванням `Number of planes`.

`Multiplane Data` – **x-planes - Planes** = `10`

– **z-planes - Planes** = `10`

– **y-planes - Planes** = `3`

LMB > Plot (*попередній рисунок праворуч*)

Зверніть увагу на назви сімей площин **x-planes, z-planes, y-planes**, які не співпадають з назвами просторових координат на рисунку.

■

Розглянемо ще одне цікаве застосування обчислюваних джерел даних `Study 1/Solution 1`. Воно полягає в тому, що існує можливість створювати

копію цих вузлів зі зміною рівня геометрії та з використанням комплекснозначного множника. Якщо, наприклад, базисне ДД `Study 1/Solution 1` діє в двовимірній зоні, то можна створити його копію і вказати, що вона (копія `Study 1/Solution 1 (2)`) діє на якійсь підмножині базисної зони (в підобласті, на лініях або точках). Це схоже на використання вузлів типу `Cut Plane` або `Cut Line`, проте внутрішній «переріз» може мати будь-яку форму, наприклад, бути параметричною поверхнею або кривою.

Продемонструємо вказані можливості на прикладі розв'язку крайової задачі в області Ω

$$\Delta u = 16 \cdot \left(x^2 + y^2\right), \quad u\big|_{\partial\Omega} = 1,$$

де Ω позначає круг одиничного радіуса з центром в початку координат. Точний розв'язок цієї задачі відомий $u = \left(x^2 + y^2\right)^2$, і ми зможемо порівнювати його (та його «частини») з розв'язком, отриманим за допомогою COMSOL Multiphysics.

Приклад 6. За допомогою майстра створіть 2D модель задачі

File->New -> Model Wizard -> 2D -> Mathematics -
Classical PDEs - Poisson's Equation (poeq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Геометрія задачі складається з кола одиничного радіуса і внутрішньої спіралі.

`Geometry->`**`Primitives->`** Circle

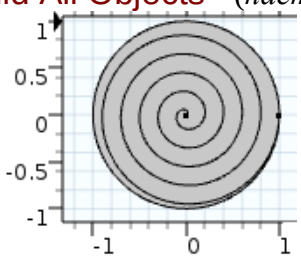
`Geometry->`**`Primitives->`** Parametric Curve

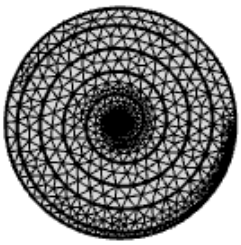
Geometry 1 - Parametric Curve 1 (pc1) :

`Parameter` – **Name, Minimum, Maximum** = `s, 0, 10*pi` (*три поля*)

`Expressions` – **x, y** = `s*cos(s)/10/pi, s*sin(s)/10/pi` (*два поля*)

LMB > Build All Objects (*наступний рисунок ліворуч*)





Генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All (*попередній рисунок праворуч*)

Задайте рівняння. Перевірте, що в ПВ гілки **Poisson's Equation (poeq)** в розділі **`Domain Selection`** в секції вибору стоїть 1, тобто обрано коло. В дереві моделі виділіть вузол **Poisson's Equation 1**. В його ПВ в розділі **`Equation`** наведено рівняння Пуассона.

$$\nabla \cdot \left(-c \nabla u\right) = f, \quad \nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right].$$

Задайте параметри рівняння

`Diffusion Coefficient` – **c** = `-1`

`Source Term` – **f** = `16*(x^2+y^2)`

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection–Selection,<> =Manual,1,2,3,4 (*граничне коло*)

Dirichlet Boundary Condition – r = 1

Виконайте обчислення.

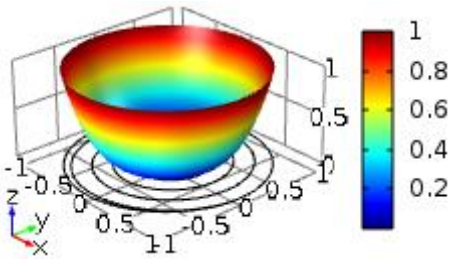
Study 1 : LMB > Compute

Після завершення обчислень в графічній зоні вузла **2D Plot Group 1 – Surface 1** буде побудована розрахункова область з функціональним забарвленням, відповідним розв'язку *u*. Шкала кольорів поруч показуватиме діапазон зміни значень. У вузол **Surface 1** додайте підвузол Height Expression.

2D Plot Group 1 – Surface 1 : RMB > Height Expression

Surface 1 – Height Expression 1 : Axis – Scale factor = True,1 (*два поля*)

LMB > Plot



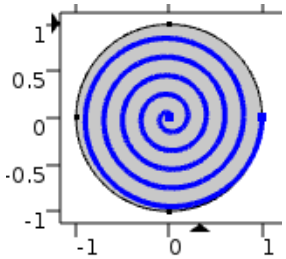
Графік розв'язку вздовж спіралі можна побудувати, якщо створити копію вузла Study 1/Solution 1, в якій змінити рівень геометрії.

Data Sets – Study 1/Solution 1 : RMB > Duplicate

Data Sets – Study 1/Solution 1 (2) : RMB > Selection

Geometric Entity Selection – Geometric entity level = Boundary

Geometric Entity Selection – Selection,<> =Manual,5 (*спіраль*)



Тепер побудуйте графік розв'язку u вздовж спіралі.

Results -> Plot Group - 2D Plot Group

2D Plot Group 2-> Add Plot – Line

2D Plot Group 2 - Line 1 :

Data – Data set = Study 1/Solution 1 (2) (*розв'язок на спіралі*)

Expression – Expression =u

Налаштуйте стиль кривої.

Coloring and Style – Line Type =Tube

–Tube radius expression =0.03

– Radius scale factor =True,1

Додайте підвузол Height Expression, в якому задайте вертикальне зміщення таким, щоб крива розташовувалась на поверхні сфери.

2D Plot Group 2- Line 1 : RMB > Height Expression

Line 1 – Height Expression 1 :

Expression – **Height data** = Expression

Expression – **Expression** = sqrt(1-x^2-y^2)

Axis – **Scale factor** = True, 1 (*два поля*)

Додайте зображення сфери.

2D Plot Group 2-> Add Plot –Surface

2D Plot Group 2 - Surface 1 :

Data – **Data set** = Study 1/Solution 1 (1) (*коло*)

Expression – **Expression** = 1

Coloring and Style – **Color table** =GrayScale

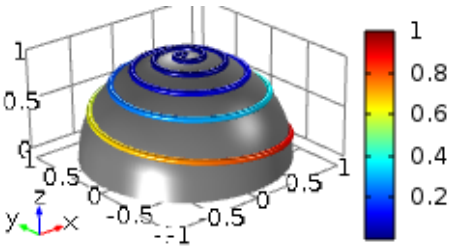
Surface 1 : RMB > Height Expression

Expression – **Height data** = Expression

Expression – **Expression** = sqrt(1-x^2-y^2)

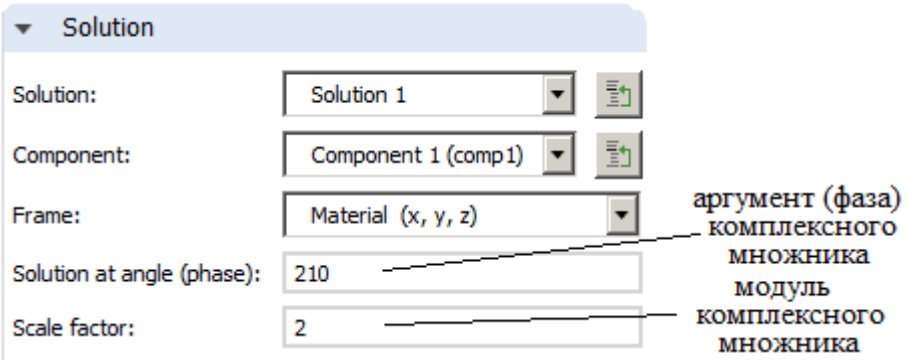
Axis – **Scale factor** = True, 1 (*два поля*)

LMB > Plot



Отже для побудови кривої використано ДД Study 1/Solution 1 (2), яке складається з «перенесених» на спіраль даних розв'язку рівняння Пуассона (спіраль зафарбована відповідно до значень розв'язку u). Сфера зображена «для краси».

Продемонструємо використання комплекснозначних множників в ПВ вузлів Study 1/Solution 1.



Комплекснозначний множник використовується для побудови змінної U, яка обчислюється із істинного розв'язку u по формулі $U = k \cdot e^{i\pi\varphi/180} \cdot u$, де k – модуль комплексного множника (дійсне число), а φ - його аргумент (кут в градусах). Модуль k вводиться в полі **Scale factor**, а аргумент φ - в полі **Solution at angle (phase)**. Наприклад, якщо k=1, а $\varphi = 180$, то отримаємо $U = e^{i\pi} \cdot u = -u$. Графічні і обчислювальні вузли, які спиратимуться на ДД з

комплексним множником, будуть отримувати не розв'язок *u*, а нову змінну *U*, до якої в похідних вузлах потрібно буде звертатися по попередньому імені *u*.

Створимо копію ДД `Study 1/Solution 1 (1)` (одиниця в дужках додана до назви основного ДД автоматично після створення копії вздовж спіралі), яка буде мати комплексний множник з модулем *k*=2 і аргументом 210 градусів.

Data Sets – Study 1/Solution 1 (1) : RMB > Duplicate

Data Sets – Study 1/Solution 1 (3) :

`Solution` – Solution at angle (phase) = `210` (*кут в градусах*)

`Solution` – Scale factor = `2` (*множник*)

Рівнем дії залишається вся геометрія.

Порівнювати краще одновимірні графіки, тому розв'язки будемо розглядати вздовж горизонтального діаметра розрахункової області. Для нього побудуємо два `Cut Line` джерела даних. Одне буде «вибирати» дані з основного ДД `Study 1/Solution 1 (1)`, друге – з ДД з комплексним множником `Study 1/Solution 1 (3)`.

Data Sets : RMB > Cut Line 2D

Data Sets – Cut Line 2D 1 :

`Data` – Data set = `Study 1/Solution 1 (1)`

`Line Data` – Line entry method = `Two points` (*лінія по двох точках*)

– **Point 1 : x, y**= `-1,0` (*два поля*)

– **Point 2 : x, y**= `1,0` (*два поля*)

Data Sets – Cut Line 2D 1 : RMB > Duplicate

Data Sets – Cut Line 2D 2 :

`Data` – Data set = `Study 1/Solution 1 (3)`

Тепер побудуємо вздовж діаметра графіки розв'язку `u` і розв'язку з комплексним множником. Оскільки другий розв'язок буде комплекснозначним, то має сенс будувати окремо графіки його дійсної і уявної частин.

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

`1D Plot Group 3`-> Add Plot – Line Graph

1D Plot Group 3 - Line Graph 1: (*графік розв'язку крайової задачі*)

`Data` – Data set = `Cut Line 2D 1`

`y-Axis Data` – Expression =`u`

`x-Axis Data` – Parameter, Expression = `Expression,x` (*два поля*)

`Coloring and Style` – Width = `5`

Задайте позначку біля кривої (легенду).

`Legends` – Show Legends, Legends = `True,Manual` (*два поля*)

`Legends` –Legends = `u` (*легенда, ведення в таблиці*)

1D Plot Group 3 - Line Graph 1 : RMB > Duplicate

1D Plot Group 3 - Line Graph 2 :

`Data` – Data set = `Cut Line 2D 2` (*розв'язок з комплексним множником*)

`y-Axis Data` – Expression =`real(u)`

`Legends` –Legends = `Re(u)` (*легенда, ведення в таблиці*)

1D Plot Group 3 - Line Graph 2 : RMB > Duplicate

1D Plot Group 3 - Line Graph 3 : (*графік розв'язку з комплексним множником*)

y-Axis Data – **Expression** =`imag(u)`

Legends –**Legends** = `Im(u)` (*легенда, ведення в таблиці*)

1D Plot Group 3: Legend –**Position** = `Lower right`

LMB > Plot (*наступний рисунок*)

Порівняємо графіки з виразами точного розв'язку $u=\left(x^2+y^2\right)^2$, і змінною $U=2\cdot e^{i\pi 210/180}\cdot u=-2e^{-i\pi/6}\cdot\left(x^2+y^2\right)^2$. Для цього додайте в групу **1D Plot Group 3** графіки цих виразів (для U – дійсної і уявної частин).

`1D Plot Group 3`-> Add Plot – Line Graph

1D Plot Group 3 - Line Graph 4:

Data – **Data set** = `Cut Line 2D 1`

y-Axis Data – **Expression** = `(x^2+y^2)^2`

x-Axis Data – **Parameter, Expression** = `Expression,x` (*два поля*)

Coloring and Style – **Width, Color** = `2, Black`

Legends – **Show Legends** = `False`

1D Plot Group 3 - Line Graph 4 : RMB > Duplicate

1D Plot Group 3 - Line Graph 5:

y-Axis Data – **Expression** =`-2*real((x^2+y^2)^2*exp(i*pi/6))`

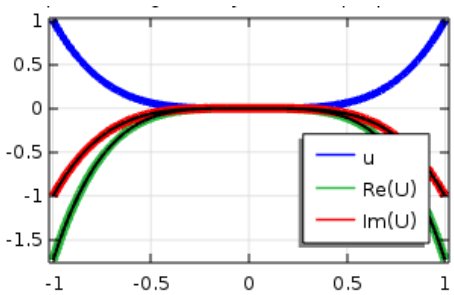
1D Plot Group 3 - Line Graph 5 : RMB > Duplicate

1D Plot Group 3 - Line Graph 6:

y-Axis Data – **Expression** =`-2*imag((x^2+y^2)^2*exp(i*pi/6))`

LMB > Plot

Чорні криві, які зображають графіки аналітичних виразів, накладаються на попередні.



Зауваження. У вузлі **1D Plot Group 3 - Line Graph 5** використання функції `real(...)` в формулі необов'язкове. При побудові графіків комплекснозначних виразів система автоматично використовує їх дійсну частину.

2.1.4. Траєкторії точок, лінії течії і анімація.

В цьому параграфі наведені приклади роботи з графіками `Point Trajectories`, `Particle Tracing`, `Particle Tracing with Mass` і `Streamline`. Перші три зазвичай використовуються в динамічних моделях з параметром часу `t`. У вузлі `Point Trajectories` будуються траєкторії точок

по їх параметричним рівнянням $x(t), y(t)$ (та $z(t)$ в 3D випадку). У вузлах `Particle Tracing` задається векторне поле швидкостей безмасових частинок $\boldsymbol{V}(t,\boldsymbol{r})$, розв'язуються рівняння $\dot{\boldsymbol{r}} = \boldsymbol{V}(t,\boldsymbol{r})$, і по розв'язку будуються траєкторії. У вузлі `Particle Tracing with Mass` задається векторне силове поле $\boldsymbol{F}(t,\boldsymbol{r},\dot{\boldsymbol{r}})$, яке діє на частинки, розв'язуються рівняння $m\ddot{\boldsymbol{r}} = \boldsymbol{F}(t,\boldsymbol{r},\dot{\boldsymbol{r}})$, і по розв'язку будуються траєкторії. У вузлах типу `Streamline` будуються лінії векторного поля $\boldsymbol{V} = (V_x, V_y, V_z)$, які визначаються рівностями $\frac{dx}{V_x} = \frac{dy}{V_y} = \frac{dz}{V_z}$.

Приклад 1. Побудуємо траєкторії двох точок, використовуючи графічні вузли `Point Trajectories`.

Створіть шаблон порожньої моделі `Blank Model`.

File -> New -> Blank Model

Додайте в модель двовимірний компонент.

`Home` -> Model -> Add Component – 2D

Геометрія компонента складатиметься з одного квадрата.

`Geometry` -> **`Primitives`** – Square

Geometry 1 – Square 1:

`Size` – **Side length** = `2`

`Position` – **Base** = `Corner`

`Position` – **x, y** = `-1,-1`

На квадраті генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Додайте в модель `Time Dependent` вузол дослідження.

Untitled.mph (root): RMB > Add Study = `Add Study` ->
Preset Studies - Time Dependent ->X

Задайте діапазон і інтервал зміни параметра часу `t`.

Study 1 - Step 1: Time Dependent:

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

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено вузол джерела даних **Study 1/Solution 1**.

Створіть двовимірну графічну групу.

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

2D Plot Group 1 :

`Data` – **Data set** = `Study 1/Solution 1`

`Data`– **Time (s)** = `6.2`

Наступні графіки `Point Trajectories` будуть малюватися при зміні параметра часу `t` від початкового значення діапазону `range(0,0.1,2*pi)` (в нашому прикладі від нуля), до значення `6.2`, обраного в списку **Time (s)**. Ви можете вибрати будь-яке значення з цього списку.

Plot Settings – **Plot data set edges** = False

В графічну групі **2D Plot Group 2** додайте вузол Point Trajectories, в якому буде малюватися траєкторія першої точки.

2D Plot Group 1 -> **More Plots** -> Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

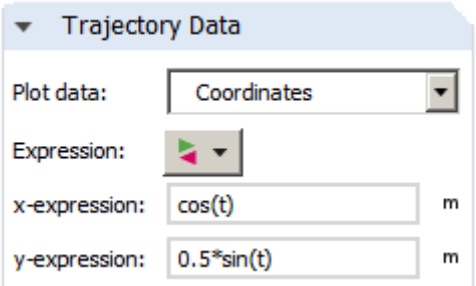
Data– **Data set** = From Parent

Trajectory Data– **Plot data** = Coordinates

Опція **Plot data**=Coordinates означає, що траєкторія точки буде задаватися параметричними рівняннями $x(t), y(t)$ в наступних полях виразів.

Trajectory Data– **x-expression** = cos(t)

– **y-expression** = 0.5*sin(t)



Налаштуйте стиль кривої.

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

– **Line style**– **Tube radius expression** =0.02

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

– **Color** =Blue

Додайте ще один вузол Point Trajectories, призначений для побудови траєкторії другої точки.

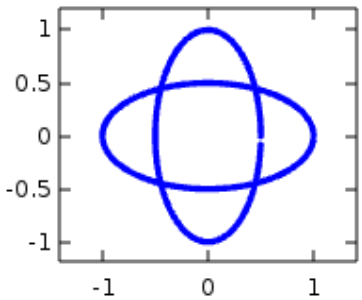
2D Plot Group 1 - Point Trajectories 1 : RMB > Duplicate

2D Plot Group 1 - Point Trajectories 2 :

Trajectory Data – **x-expression** = 0.5*cos(t)

– **y-expression** = sin(t)

LMB > Plot



Збережіть файл з ім'ям, наприклад, PointTrajectories_1.mph, але не закривайте його.

Розглянемо роботу з графіками типу Particle Tracing (трасування частинок). Вони використовується для побудови траєкторій безмасових частинок з заданим полем швидкостей.

Розглянемо двовимірний випадок. Нехай поле (вектор) швидкостей має вигляд $\boldsymbol{V}=(V_x(t,x,y),V_y(t,x,y))$. Тоді радіус-вектор частинки $\boldsymbol{r}(t)=(x(t),y(t))$ задовольнятиме системі диференціальних рівнянь $\boldsymbol{r}_t'=\boldsymbol{V}(t,\boldsymbol{r})$, яка в покоординатній формі матиме вигляд

$$\begin{cases} \dfrac{dx}{dt}=V_x(t,x,y) \\ \dfrac{dy}{dt}=V_y(t,x,y) \end{cases}, \qquad (1)$$

і до якої потрібно додати початкове положення частинки $x(0)=x_0$, $y(0)=y_0$.

В COMSOL для побудови траєкторій частинок, координати яких задовольняють системі (1), можна використовувати `Particle Tracing` графіки. Звичайно, що в їх налаштуваннях треба задавати векторне поле $\boldsymbol{V}(t,\boldsymbol{r})$, та початкові положення точок (x_0,y_0), яких може бути багато.

Побудуємо траєкторію однієї частинки, яка спочатку розташована в точці $(x_0,y_0)=(1,0)$, і має вектор швидкостей $\boldsymbol{V}=(-y,x)$. Ми заздалегідь знаємо відповідь, це коло одиничного радіуса $x=\cos t$, $y=\sin t$.

Додайте в файл `PointTrajectories_1.mph`, двовимірну графічну групу.

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

2D Plot Group 2 :

`Data` – **Data set** = `Study 1/Solution 1`

`Data`– **Time (s)** = `6.2`

`Plot Settings` – **Plot data set edges** = `False`

В графічну групі **2D Plot Group 2** додайте вузол `Particle Tracing`.

`2D Plot Group 2` -> **`More Plots`** -> Particle Tracing

2D Plot Group 2 - Particle Tracing 1 :

`Data`– **Data set** = `From Parent`

`Expression` – **x component** = `-y`

– **y component** = `x`

`Particle Positioning` – **Positioning** = `Start point controlled`

– **x, y** = `1,0` (*два поля*)

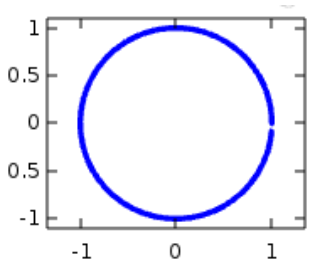
`Coloring and Style` – **`Line style`**–**Type** =`Tube`

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

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

– **Color** =`Blue`

LMB > Plot



Таким чином, у вузлі `Particle Tracing` було розв'язано систему диференціальних рівнянь $x'_t = -y$, $y'_t = x$ з початковою умовою $x(0)=1$, $y(0)=0$ і побудовано графік розв'язку на інтервалі часу, заданому в полі **`Data`–Time (s)** графічної групи **2D Plot Group 1**.

Побудуємо графік трасування кількох частинок в новій графічній групі.

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

2D Plot Group 3 :

`Data` – **Data set** = `Study 1/Solution 1`

`Data`– **Time (s)** = `6.2`

`Plot Settings` – **Plot data set edges** = `False`

`2D Plot Group 3` -> **`More Plots`** -> Particle Tracing

2D Plot Group 3 - Particle Tracing 1 :

`Expression` – **x component** = `x*(t+0.5)/8`

– **y component** = `y*sqrt(t)/8`

Тут векторне поле залежить не тільки від поточного положення точок (`x,y`), а також від часу `t`.

Спочатку `12` точок будуть розташовано на колі радіуса `1/3`, утворюючи одне з одним кути `pi/6`.

`Particle Positioning` – **Positioning** = `Start point controlled`

– **x** = `cos(range(0,pi/6,2*pi))/3`

– **y** = `sin(range(0,pi/6,2*pi))/3`

Налаштуйте стилі кривих.

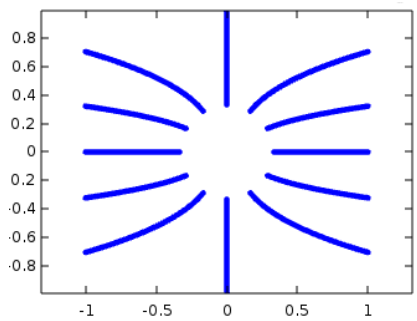
`Coloring and Style` – **`Line style`**–**Type** =`Tube`

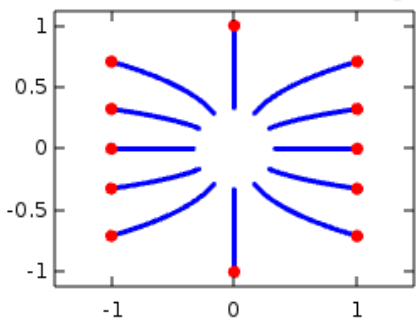
– **`Line style`**–**Tube radius expression** =`0.01`

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

– **Color** =`Blue`

LMB > Plot (*наступний рисунок ліворуч*)





Графіки «траєкторій» одночасно з кривими можуть відображати точки, які їх описують (кінцеві точки кривих). Для цього в розділі **`Coloring and Style`** є секція **Point style**. За замовчуванням відображення точок відключено. Поверніться до вузла **Particle Tracing 1** і «включіть» точки.

2D Plot Group 3 - Particle Tracing 1 :

`Coloring and Style` – **Point style** –**Type** = `Point`

– **Point style** –**Point radius expression** = `0.05`

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

– **Point style** – **Color** = Red

LMB > Plot (*попередній рисунок праворуч*)

Ще раз збережіть файл, але не закривайте його, бо ми продовжимо роботу з ним далі.

Проаналізуємо роботу з графіками типу Particle Tracing with Mass (трасування частинок з масою). Вони використовуються для побудови траєкторій частинок, на які діє відоме зовнішнє силове поле. Знову для спрощення розглянемо двовимірний випадок.

Нехай силове поле має вигляд $\boldsymbol{F} = \left(F_x(t, x, y, \dot{x}, \dot{y}), F_y(t, x, y, \dot{x}, \dot{y})\right)$. Тоді радіус-вектор частинки $\boldsymbol{r}(t) = (x(t), y(t))$ задовольнятиме системі диференціальних рівнянь $m \cdot \boldsymbol{r}''_{tt} = \boldsymbol{F}(t, \boldsymbol{r}, \dot{\boldsymbol{r}})$, яке в покоординатній формі матиме вид

$$\begin{cases} m\dfrac{d^2 x}{d t^2} = F_x(t, x, y, \dot{x}, \dot{y}) \\ m\dfrac{d^2 y}{d t^2} = F_y(t, x, y, \dot{x}, \dot{y}) \end{cases}, \qquad (2)$$

Для однозначного визначення положення частинки до системи потрібно додати початкове положення частинки $x(0) = x_0$, $y(0) = y_0$ і її початкову швидкість $x'(0) = u_0$, $y'(0) = v_0$.

В COMSOL для побудови траєкторій частинок, координати яких задовольняють системі (2), можна використовувати Particle Tracing with Mass графіки. Звичайно, що в їх налаштуваннях треба задавати векторне поле $\boldsymbol{F}(t, \boldsymbol{r}, \dot{\boldsymbol{r}})$, та початкові умови: положення точок (x_0, y_0), та їх швидкість (u_0, v_0). Пояснимо роботу з цими графіками на прикладі.

В останній файл додайте нову двовимірну графічну групу і в ній створіть вузол типу Particle Tracing with Mass.

Results -> Plot Group - 2D Plot Group

2D Plot Group 4 :

Data – **Data set** = Study 1/Solution 1

Data– **Time (s)** = 6.2

Plot Settings – **Plot data set edges** = False

2D Plot Group 4 -> **More Plots** -> Particle Tracing with Mass

2D Plot Group 4 - Particle Tracing with Mass 1 :

Equation of Motion – **Fx** = -x

Equation of Motion – **Fy** = -y

Тут силове векторне поле (Fx,Fy) може залежити не тільки від поточного положення точок (x,y), а також від часу t.

В розділі **Mass and Velocity** задаються маси частинок і їх початкові швидкості.

Mass and Velocity – **Mass** = 1 (*маса може бути виразом*)

– **Initial velocity – x-component** = 0

– **Initial velocity – y-component** = 0.5

В розділі **Particle Positioning** задаються точки (початкове положення), траєкторії яких будуть зображатися.

Particle Positioning – **Positioning** = Start point controlled

– **x, y** = 1,0 (*два поля*)

Далі іде налаштування зовнішнього вигляду кривих.

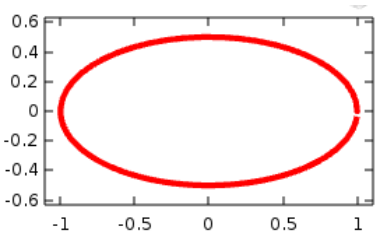
Coloring and Style – **Line style**–**Type** =Tube

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

– **Line style**– **Radius scale factor** =1

– **Color** =Red

LMB > Plot

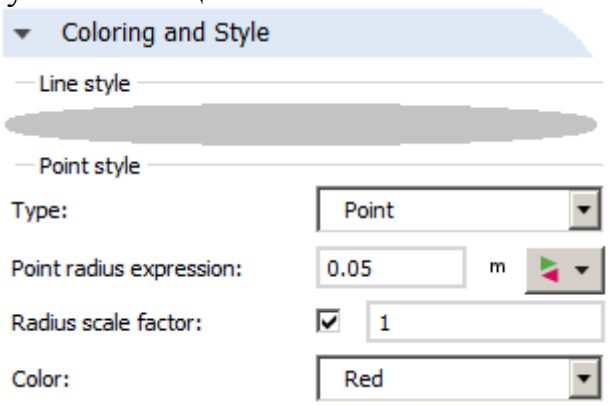


■

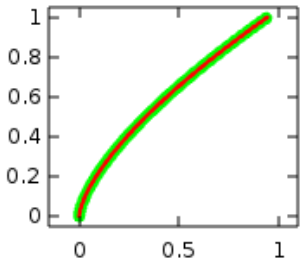
Зауваження 1. При роботі з графіками Particle Tracing with Mass можна одночасно будувати траєкторії для багатьох точок. Для цього треба в розділі **Particle Positioning** задати списки абсцис і ординат точок.

Зауваження 2. В тривимірних випадках додається третє поле z, а всі інші міркування залишаються незмінними.

Зауваження 3. Всі три типа графіків Point Trajectories, Particle Tracing і Particle Tracing with Mass в розділах **Coloring and Style** мають секції **Point style**, в яких можна включити (**Type**=Point) і налаштувати стиль відображення рухомих/кінцевих точок.



Вправа. Побудуйте графік кривої, яка задана в ДД Parameterized Curve 2D параметричними рівняннями $x = \frac{1}{3}s^3$, $y = \frac{1}{2}s^2$. Побудуйте траєкторію безмасової точки (0,0) по рівнянням $x'_t = t^2$, $y'_t = t$. Переконайтеся, що криві співпадають.



Створимо модель, в якій буде динамічно малюватися крива Ліссажу.

Приклад 2. Створіть шаблон порожньої моделі.

File -> New -> Blank Model

Створіть функції $x_c(t) = \cos t, y_c(t) = \sin 3t$, які будуть представляти параметричні рівняння кривої Ліссажу.

Global Definitions : RMB > Functions – Analytic

Function name = xc

Definition –**Expression, Arguments** = cos(t),t (*два поля*)

Global Definitions : RMB > Functions – Analytic

Function name = yc

Definition – **Expression, Arguments** = sin(3*t),t (*два поля*)

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Геометрія компонента складається з одного прямокутника. Він буде репрезентувати зону, в якій буде малюватися крива.

Geometry -> **Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

Size – **Width, Height** = 3,3 (*два поля*)

Position – **Base, x, y** = Center,0,0 (*три поля*)

На прямокутнику побудуйте СкЕ сітку.

Mesh 1 : Mesh Settings – **Element size** = Finer

Додайте в модель вузол обчислення.

Home -> Study - Add Study

В панелі Add Study, яка з'явиться збоку, виберіть Time Dependent тип дослідження, і закрийте панель.

Задайте діапазон і інтервал зміни параметра часу t.

Study 1 - Step 1: Time Dependent:

Study Settings –**Times** = range(0,0.1,8)

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено вузол джерела даних **Study 1/Solution 1**.

Створіть двовимірну графічну групу 2D Plot Group.

Results -> Plot Group - 2D Plot Group

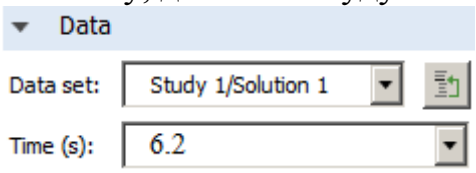
2D Plot Group 1 :

Data – **Data set** = Study 1/Solution 1

Data– **Time (s)** = 6.2

Plot Settings – **Plot data set edges** = False

Зверніть увагу на те, що при виборі динамічного джерела даних Study 1/Solution 1 (яке «містить» час), в розділі **Data** з'являється поле-список **Time (s)** з моментом часу, для якого будуватиметься графік.



В графічній групі **2D Plot Group 1** створіть вузол Point Trajectories, який буде містити зображення кривої (траєкторію руху точки) та самої точки.

2D Plot Group 1-> **More Plots** -> Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

Data– **Data set** = From Parent

Trajectory Data – **Plot data** = Coordinates

– **x-expression** = xc(t)

– **y-expression** = yc(t)

Графік Point Trajectories будується за функціями xc(t), yc(t) при зміні параметра t від початкового значення діапазону range(0,0.1,8) (в нашому прикладі від нуля), до значення 6.2, обраного в списку **Time (s)** ПВ батьківського вузла. Ви можете вибрати будь-яке значення зі списку.

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

– **Line style**– **Tube radius expression** =0.02

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

– **Color** =Blue

Щоб зобразити рухому точку, треба налаштувати секцію **Point style**.

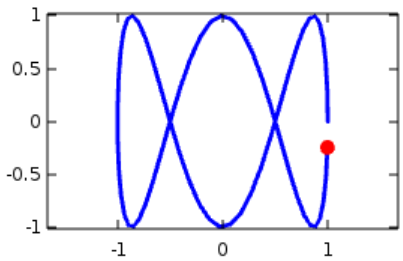
Coloring and Style – **Point style** –**Type** = Point

– **Point style** –**Point radius expression** = 0.07

– **Point style** – **Radius scale factor** = True,1

– **Point style** – **Color** = Red

LMB > Plot

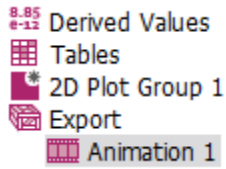


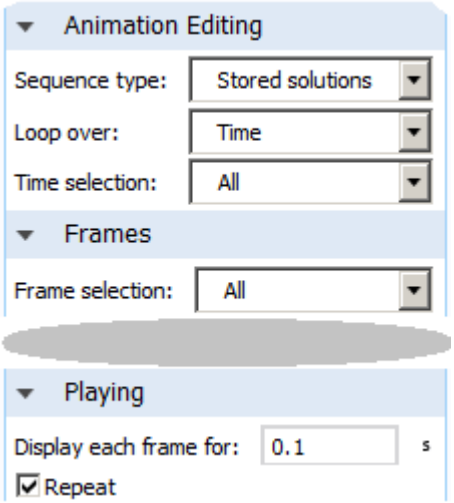
Тепер побудуйте анімацію. Для цього в гілку **Results - Export** командою

Results -> Animation -> Player

додайте вузол, який автоматично отримає ім'я **Animation 1**. В його полі **Scene-Subject** виберіть 2D Plot Group 1. Це означає, що анімація буде містити кадри зображень, які побудовані в графічній групі **2D Plot Group 1** в різні

моменти часу `t`. Значення інших полів ПВ вузла **Animation 1** задайте так, як показано на рисунку.

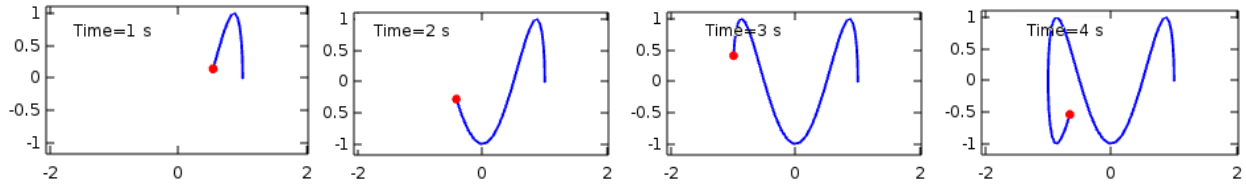




Тут в списку **Time selection** вибрано значення `All`, що означає побудову анімації на всьому інтервалі часу `0≤t≤8`. В списку **Frame selection** теж вибрано значення `All`, що означає генерування кадрів анімації для всіх моментів часу з діапазону `range(0,0.1,8)`. Сам діапазон заданий в ПВ вузла **Study 1 - Step 1: Time Dependen**. В полі **Display each frame for** вводиться інтервал часу між кадрами анімації. Прапорець **Repeat** включає повторення анімації.

Вгорі ПВ вузла **Animation 1** натисніть кнопку Generate Frame. Почнеться процес генерації кадрів, який ви можете спостерігати в рядку стану в правому нижньому куті вікна програми. Після завершення генерації кадрів стає доступною панель управління анімацією, яка розташована в лівому верхньому куті графічного вікна. Кнопка Play цієї панелі запускає анімацію, кнопка Stop - зупиняє, а кнопки перегортають анімацію «покадрово».

Запустіть анімацію кнопкою Play. На наступному рисунку показані кадри, побудовані в моменти часу `t=1,2,3,4`.



Ви можете записати анімацію в відеофайл. Для цього в гілку **Results - Export** слід додати вузол Animation. Він має налаштування, схожі до «плеєра», а також поля для введення формату і імені файла.

Зауваження. Якщо в динамічній моделі потрібно намалювати лише плоску криву по її параметричному рівнянню, то для цього достатньо застосувати `Global plot` графічний вузол. Він може містити графіки «глобальних» виразів, тобто аналітичних виразів, що задаються безпосередньо в полях ПВ цього вузла. Вирази можуть залежати від якогось одного зовнішнього параметра (при параметричному аналізі моделі), або від часу (в динамічних моделях). Для

прикладу побудуємо графік контура квадратної області $[-1,1]\times[-1,1]$. Його параметричні рівняння можна записати у вигляді:

$$\begin{aligned} x_s(t) &= 1+\frac{1}{2}\left(-|t|+|t-2|+|t-4|-|t-6|\right), \\ y_s(t) &= 1+\frac{1}{2}\left(-|t-2|+|t-4|+|t-6|-|t-8|\right), \end{aligned} \quad (0\le t\le 8). \qquad (3)$$

В поточному файлі створіть вузли для двох аналітичних функцій, які будуть репрезентувати вирази (3).

Global Definitions : RMB > Functions – Analytic

Function name = `xs`

`Definition` –**Expression** =

`1-abs(t)/2+abs(t-2)/2+abs(t-4)/2-abs(t-6)/2`

`Definition` –**Arguments** = `t`

Global Definitions : RMB > Functions – Analytic

Function name = `ys`

`Definition` –**Expression** =

`1-abs(t-2)/2+abs(t-4)/2+abs(t-6)/2-abs(t-8)/2`

`Definition` –**Arguments** = `t`

Переобчисліть модель, щоб функції стали відомі в розв'язку `Study 1/Solution 1`.

Study 1 : LMB > Update Solution

Створіть групу одновимірних графіків з `Global` вузлом в ній.

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

1D Plot Group 2:

`Data Set` – **Data** = `Study 1/Solution 1`

Задавати джерело даних `Study 1/Solution 1` потрібно тому, що воно містить інформацію про «відкомпільовані» функції `xs(t)` і `ys(t)`.

RMB > Global

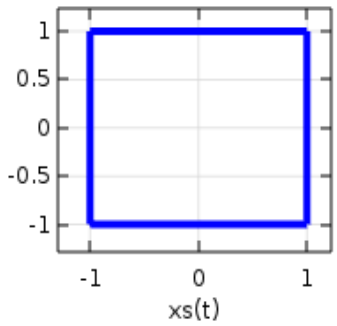
1D Plot Group 2 - Global 1:

`y-Axis Data` – **Expression** = `ys(t)` (*введення в таблиці виразів*)

`x-Axis Data` – **Parameter, Expression** = `Expression,xs(t)`

В розділі **`Coloring and Style`** задайте стиль, колір і товщину лінії.

LMB > Plot



Тут, задаючи вираз `xs(t)` в полі **`x-Axis Data`** – **Expression**, ми будуємо криву по параметричним рівнянням `xs(t),ys(t)`, а не графіки явно заданих функцій.

Вправа. Побудуйте в поточному фалі ще одну криву - квадрат, який аналогічно до попередньої кривої, описується рухомою точкою. Використайте вузли функцій `xs(t),ys(t)`, що задані рівняннями (3), і вже створені в поточному файлі. Далі повторіть всі інструкції побудови двовимірної графічної групи і анімації.

Розв'язок. Спочатку створіть функції `xs(t),ys(t)`, якщо їх ще немає в моделі, і виконайте повторно команду обчислення.

`Home` **->** Compute

Створіть двовимірну графічну групу.

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

2D Plot Group 2 :

`Data` – **Data set** = `Study 1/Solution 1`

`Data`– **Time (s)** = `7.1`

`Plot Settings` – **Plot data set edges** = `False`

В графічній групі **2D Plot Group 2** так само, як і в **2D Plot Group 1**, створіть і налаштуйте вузол `Point Trajectories`.

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

2D Plot Group 2 - Point Trajectories 1 :

`Data`– **Data set** = `From Parent;`

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

– **x-expression** = `xs(t)`

– **y-expression** = `ys(t)`

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

– **`Line style`**–**Tube radius expression** =`0.02`

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

– **Color** =`Blue`

Щоб зобразити рухому точку, налаштуйте секцію **Point style**.

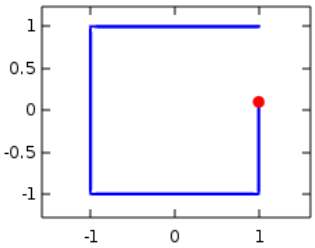
`Coloring and Style` – **Point style** –**Type** = `Point`

– **Point style** –**Point radius expression** = `0.07`

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

– **Point style** – **Color** = `Red`

LMB > Plot



Аналогічно до попереднього створіть анімацію (самостійно). ■

Приклад 3. Траєкторія точки на поверхні.

Побудуйте модель, в якій динамічно малюватиметься крива на сфері. Створіть шаблон порожньої моделі.

File -> New -> Blank Model

Створіть функції, які будуть представляти параметричні рівняння поверхні (сфери).

Global Definitions : RMB > Functions – Analytic

Function name = X

Definition –**Expression** = cos(u)*cos(v)

Definition –**Arguments** = u,v

Аналогічно створіть функції

Y(u,v)= sin(u)*cos(v)

Z(u,v)= sin(v)

Для стислості ми не наводимо вміст полів в панелях властивостей цих функцій. Потім додайте в модель тривимірний компонент.

Home->Add Component->3D

В компоненті побудуйте куб, що буде репрезентувати частину тривимірного простору, в якому будуть розташовані сфера і крива.

Geometry -> Primitives -> Block

Geometry - Block 1:

Size –**Width, Depth, Height** =2,2,2 (*три поля*)

Position – **Base**=Center

Position – **x, y, z** =0,0,0 (*три поля*)

LMB >Build All Objects

Генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Додайте в модель вузол дослідження.

Home -> Add Study

В панелі Add Study, що з'являється праворуч від графічного вікна, виберіть тип дослідження Time Dependent і закрийте панель. Далі задайте діапазон і інтервал зміни часу.

Study 1 – Step 1:Time Dependent:

Study Settings – **Times**= range(0,0.15,6.3)

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено джерело даних Sudy 1/Solution 1.

Побудуйте сферу, створивши спочатку джерело даних Parameterized Surface.

Results -> Data Set - More Data Sets-> Parameterized Surface

Data Sets – Parameterized Surface 1 :

Data– **Data set**= Study 1/Solution 1

Parameters – **First parameter** – **Name**=u

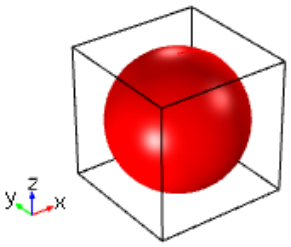
– **First parameter** – **Minimum, Maximum**=0,2*pi

– **Second parameter** – **Name**=v

– **Second parameter**–**Minimum, Maximum** =-pi/2,pi/2

Expressions – **x, y, z** =X(u,v),Y(u,v),Z(u,v) (*три поля*)

LMB >Plot



Створіть тривимірну графічну групу і в ній намалюйте поверхню сфери.

Home->**Add Plot Group**->3D Plot Group

Results – 3 D Plot Group 1:

Data– **Data set** = Study 1/Solution 1

Data– **Time (s)** = 6.15

3D Plot Group 1-> Add Plot - Surface

3D Plot Group 1 – Surface 1:

Data– **Data set** = Parameterized Surface 1

Data– **Time (s)** = 6.15

Expression – **Expression** = 1

Coloring and Style – **Coloring, Color table** =Color table,Wave

– **Color legend** = True

LMB >Plot

Додайте в графічну групу вузол Point Trajectories і налаштуйте його.

3D Plot Group 1-> **Add Plot - More Plots** -> Point Trajectories

3D Plot Group 1 – Point Trajectories 1:

Data– **Data set** = From Parent

Trajectory Data – **Plot data** = Coordinates

– **x-expression** = X(t,t)

– **y-expression** = Y(t,t)

– **z-expression** = Z(t,t)

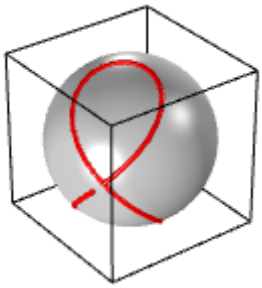
Coloring and Style –**Line style**–**Type** =Tube

– **Line style**– **Tube radius expression** =0.03

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

– **Color** =Red

Зверніть увагу, що крива задана рівняннями $u=t, v=t$. Їх підстановка в параметричне рівняння поверхні $X(u,v)$, $Y(u,v)$, $Z(u,v)$ створює просторове параметричне рівняння кривої, яка обов'язково буде розташована на поверхні сфери. Виконайте команду Plot і отримайте зображення кривої для моменту часу t=6.15.



Тепер побудуйте анімацію.

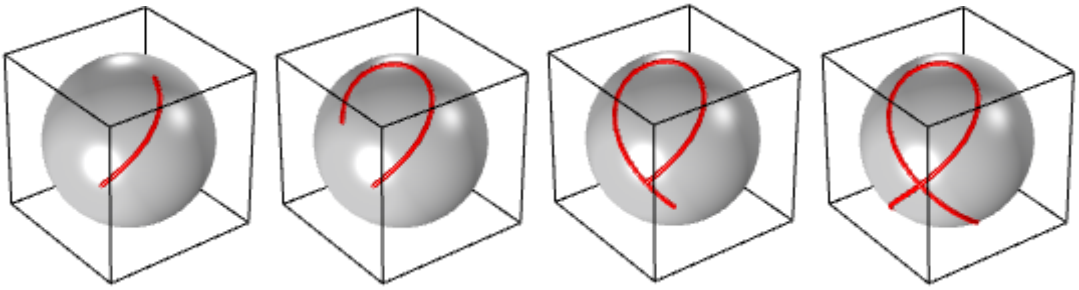
`Results` –> Animation –> Player

Export – Animation 1:

`Scene` – **Subject** = `3D Plot Group 1`

`Frames`– **Frame selection** = `All`

Згенеруйте кадри анімації, натиснувши кнопку Generate Frame. По завершенні запустіть анімацію кнопкою Play. На наступному рисунку зображено кілька кадрів в моменти часу `t=1.2,2.55,3.45,6.3`.



■

Схожими до траєкторій точок є графіки векторних ліній. Вони зображають криві, дотичні до яких в будь-який момент часу співпадають з вектором поля $\boldsymbol{V}=(V_x,V_y,V_z)$, і характеризуються рівностями $\frac{dx}{V_x}=\frac{dy}{V_y}=\frac{dz}{V_z}$. В гідромеханіці їх називають лініями течії векторного поля. Траєкторії частинок рідини з радіус-векторами $\boldsymbol{r}(t)$ визначаються рівняннями $\boldsymbol{r}_t'=\boldsymbol{V}(t,\boldsymbol{r})$. При усталеному русі, коли вектор поля $\boldsymbol{V}(\boldsymbol{r})$ не залежить від часу, лінії течії збігаються з траєкторіями частинок, тобто частинки рідини рухаються вздовж цих ліній. При неусталеному русі лінії течії і траєкторії частинок не співпадають.

В COMSOL Multiphysics для побудов ліній течії призначені `Streamline` графіки. Розглянемо їх.

Приклад 4. Створіть шаблон порожньої моделі `Blank Model`.

File -> New -> Blank Model

Додайте в модель двовимірний компонент.

`Home` -> Model -> Add Component – 2D

Геометрія компонента складатиметься з одного квадрата.

`Geometry` -> **`Primitives`** – Square

Geometry 1 – Square 1:

`Size` – **Side length** = `2`

`Position` – **Base, x, y** = `Center,0,0` (*три поля*)

На квадраті генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Додайте в модель `Time Dependent` вузол дослідження (параметр часу знадобиться, коли для порівняння з лініями течії ми будуватимемо траєкторії частинок).

Untitled.mph (root): RMB > Add Study = `Add Study` ->
Preset Studies - Time Dependent ->X

Задайте діапазон і інтервал зміни параметра часу `t`.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times** = `range(0,0.1,6.3)`

Виконайте обчислення.

Study 1 : LMB > Compute

Створіть двовимірну графічну групу з `Streamline` вузлом.

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

`Data` – **Data set** = `Study 1/Solution 1`

`Data`– **Time (s)** = `6.3`

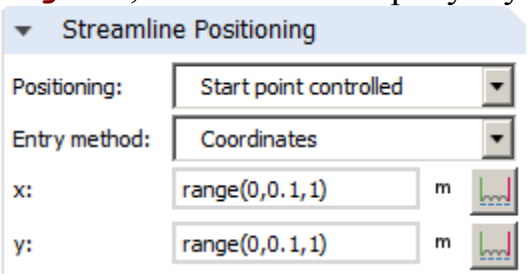
`2D Plot Group 1` -> Add Plot - Streamline

2D Plot Group 1 - Streamline 1 :

`Data` – **Data set** = `From parent`

`Expression` – **x component, y component** =`-y,x` (*два поля*)

Лінії течії в будь-якій своїй точці (*x*, *y*) матимуть дотичний вектор (-*y*, *x*). Для однозначного визначення ліній слід задати точки, через які вони проходитимуть. Для цього призначені поля **x** і **y** розділу **`Streamline Positioning`**. В них задаються абсциси і ординати точок. Заповніть розділ **`Streamline Positioning`** так, як показано на рисунку.



Налаштуйте тип, товщину і колір ліній.

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

–**Tube radius expression** =`0.03`

– **Radius scale factor** =`True,1`

– **Color** =`Blue`

Для наочності нанесіть на рисунок точки, які були використані для визначення ліній течії. Для цього в гілку **2D Plot Group 1** додайте вузол `Scatter Surface`.

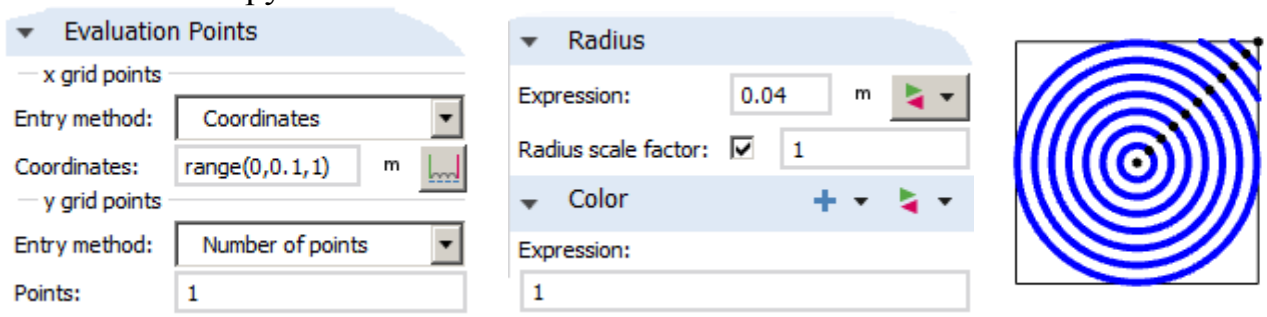
2D Plot Group 1 : RMB > More Plots - Scatter Surface

2D Plot Group 1 - Scatter Surface 1 :

`Expression` – **x component**= `x`

`Expression` – **y component**= `x` (знову *x – зверніть увагу*)

Розділи **Evaluation Points** і **Radius** заповніть так, як показано нижче. В ПВ ці області розташовані одна під одною, але для економії місця ми розмістили їх поруч.



Coloring and Style – **Coloring, Color** = `Uniform, Black` (*два поля*)

LMB > Plot (*попередній рисунок праворуч*)

Налаштування в групі **y grid points** не мають значення, оскільки в полях розділу **Expression** змінна `y` не використовується (згадайте особливості `Scatter Surface` вузлів). Зверніть також увагу на те, що лінії течії, що виходять за межі джерела даних, не повертаються назад в область, навіть якщо їх продовження повинно там з'явитися.

На останньому графіку використовується стаціонарне векторне поле $\boldsymbol{V} = (-y, x)$. Тому траєкторії точок повинні співпадати з лініями течії. Це означає, що коли побудувати `Particle Tracing` графік з тими ж початковими точками, що і у вузлі **Streamline 1**, то повинні побудуватися ті самі криві.

2D Plot Group 1 : RMB > More Plots - Particle Tracing

2D Plot Group 1 - Particle Tracing 1 :

Expression – **x component, y component** =`-y,x` (*два поля*)

Задайте початкове положення рухомих точок

Particle Positioning – **Positioning** = `Start point controlled`

– **x** = `range(0,0.1,1)`

– **y** = `range(0,0.1,1)`

Налаштуйте стиль кривих.

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

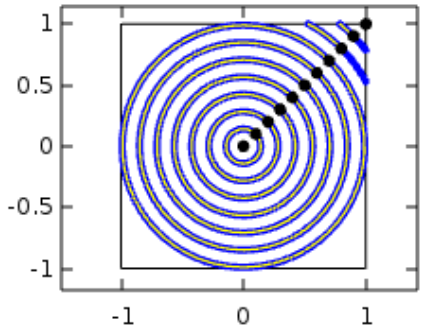
– **Line style** – **Tube radius expression** =`0.01`

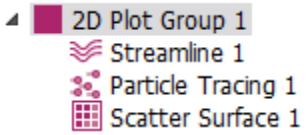
– **Line style** – **Radius scale factor** =`True,1`

– **Color** = `Yellow`

Щоб точки малювалися поверх кривих, перемістіть вузол **Scatter Surface 1** вниз графічної гілки **2D Plot Group 1**. Для цього захопіть вузол лівою кнопкою миші (натисніть кнопку і не відпускайте) і перетягніть вниз групи. В процесі пересування позиція вузла в дереві позначається червоним відрізком.

LMB > Plot





Як бачимо, траєкторії точок і лінії течії співпали.

Подивимось, чи будуть співпадати лінії течії і траєкторії точок у випадку, коли векторне поле залежить від часу.

2D Plot Group 1 : RMB > Duplicate

2D Plot Group 2- Streamline 1 :

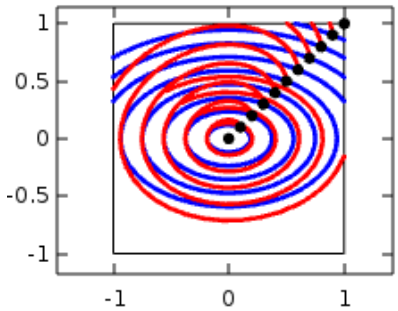
`Expression` – x component, y component `=-y*sqrt(t),x` (*два поля*)

2D Plot Group 2 - Particle Tracing 1 :

`Expression` – x component, y component `=-y*sqrt(t),x` (*два поля*)

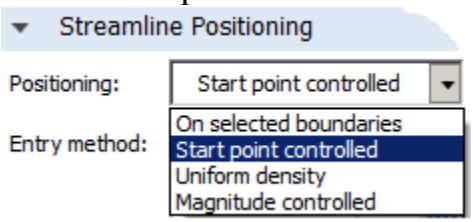
`Coloring and Style` – Color = `Red`

LMB > Plot



Стартові точки кривих залишилися однаковими, векторне поле однакове, але сім'ї кривих відрізняються.

Зауваження. В розділі **`Streamline Positioning`** визначається спосіб розрізнення ліній векторного поля. Це можуть бути координати точок, через які проходитимуть лінії течії (`Start point controlled`). Вибір в списку **Positioning** рядка `On selected boundaries` означає, що якась границя буде обрана для розташування стартових точок, а їх кількість на цій границі потрібно буде задати. Вибір `Uniform density` означає, що користувач задаватиме початкову відстань між кривими.



■

Побудуємо лінію течії в тривимірному просторі.

Приклад 5. Створіть шаблон порожньої моделі `Blank Model`.

File -> New -> Blank Model

Додайте в модель тривимірний компонент.

Home –> Model –> Add Component – 3D

Геометрія компонента складатиметься з одного куба.

Geometry –> **Primitives** – Block

Geometry 1 – Block 1:

Size – Width, Depth, Height = 2,2,2 (*три поля*)

Position – Base = Center

– x, y, z = 0,0,0 (*три поля*)

Генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

В гілку даних **Data Sets** дерева моделі додайте джерело даних, яке буде репрезентувати цю сітку.

Results - Data Sets : RMB > Mesh

Data Sets - Mesh 1 : Mesh – Mesh = Mesh 1

Створіть тривимірну графічну групу з підвузлом типу Streamline.

Results –> Plot Group - 3D Plot Group

3D Plot Group 1 : Data – Data set = Mesh 1

3D Plot Group 1 –> Add Plot - Streamline

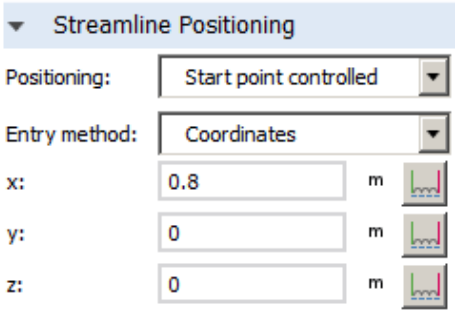
3D Plot Group 1 - Streamline 1 :

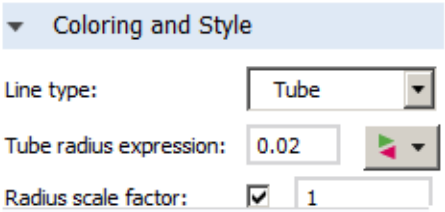
Expression – x component =-y

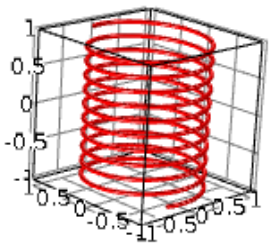
– y component = x

– z component = 0.03

Заповніть розділи **Streamline Positioning** і **Coloring and Style** так, як показано на рисунку. Виконайте команду Plot і отримайте графік кривої, показаний праворуч. Крива одна, оскільки було задано одну стартову точку.







Таким чином, тут зображено лінію течії векторного поля $(-y, x, 0.03)$, яка проходить через точку (0.8,0,0).

2.1.5. Допоміжні графіки.

Окрім графіків, що зображають розв'язки (або якісь вирази), існує ще кілька типів графіків, які допомагають користувачеві зрозуміти постановку задач і надати виразну пояснювальну графічну інформацію. Наприклад, графічні вузли Coordinate System Volume, Coordinate System Surface і Coordinate System Line креслять репери координатних векторів в точках геометрії, до якої вони застосовуються.

Розглянемо роботу з графічними вузлами Coordinate System Line, Max/Min Line і змінними локальної системи координат Boundary System 1 (sys1).

Приклад 1. Спочатку створимо шаблон моделі з 2D компонентом, який буде використано далі для демонстрування роботи двовимірних службових графіків.

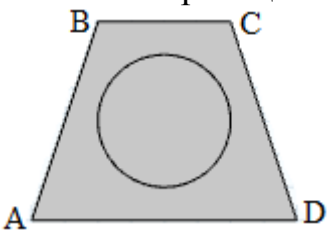
Створіть шаблон порожньої моделі Blank Model.

File –> New –> Blank Model

Додайте в модель двовимірний компонент.

Home –> Model –> Add Component – 2D

Геометрія компонента складатиметься з трапеції і кола.

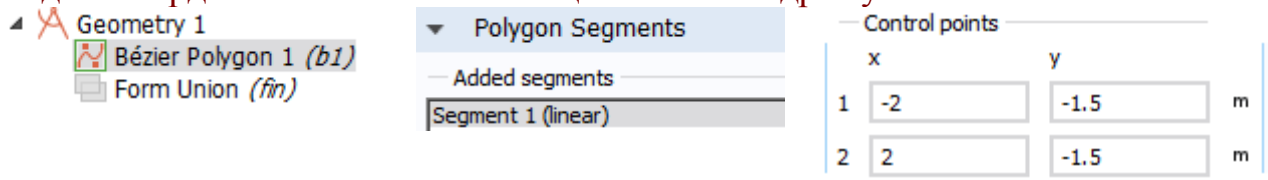


Трапецію можна побудувати за допомогою графічного примітива Bézier Polygon.

Geometry -> **Pimitives** – Bézier Polygon

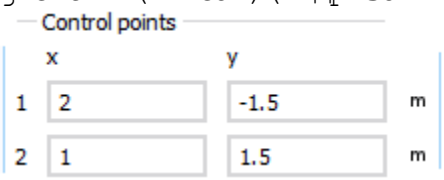
Bézier Polygon 1 : LMB > Add Linear

Введіть координати початкової і кінцевої точок відрізку AD.

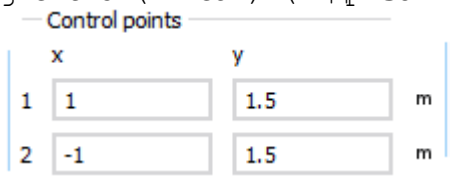


Ще 2 рази клацніть по кнопці Add Linear і кожного разу вводьте координати початкової і кінцевої точок відрізків DC і CB.

Segment 2 (linear) (відрізок DC)

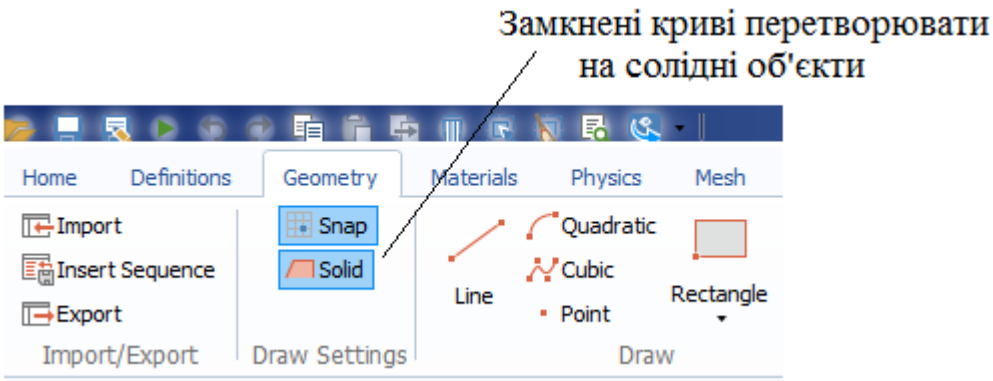


Segment 3 (linear) (відрізок CB)



Клацніть по кнопкам Add Linear і Close Curve, замикаючи ламану відрізком BA (її поки що не видно). Завершіть будування трапеції клацанням по кнопці Build Selected.

Зверніть увагу на те, що ми будували ламану, а було отримано двовимірну область. Це тому, що на панелі Geometry в розділі **Draw Settings** увімкнена опція **Solid**.



Додайте в геометрію коло.

`Geometry` **-> `Primitives`** – Circle

Component 1 - Geometry 1 - Circle 1 : `Size and Shape` – **Radius:** = `1`

LMB > Build All Objects

Генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Додайте в модель `Stationary` тип дослідження.

`Home` **-> `Study`** – Add Study = `Add Study ->`

Preset Studies - Stationary **->x**

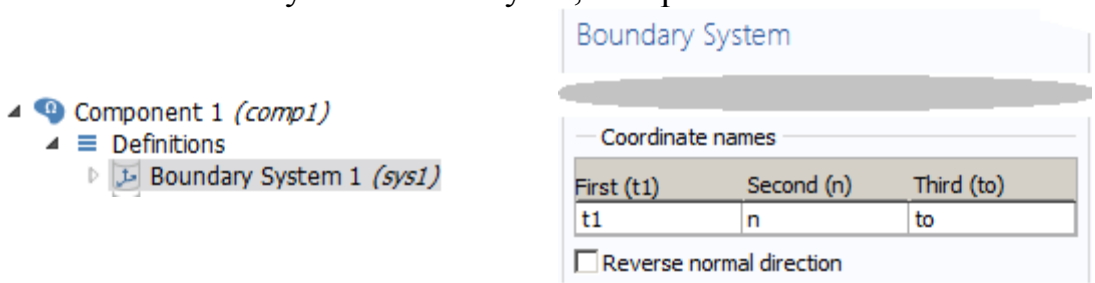
В результаті в ДМ з'являється вузол **Study 1** і кілька його підвузлів.

Виконайте обчислення.

Study 1 : LMB > Compute

Ніяких реальних обчислень виконуватися не буде, але в гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**.

В COMSOL Multiphysics всі об'єкти репрезентуються вузлами дерева моделей. Це стосується і локальних систем координат. Вони створюються в гілці **Component – Definitions** ї представляють локальні системи базисних векторів на границях розрахункових областей. Зазвичай ці вектори використовуються для задання нестандартних граничних умов, які, наприклад, можуть залежати від напрямку границі (це може бути зовнішня сила прикладена вздовж дотичної, нормалі, тощо). В 2D і 3D моделях один вузол **Boundary System 1 (sys1)** створюється завжди. На наступному рисунку праворуч показано частину ПВ такого вузла, створеного в 2D компоненті.



В двовимірних компонентах базисні вектори позначаються за замовчуванням як `t1`, `n`, `to`. (дотичний і нормальний вектори, а також їх векторний добуток **n**×**t1**). Ці імена стоять в таблиці **Coordinate names** ПВ вузла **Boundary System 1 (sys1)**. В виразах, які використовуються на границях, можна застосовувати координати цих векторів `t1x`,`t1y`,`nx`,`ny`.

Розглянемо застосування графічного вузла Coordinate System Line. За його допомогою можна зобразити базисні вектори локальної системи координат **Boundary System 1 (sys1)** вздовж внутрішньої лінії геометрії або її межі.

Створіть в поточному файлі два джерела даних типу Edge 2D: одне буде репрезентувати коло, інше – трапецію (зовнішню межу розрахункової області).

Results -> Data Set - More Data Sets-> Edge 2D

Edge 2D 1: Selection – **Selection,<>** =Manual,5,6,7,8 (*коло*)

Results -> Data Set - More Data Sets-> Edge 2D

Edge 2D 2 : Selection – **Selection,<>** =Manual,1,2,3,4 (*трапеція*)

Results -> Plot Group - 2D Plot Group

2D Plot Group 1 -> Add Plot - More Plots - Coordinate System Line

2D Plot Group 1 - Coordinate System Line 1:

Data – **Data set** = Edge 2D 1 (*локальна координатна система на колі*)

Coordinate System – **Coordinate System**=Boundary System 1(sys)

Опція **Coordinate System**=Boundary System 1(sys) визначає координатну систему (їх може бути декілька), вектори якої будуть зображатися.

Coloring and Style – **Scale factor** = True,1

Coloring and Style – **Number of arrows** = 20 (*кількість реперів*)

LMB >Plot (*наступний рисунок ліворуч*)

2D Plot Group 1 : RMB > Duplicate

2D Plot Group 2 - Coordinate System Line 1:

Data – **Data set** = Edge 2D 2 (*локальна координатна система на трапеції*)

Coloring and Style – **Number of arrows** = 40

LMB >Plot (*наступний рисунок в центрі*)

Якщо в якості ДД обрати розв'язок Study 1/Solution 1, то будуть побудовані вектори на обох кривих: колі і квадраті.

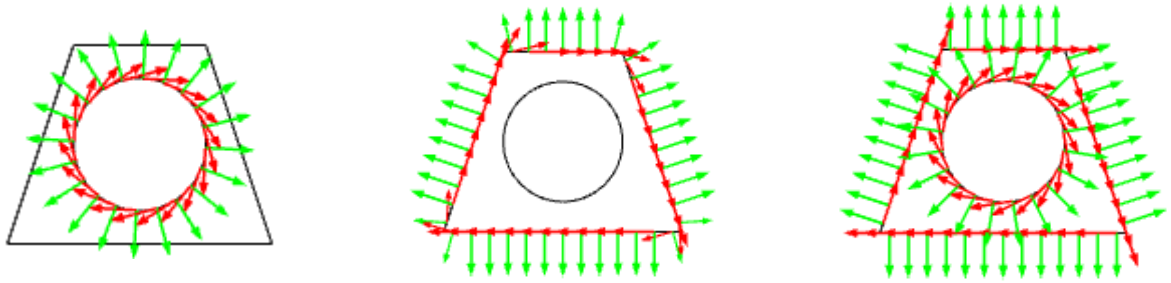
2D Plot Group 2 : RMB > Duplicate

2D Plot Group 3 - Coordinate System Line 1:

Data – **Data set** = Study 1/Solution 1 (*локальна координатна система на всіх одновимірних елементах геометрії*)

Coloring and Style – **Number of arrows** = 60

LMB >Plot (*наступний рисунок праворуч*)



Графік Max/Min Line будує на лінії точки мінімуму і максимуму виразу, який вказано в полі **Expression**.

2D Plot Group 1 - Coordinate System Line 1: RMB > Disable

2D Plot Group 1 –> Add Plot - More Plots - Max/Min Line

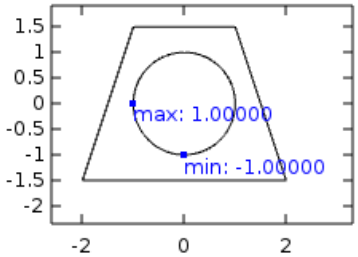
2D Plot Group 1 - Max/Min Line 1:

Data – **Data set** = Edge 2D 1 (*екстремальні точки на колії*)

Expression – **Expression** = x^2-y^2

Coloring and Style – **Color** = Blue

LMB >Plot (*наступний рисунок праворуч*)



Після побудови точок мінімуму і максимуму в ДМ в гілці **Tables** з'являється таблиця, яка містить координати екстремальних точок і значення екстремумів.

x	y	x^2-y^2 (m^2)
0.0000	-1.0000	-1.0000
-1.0000	0.0000	1.0000

2D Plot Group 2 - Coordinate System Line 1: RMB > Disable

2D Plot Group 2 –> Add Plot - More Plots - Max/Min Line

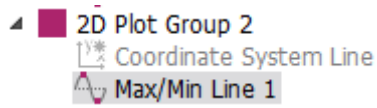
2D Plot Group 2 - Max/Min Line 1:

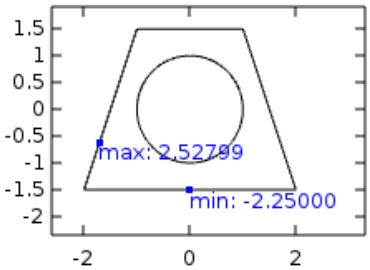
Data – **Data set** = Edge 2D 2 (*екстремальні точки на трапеції*)

Expression – **Expression** = x^2-y^2

Coloring and Style – **Color** = Blue

LMB >Plot (*наступний рисунок в середині*)





x	y	x^2-y^2 (m^2)
1.7208E-15	-1.5000	-2.2500
-1.7077	-0.62306	2.5280

Графік типу Point Trajectories при фіксованих значеннях в полях **x-expression** і **y-expression** відображає точку з заданими координатами.

2D Plot Group 1 –> Add Plot - More Plots - Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

Data– **Data set** = Study 1/Solution 1

Trajectory Data – **Plot data** = Coordinates

– **x-expression** = sqrt(2)/2

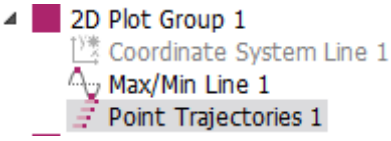
– **y-expression** = sqrt(2)/2

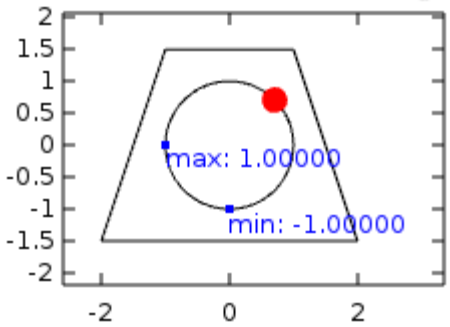
Coloring and Style – **Point style -Type** = Point

– **Point style** – **Point radius expression** = 0.2

– **Point style** – **Radius scale factor**= True,1

– **Point style** – **Color** = Red





Поглянемо на застосовування базисних векторів локальної системи координат **Boundary System 1 (sys1)**. Як ми казали, в вирази, що використовуються на лініях, можна підставляти змінні, які репрезентують координати цих векторів, `t1x,t1y,nx,ny` (якщо ви їх не перепозначили).

Створимо нову графічну групу і в ній лінійний `Line` графік, в полі виразів якого використаємо змінні `t1x,t1y,nx,ny`.

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

2D Plot Group 4: `Data` – Data set = `Edge 2D 1` (*вираз на колі*)

`2D Plot Group 4` –> Add Plot - Line

2D Plot Group 4 - Line 1:

`Expression` – Expression = `(t1x+t1y+nx)/(ny^2+1)`

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

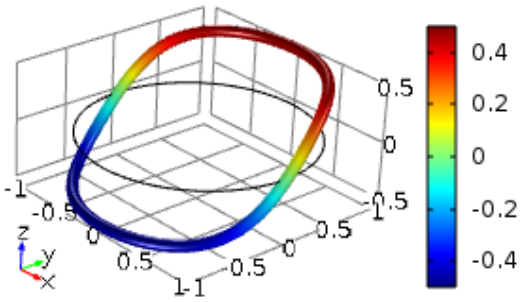
– **`Line style`–Tube radius expression** =`0.05`

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

2D Plot Group 4 - Line 1 : RMB > Height Expression

Line 1 – Height Expression 1 : `Axis` – Scale factor = `True, 1` (*два поля*)

LMB > Plot



Інколи корисно використати змінні у вузлах `Deformation`. Для приклада намалюємо внутрішнє коло нашої геометрії, деформоване в горизонтальному напрямку пропорційно величині `nx`.

2D Plot Group 4 : RMB >Duplicate

2D Plot Group 4 - Line 1 - Height Expression : RMB > Delete

2D Plot Group 5 - Line 1: (*недеформоване коло*)

`Expression` – Expression = `1`

`Coloring and Style` – Coloring, Color =`Uniform, Blue` (*два поля*)

2D Plot Group 5 - Line 1 : RMB >Duplicate

2D Plot Group 5 - Line 2: (*деформоване коло*)

`Coloring and Style` –Coloring, Color =`Uniform,Red` (*два поля*)

2D Plot Group 5 - Line 2 : RMB > Deformation

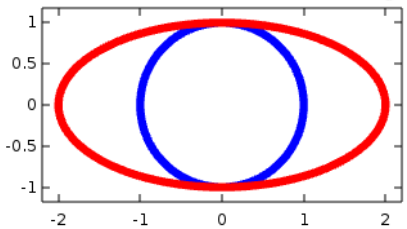
2D Plot Group 5 - Line 2 – Deformation 1:

Expression – **x component** = nx

– **y component** = 0

Scale – **Scale factor** = True, 1 (*два поля*)

LMB > Plot



Змінні t1x,t1y,nx,ny можна використовувати в виразах будь-яких вузлів, які «працюють» на лініях (у вузлах лінійних графіків, граничних умов, тощо).

■

Розглянемо роботу з графічними вузлами Coordinate System Surface, Max/Min Surface, Max/Min Volume, Principal Stress Surface, Principal Stress Volume. Для цього нам знадобиться тривимірна модель.

Приклад 2. Створіть шаблон порожньої моделі.

File -> New -> Blank Model

Додайте в модель тривимірний компонент.

Home -> Model -> Add Component – 3D

Геометрія компонента складатиметься зі зрізаного зверху і збоку конуса.

Geometry -> **Primitives** – Cone

Geometry 1 – Cone 1:

Size and Shape – **Bottom radius, Height** = 4,3 (*два поля*)

– **Specify top size using** = Radius

– **Top radius** = 1

Position – **x, y, z** = 0,0,0 (*три поля*)

Geometry -> **Primitives** – Block

Geometry 1 – Block 1:

Size – **Width, Depth, Height** = 10,10,5 (*три поля*)

Position – **Base** = Corner

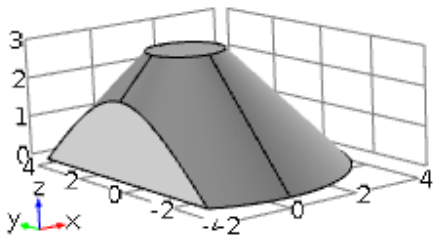
Position – **x, y, z** = -2,-5,0 (*три поля*)

Geometry -> **Booleans and Partitions** – Intersection

Geometry 1 - Intersection 1:

Intersection – Input **objects** = cone1, blk1 (*зрізаний конус та блок*)

LMB > Build All Objects



Генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Додайте в модель `Stationary` тип дослідження.

`Home` **-> `Study`** – Add Study = `Add Study ->`

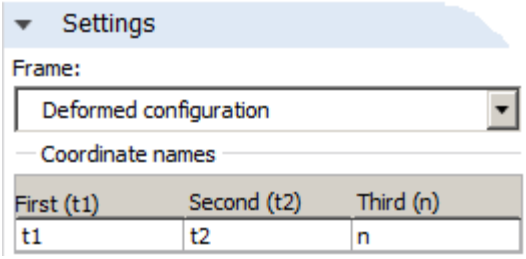
Preset Studies - Stationary **-> x**

В ДМ з'являється вузол **Study 1** і кілька його підвузлів. Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**.

Нагадаємо, що у вузлі **Component – Definitions** можна створювати вузли типу `Boundary System`. Вони представляють локальні системи базисних векторів на межі розрахункової області. Один вузол **Boundary System 1 (sys1)** створюється завжди. В тривимірних компонентах базисні вектори за замовчуванням позначаються `t1,t2,n` (два дотичних і нормальний вектори). Ці імена позначені в таблиці **Coordinate names** ПВ вузла **Boundary System 1 (sys1)**.



В виразах, що використовуються на границях, можна застосовувати координати цих векторів, які позначаються змінними `t1x,t1y,t1z,t2x,t2y,t2z,nx, ny,nz`.

Розглянемо графічний вузол `Coordinate System Surface`. За його допомогою можна зобразити базисні вектори локальної системи координат **Boundary System 1 (sys1)** на межі тіла.

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

`3D Plot Group 1` -> Add Plot - More Plots - Coordinate System Surface

3D Plot Group 1 - Coordinate System Surface 1:

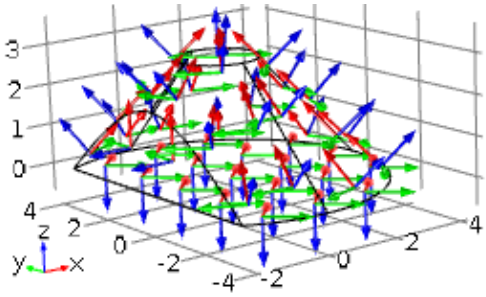
`Coordinate System` – **Coordinate System**=`Boundary System 1(sys)`

Опція **Coordinate System**=`Boundary System 1(sys)` визначає координатну систему, вектори якої будуть зображатися.

`Coloring and Style` – **Number of arrows** = `40` (*кількість реперів*)

`Coloring and Style` – **Scale factor** = `True,1.6`

LMB >Plot



Тут координатні репери (трійки одиничних базисних векторів) побудовані на поверхні тіла.

Графік `Max/Min Surface` будує на поверхні точки мінімуму і максимуму виразу, який вказано в полі **Expression**.

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

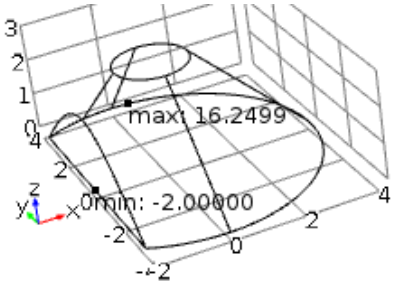
`3D Plot Group 2` –> Add Plot - More Plots - Max/Min Surface

3D Plot Group 2 - Max/Min Surface 1:

`Expression` – **Expression** = `(x+y^2)/(1+z^2)`

LMB >Plot

Одночасно з графіком створюється таблиця, яка містить координати екстремальних точок і значення виразу в них.



x	y	z	(x+y^2)/(1+z^2)
-2.0000	3.6606E-6	0.0000	-2.0000
0.49040	3.9698	0.0000	16.250

Графік `Max/Min Volume` будує точки мінімуму і максимуму в об'ємі тіла.

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

`3D Plot Group 3` –> Add Plot – Surface

3D Plot Group 3 - Surface 1:

`Expression` – **Expression** = `x^2+y^2+(z-1)^2`

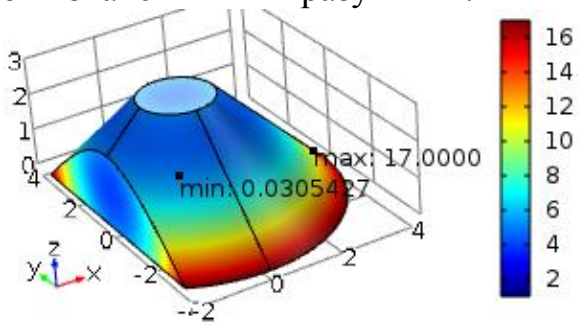
`3D Plot Group 3` –> Add Plot - More Plots - Max/Min Volume

3D Plot Group 3 - Max/Min Volume 1:

`Expression` – **Expression** = `x^2+y^2+(z-1)^2`

LMB >Plot

Одночасно з графіком створюється таблиця з координатами екстремальних точок і значеннями виразу в них.



x	y	z	x^2+y^2+(z-1)^2
-0.088554	-0.0082495	0.84956	0.030543
3.9845	0.35227	0.0000	17.000

Графік `Principal Stress Surface` будує трійки векторів на поверхні тіла. Направляючи вектори репера задаються в таблиці **Principal directions,** а скалярні множники до їх довжин в полях **Value** розділу **`Principal Components`**. У напівпрозорому зображенні видно, що вектора двонаправлені, і зазвичай видно ту частину, яка йде назовні поверхні.

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

`3D Plot Group 4` –> Add Plot – Surface

3D Plot Group 4 - Surface 1:

`Expression` – **Expression** = `x^2+y^2+(z-1)^2`

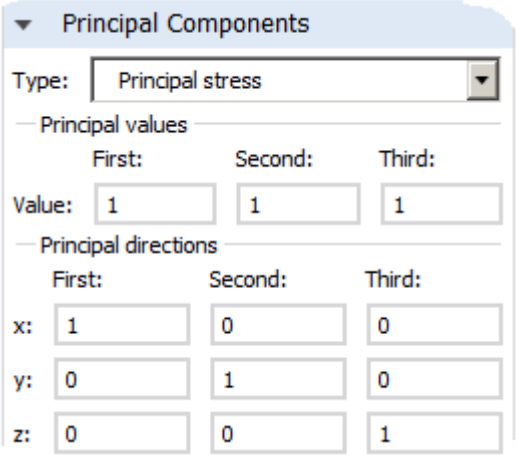
`3D Plot Group 4` –> Add Plot - More Plots - Principal Stress Surface

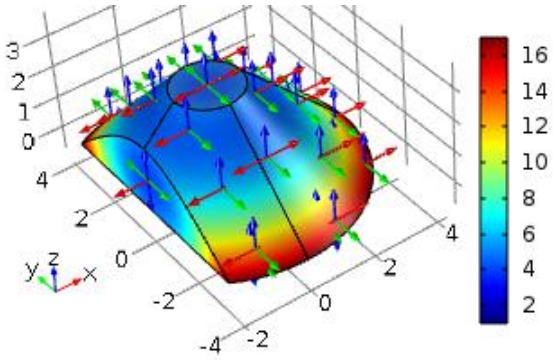
3D Plot Group 4 - Principal Stress Surface 1:

Coloring and Style – **Number of arrows** = 30 (*кількість реперів*)

Coloring and Style – **Scale factor** = True,1.6

Інші поля вузла **Principal Stress Surface 1** заповніть так, як зображено на рисунку.





LMB >Plot (*попередній рисунок праворуч*)

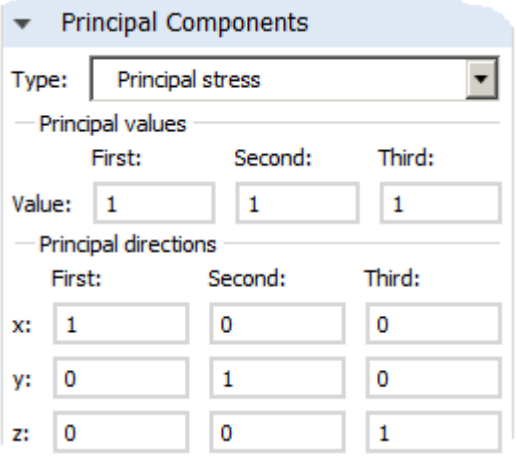
Графік типу Principal Stress Volume зображає трійки векторів в об'ємі тривимірного тіла. Направляючи вектори реперів задаються в таблиці **Principal directions**, а скалярні множники до їх довжин в полях **Value** розділу **Principal Components**.

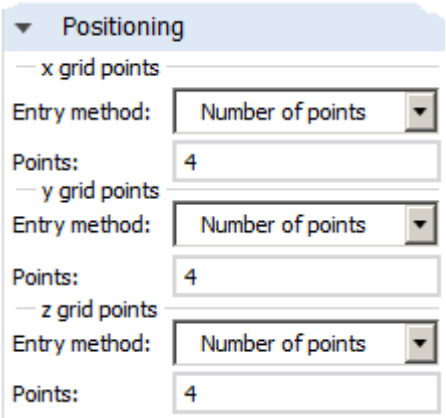
Results –> Plot Group - 3D Plot Group

3D Plot Group 5 –> Add Plot - More Plots - Principal Stress Volume

3D Plot Group 5 - Principal Stress Volume 1:

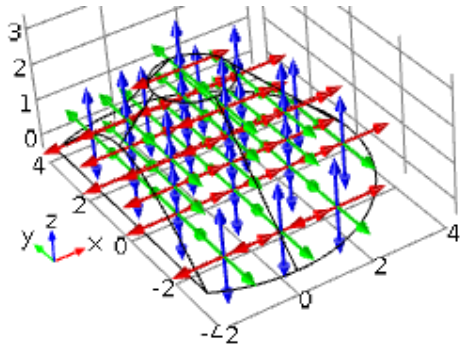
Направляючи вектори задайте в розділі **Principal Components** так, як показано на наступному рисунку ліворуч. Кількість реперів визначається в розділі **Positioning**, який зображено праворуч.





Coloring and Style – **Scale factor** = True,1.6

LMB >Plot



Раніше ми казали, що в виразах застосовуваних на границях, можна використовувати змінні, які репрезентують базисні вектори локальної системи координат. Простим прикладом застосування цієї можливості є фарбування і деформація поверхні відповідно виразів, які містять такі змінні.

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

`3D Plot Group 6` –> Add Plot – Surface

3D Plot Group 6 - Surface 1:

`Expression` – **Expression** = `asin(t2x)+acos(ny)-t1z`

2D Plot Group 6 - Surface 1 : RMB > Deformation

2D Plot Group 6 - Surface 1 – Deformation 1:

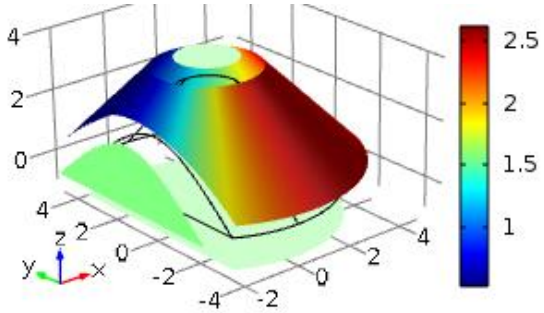
`Expression` – **x component** = `nx`

– **y component** = `ny`

– **z component** = `nz`

`Scale` – **Scale factor** = `True, 1` (*два поля*)

LMB > Plot



Зауваження. Окрім змінних, які репрезентують на границях вектори локальної системи координат, існують змінні, які відіграють роль криволінійних координат на поверхнях або кривих.

■

Зауваження 1. Більшість розглянутих в попередніх параграфах графічних вузлів мають декілька варіантів застосування, а ми, переважно, показували лише один. Сподіваємось, що користувач, маючи уявлення про призначення того чи іншого графіка, зможе самостійно розібратися в інших способах їх використання і з варіантами налаштування.

Зауваження 2. При побудові графіків, що спираються на різні джерела даних, ми застосовували математичні вирази або глобальні функції. Однак при розв’язанні прикладних задач ви, головним чином, будете використовувати вирази, що містять розв’язки ДРЧП. Щоб не ускладнювати ознайомлення з

графічними можливостями системи в цій главі ми не зверталися до розв’язків задач.

2.2. Обчислення інтегралів, довжин, площ та об’ємів.

При побудові графіків використовуються два основних математичних поняття: вираз, що зображується, і область, в якій виконується побудова. Область, як правило, визначається в джерелі даних. Математичний вираз ми задаємо явно або вибираємо з джерела даних, якщо він створюється в результаті розв’язання задачі (наприклад, ім'я розв’язку `u`). Але ці два поняття використовуються також при обчисленні інтегралів, середніх і екстремальних значень різних виразів, оскільки цим операціями, крім досліджуваних функцій, потрібно вказувати зони, на яких вони виконуються. Тут ми пояснимо, як використовувати джерела даних і `Derived Values` вузли для обчислення інтегралів, довжин кривих, площ поверхонь і об'ємів тіл.

2.2.1. Обчислення інтегралів, середніх і екстремальних значень.

Тут будуть розглянуто приклади використання джерел даних для обчислення інтегралів, середніх і екстремальних значень.

Приклад 1. Створіть шаблон порожньої моделі `Blank Model`.

File -> New -> Blank Model

Створіть функцію-заглушку.

Global Definitions : RMB > Functions – Analytic

Function name = `f`

Definition –**Expression, Arguments** = `1,x` (*два поля*)

Створіть одновимірне джерело даних `Grid 1D`.

`Results` -> Data Set - More Data Sets -> Grid 1D

Data Sets - Grid 1D 1 :

Data – **Source, Function** = `Function, All` (*два поля*)

Paramemter Bounds – **Name**= `x`

Paramemter Bounds – **Minimum, Maximum** =`-1,2` (*два поля*)

Обчислення інтегралів в COMSOL Multiphysics можна виконувати по-різному. Покажемо це на прикладі наступного інтеграла

$$\int_{-1}^{2} x\,d\,x = \left.\frac{x^2}{2}\right|_{-1}^{2} = 2 - \frac{1}{2} = \frac{3}{2}.$$

Перший спосіб: *використання вузла* `Derived Values - Line Integration`.

`Results` -> **`Derived Values`**-More Derived Values->Line Integration

Derived Values – Line Integration 1:

Label = `Інтеграл`

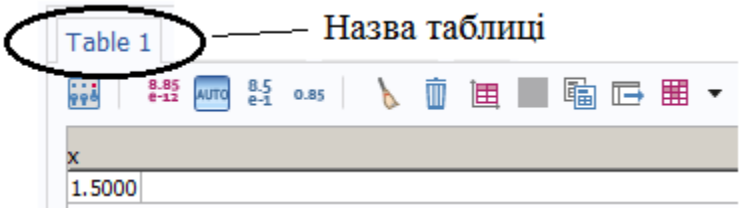
Назва вузла зміниться на «Інтеграл».

Data– **Data set** = `Grid 1D 1`

Expression – **Expression** = `x`

LMB > Evaluate

У правій нижній частині вікна програми буде створена таблиця з одним стовпцем і одним рядком. В клітині цієї таблиці з'явиться результат: `1.5000`.



Тут інтегрування виконувалося по ДД `Grid 1D 1`, тобто по відрізку [`-1,2`].

Другий спосіб: *використання джерела даних типу* `Integral`.

`Results -> Data Set - More Data Sets -> Integral`

Data Sets – Integral 1:

`Data` – **Data set**=`Grid 1D 1`

`Results -> Derived Values -> Point Evaluation`

Derived Values – Point Evaluation 1:

`Data` – **Data set** =`Integral 1`

`Expression` – **Expression** = `x`

LMB > Evaluate >Table 1 – Інтеграл

В новій комірці попередньої таблиці з'явиться той же результат `1.5000`.

x	x, Integral 1
1.5000	1.5000

Тут процес обчислення розділений на два кроки: в ДД **Integral 1** обчислюється інтеграл, а за допомогою вузла **Point Evaluation 1** він друкується в таблицю.

Третій спосіб подібний до другого і складається зі створення вузла `Global Evaluation`, який використовує те ж саме ДД **Integral 1**.

`Results -> Derived Values -> Global Evaluation`

Derived Values – Global Evaluation 1:

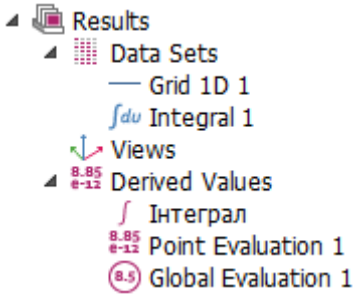
`Data` – **Data set** =`Integral 1`

`Expression` – **Expression** = `x`

LMB > Evaluate >Table 1 – Інтеграл

x	x, Integral 1	x
1.5000	1.5000	1.5000

Частина дерева моделі, яка містить створені вузли, зображена на наступному рисунку.



Обчислимо повторний інтеграл $\int_{-1}^{2}\left(\int_{-1}^{2} x\,dx\right)dx$.

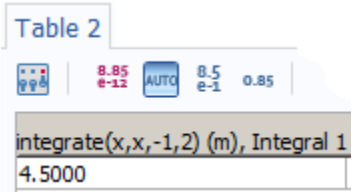
`Results -> Derived Values -> Point Evaluation`

Derived Values – Point Evaluation 2:

`Data` – **Data set** = `Integral 1`

`Expression` – **Expression** = `integrate(x,x,-1,2)`

LMB > Evaluate – New Table



Оператор `integrate(x,x,-1,2)` обчислює інтеграл $\int_{-1}^{2} x\,d\,x = 1.5$, записаний в полі **Expression** вузла **Point Evaluation 2**, і використовується в якості підінтегрального виразу джерела даних **Integral 1**. Отже тут ДД **Integral 1** ми повинні сприймати, як шаблон визначеного інтеграла $\int_{-1}^{2} \ldots d\,x$.

Обчислимо повторний інтеграл $\int_{-1}^{2}\left(\int_{-1}^{x} v^2\,d\,v\right)d\,x$.

`Results -> Derived Values -> Global Evaluation`

Derived Values – Global Evaluation 2:

`Data` – **Data set** = `Integral 1`

`Expression` – **Expression** = `integrate(v^2,v,-1,x)`

LMB > Evaluate >Table 2

integrate(x,x,-1,2) (m), Integral 1	integrate(v^2,v,-1,x)
4.5000	2.2500

Оператор `integrate(v^2,v,-1,x)` являє підінтегральний вираз $\int_{-1}^{x} v^2\,d\,v = \frac{x^3+1}{3}$ для зовнішнього інтеграла. Тут знову ДД **Integral 1** ми повинні сприймати, як шаблон $\int_{-1}^{2} \ldots d\,x$, при чому ім'я x змінної інтегрування суттєве.

Інтеграл по області також можна знаходити декількома способами. Природно, що для цього потрібно двовимірне джерело даних.

`Results -> Data Set - More Data Sets -> Grid 2D`

Data Sets – Grid 2D 2:

`Data` – **Source, Function** = `Function, All` (*два поля*)

В розділі **`Parameter Bounds`** в полях **Minimum** і **Maximum** обох параметрів залиште значення 0 і 1.

Використовуючи це джерело даних, обчислимо інтеграл $\iint\limits_{[0,1]\times[0,1]} (x+y)\,d\,x\,d\,y$ по квадрату $[0,1]\times[0,1]$.

Перший спосіб полягає в використанні вузла типу `Surface Integration`.

`Results->`**`Derived Values`**`-More Derived Values->Surface Integration`

Derived Values – Surface Integration 2:

Data – **Data set** = Grid 2D 2

Expression – **Expression** = x+y

LMB > Evaluate > Table 1 - Інтеграл

В новій комірці таблиці **Table 1 - Інтеграл** з'явиться результат 1.0000.

x	x, Integral 1	x	x+y
1.5000	1.5000	1.5000	1.0000

Другий спосіб полягає в використанні ДД Integral і подальшому друкуванні результату з вузла типу Global Evaluation (або Point Evaluation).

Results -> **Data Set** - More Data Sets -> Integral

Data Sets – Integral 2:

Data – **Data set**=Grid 2D 2

Results-> Derived Values-> Global Evaluation

Derived Values – Global Evaluation 3:

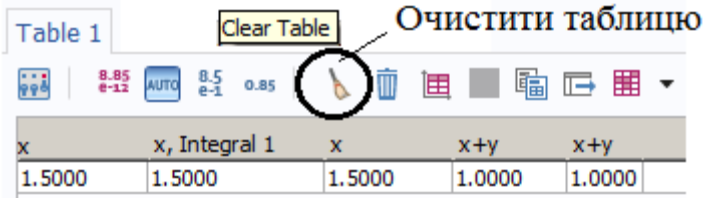
Data – **Data set**=Integral 2

Expression – **Expression** = x+y

LMB > Evaluate > Table 1 - Інтеграл

Тут ДД **Integral 2** повинно сприйматися, як шаблон $\iint\limits_{\text{Grid 2 D}} \ldots d\,x\,d\,y$, тобто як подвійний інтеграл по квадрату $[0,1]\times[0,1]$, і імена змінних інтегрування x, y суттєві.

В новій комірці **Table 1 – Інтеграл** з'явиться результат 1.0000. Якщо комірка таблиці, в якій виводиться результат, розташована далеко праворуч, то можна очистити таблицю. Для цього над таблицею є кнопка Clear Table.



Те, що результат вірний, можна переконатися за допомогою простих розрахунків.

$$\iint\limits_{[0,1]\times[0,1]} (x+y)\,d\,x\,d\,y = \int_0^1 \left(\int_0^1 (x+y)\,d\,x \right) d\,y = \int_0^1 \left(\left.\frac{x^2}{2}\right|_0^1 + y\,x\Big|_0^1 \right) d\,y = \int_0^1 \left(\frac{1}{2} + y \right) d\,y = 1.0$$

Аналогічно двовимірним інтегралам, можна обчислювати інтеграли за об'ємом.

Results -> Data Set - More Data Sets -> Grid 3D

Data Sets – Grid 3D 3:

Data – **Source, Function** = Function, All (*два поля*)

В розділі **Parameter Bounds** в полях **Minimum** і **Maximum** всіх трьох параметрів залиште значення 0 і 1.

Використовуючи це джерело даних, обчислимо інтеграл $\iiint\limits_{\text{Grid3D}} x\,d\,x\,d\,y\,d\,z$ по кубу $\text{Grid 3D} = [0,1]\times[0,1]\times[0,1]$.

Перший спосіб полягає у використанні вузла типу `Volume Integration`.

`Results->`**`Derived Values`**`-More Derived Values->Volume Integration`

Derived Values – Volume Integration 3:

`Data` – **Data set** = `Grid 3D 3`

`Expression` – **Expression** = `x`

LMB > Evaluate > Table 1 - Інтеграл

В новій клітинці попередньої таблиці з'явиться результат `0.5`.

x	x, Integral 1	x	x+y	x+y	x
1.5000	1.5000	1.5000	1.0000	1.0000	0.50000

Другий спосіб полягає в використанні ДД `Integral` і подальшому друкуванні результату з вузла типу `Global Evaluation` (або `Point Evaluation`).

`Results -> `**`Data Set`**` - More Data Sets -> Integral`

Data Sets – Integral 3:

`Data` – **Data set**=`Grid 3D 3`

`Results-> Derived Values-> Global Evaluation`

Derived Values – Global Evaluation 4:

`Data` – **Data set**=`Integral 3`

`Expression` – **Expression** = `x`

LMB > Evaluate > Table 1 - Інтеграл

В новій клітинці попередньої таблиці ви отримаєте результат `0.5`.

x	x, Integral 1	x	x+y	x+y	x	x
1.5000	1.5000	1.5000	1.0000	1.0000	0.50000	0.50000

У правильності результату можна переконатися за допомогою простих обчислень.

$$\iiint\limits_{\text{Grid3D}} x\,dx\,dy\,dz = \int_0^1\left(\int_0^1\left(\int_0^1 x\,dx\right)dy\right)dz = \int_0^1\left(\int_0^1 \frac{1}{2}\,dy\right)dz = \frac{1}{2}.$$

Гілки **Data Sets** і **Derived Values**, побудовані попередніми інструкціями, зображені на наступному рисунку.

В гілці **Derived Values**, крім вузлів, призначених для інтегрування, можна використовувати вузли, що дозволяють обчислювати в області джерела даних середні і екстремальні значення. Для прикладу знайдемо середнє, максимальне і мінімальне значення виразу $e^x \sin \pi x$ на відрізку [`-1,2`]. У нас цей відрізок представляє джерело даних **Grid 1D 1**.

Results->**Derived Values** - More Derived Values-> Line Average

Derived Values – Line Average 1:

Data – **Data set** = Grid 1D 1

Label = Різне

Назва вузла зміниться на «Різне»

Expression – **Expression** = exp(x)*sin(pi*x)

LMB > Evaluate - New Table

Буде створена нова таблиця **Table 3** і в її клітині з'явиться середнє значення функції -0.74731.

Для пошуку мінімуму виразу $e^x \sin \pi x$ виконайте наступні інструкції.

Results->**Derived Values** - More Derived Values-> Line Minimum

Derived Values – Line Minimum 1:

Data – **Data set** = Grid 1D 1

Expression – **Expression** = exp(x)*sin(pi*x)

LMB > Evaluate -> Table 3 - Різне

В новій клітині таблиці з'явиться мінімальне значення функції -4.7107.

Для пошуку максимуму виразу $e^x \sin \pi x$ виконайте наступні інструкції.

Results->**Derived Values** - More Derived Values-> Line Maximum

Derived Values – Line Maximum 1:

Data – **Data set** = Grid 1D 1

Expression – **Expression** = exp(x)*sin(pi*x)

LMB > Evaluate -> Table 3 - Різне

В новій клітинці попередньої таблиці з'явиться максимальне значення 1.7330.

exp(x)*sin(pi*x)	exp(x)*sin(pi*x)	exp(x)*sin(pi*x)
-0.74731	-4.7107	1.7330

Так само, як при обчисленні інтегралів, у вас є можливість з джерела даних Grid створити джерело Average, Minimum і Maximum, а потім виконати обчислення за допомогою вузлів Point Evaluation або Global Evaluation.

Виконайте послідовність команд.

Results -> **Data Set** - More Data Sets -> Minimum

Results – Data Sets - Minimum 1:

Data – **Data set** = Grid 1D 1

Results -> **Derived Values** -> Point Evaluation

Derived Values – Point Evaluation 3:

Data – **Data set** = Minimum 1

Expression – **Expression** = exp(x)*sin(pi*x)

LMB > Evaluate -> Table 3 - Різне

В попередній таблиці ще раз з'явиться мінімальне значення функції -4.7107.

Вправа. Використовуючи ДД типу Grid 1D і Line Integration вузли, обчисліть наступні інтеграли.

	Інтеграл	Відповідь
1.	$\int_0^1 \cos(10\cos x)\,dx$	-0.30193
2.	$\int_0^1 x^x\,dx$	0.78343
3.	$\int_0^1 \left(\int_{-\sqrt{1-x^2}}^{\sqrt{1-x^2}} 1\,dy \right) dx$	1.5708

Аналогічно джерел даних типу `Grid` використовуються ДД типу `Mesh`. Але вони створюються в геометрії компонентів, і можуть мати досить складну форму.

Приклад 2. Створіть шаблон порожньої моделі `Blank Model`.

File -> New -> Blank Model

Додайте в модель двовимірний компонент.

Home -> Model -> Add Component – 2D

Геометрія компонента складатиметься з чверті кола.

Geometry -> **Primitives** – Circle

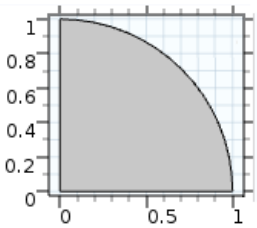
Component 1 - Geometry 1 - Circle 1:

Size and Shape – **Radius** = 1

Size and Shape – **Sector angle** = 90

Position – **Base, x, y** =Center,0,0 (*три поля*)

LMB > Build All Objects



Побудуйте СкЕ сітку.

Component 1 - Mesh 1 : Mesh Settings – **Element size** = Finer

LMB > Build All

Обчисліть площу області. Для цього спочатку створіть ДД типу `Mesh`, а потім використайте його для інтегрування у вузлі `Surface Integration`.

Results-> **Data Set** - More Data Sets->Mesh

Derived Values – Mesh 1:

Mesh – **Mesh** = Mesh 1

Results->**Derived Values**-More Derived Values->Surface Integration

Derived Values – Surface Integration 1:

Data – **Data set** = Mesh 1

Selection – **Selection** = All domains

Expression – **Expression** = 1

Інтегрування одиниці по чверті кола повинно повернуту його площу.

LMB > Evaluate

В новій таблиці з'явиться результат `0.78522`, що близько до точного значення $\pi/4 \approx 0.785398$. Якщо СкЕ сітку зробити дрібнішою, то результат буде точнішим.

Обчислимо периметр фігури, тобто периметр чверті кола, яка вочевидь дорівнює $p = 2 + \pi/2 \approx 3.570796$.

`Results->` **`Derived Values`**`-More Derived Values-> Line Integration`

Derived Values – Line Integration 2:

`Data` – **Data set** = `Mesh 1`

`Selection` – **Selection** = `All boundaries`

`Expression` – **Expression** = `1`

LMB > Evaluate > New Table

1 (m)
3.5707

Тут функціонал «Line Integration» автоматично змусив нас обирати ділянки межі області, тобто криві, вздовж яких буде виконуватися інтегрування. Ми обрали всю границю, і інтеграл від одиниці повернув периметр.

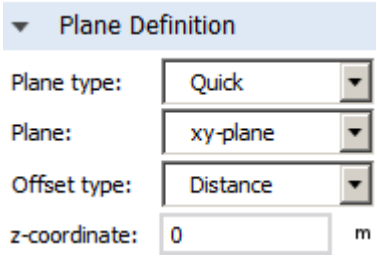
Для обчислення об'ємів нам знадобиться тривимірний компонент. Створіть його в поточному файлі.

`Home` –> Model –> Add Component – 3D

Component 2:> `Geometry` –> Work Plane - Work Plane

Component 2 – Work Plane 1:

Розділ **`Plane Definition`** заповніть так, як показано нижче (всі значення обрані за замовчуванням).



LMB > Show Work Plane

Активуйте вузол

Geometry 2 – Work Plane 1 – Plane Geometry:

На стрічці з'явиться панель `Work Plane`.

`Work Plane` -> **`Primitives`** – Circle

Component 2 – Geometry 2 – Circle 1:

`Size and Shape` – **Radius** = `1`

– **Sector angle** = `90`

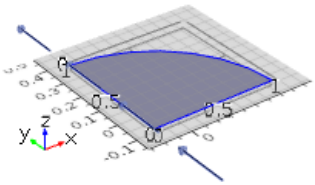
`Position` – **Base, x, y** = `Center, 0, 0` (*три поля*)

LMB > Build Selected

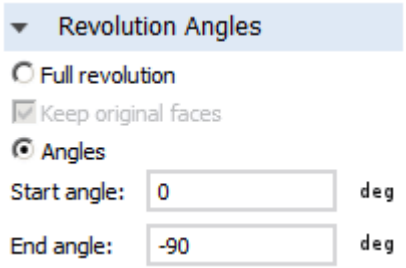
Component 2 – Geometry 2 - Work Plane:>

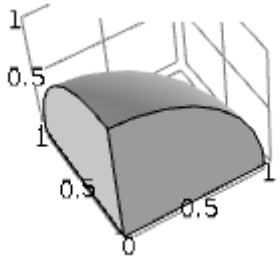
`Geometry` – Operations -> Revolve (*наступний рисунок*)

В дереві моделі з'явиться вузол **Geometry 2 - Revolve 1**.



Простежте, щоб в його ПВ в секції **General** - **Input objects** з'явилося ім'я робочої площини wp1. Розділ **Revolution Angles** заповніть так, як показано нижче.





Виконайте команду Build All Objects і отримайте зображення тіла, яке показано на попередньому рисунку праворуч.

Побудуйте СкЕ сітку.

Component 2 – Mesh 2:

Mesh Settings – **Element Size** = Finer

LMB > Build All

Обчисліть об'єм побудованого тіла. Для цього створіть джерело даних типу Mesh, а потім використайте його у вузлі Volume Integration.

Results -> **Data Set** - More Data Sets -> Mesh.

Derived Values – Mesh 2:

Mesh – **Mesh** = Mesh 2

Results->Derived Values - More Derived Values->Volume Integration

Derived Values – Volume Integration 3:

Data – **Data set** = Mesh 2

Selection – **Selection** = All domains

Expression – **Expression** = 1

LMB > Evaluate > New Table

В новій таблиці з'явиться результат 0.52305, що близько до точного значення $\pi/6 \approx 0.523599$.

Обчислимо площу поверхні тіла.

Results->**Derived Values**-More Derived Values->Surface Integration

Derived Values – Surface Integration 4:

Data – **Data set** = Mesh 2

Selection – **Selection** = All boundaries

Expression – **Expression** = 1

Інтегрування одиниці повинно повернуту площу поверхні тіла.

LMB > Evaluate > New Table

В новій таблиці **Table 4** з'явиться результат 3.9249, що близько до точного значення $3\cdot\frac{\pi}{4}+\frac{\pi}{2}\approx 3.926991$.

Вузол Surface Integration змусив нас обирати ділянки границі тіла, тобто поверхні, вздовж яких буде виконуватися інтегрування. Ми обрали всю межу, і інтеграл від одиниці повернув площу поверхні тіла.

Обчислимо сумарну довжину ребер тіла.

Results-> **Derived Values**-More Derived Values-> Line Integration

Derived Values – Line Integration 5:

Data – **Data set** = Mesh 2

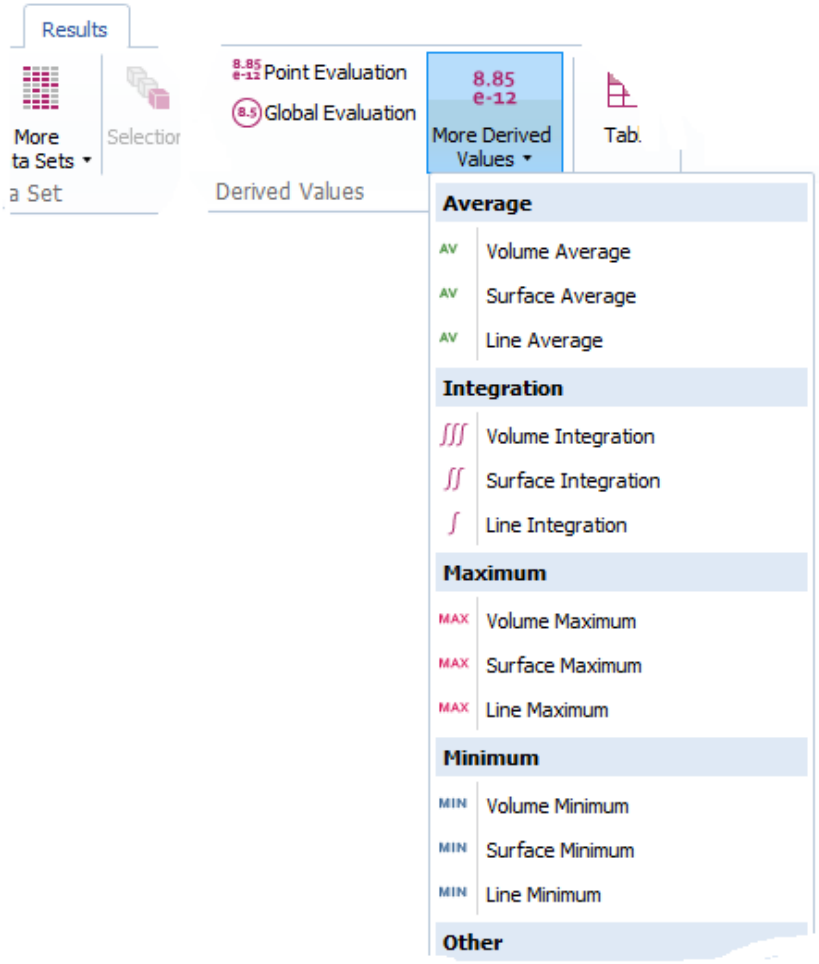
Selection – **Selection** = All edges

Expression – **Expression** = 1

LMB > Evaluate > New Table

В новій таблиці з'явиться результат 7.7118, що близько до точного значення $3+3\cdot\frac{\pi}{2}\approx 7.712389$.

Як бачите, Derived Values вузли Line Integration, Surface Integration та Volume Integration дозволяють обирати ділянки геометрії відповідного рівня (криві, поверхні, частини тіла), і виконують на них інтегрування. Схожим чином використовуються Derived Values вузли типу Average, Maximum і Minimum. Повний список «обчислювальних» команд, використовуваних у вузлі **Derived Values**, можна побачити на стрічці на панелі Results в розділі **Derived Values**, якщо розкрити кнопку-список **More Derived Values**.



Продемонструємо використання деяких з цих команд. Знайдемо середнє, максимальне і мінімальне значення виразу $\frac{1+x+y}{1+e^{1-x-y-z}}$ в тривимірному тілі (частина кулі), яке побудовано нами раніше.

Results->**Derived Values**-More Derived Values->Volume Average

Derived Values – Volume Average 1:

Data – **Data set** = Mesh 2

Selection – **Selection** = Manual, 1 (*два поля*)

Expression – **Expression** = (1+x+y)/(1+exp(1-x-y-z))

LMB > Evaluate - New Table

Буде створена нова таблиця **Table 6** і в її клітині з'явиться середнє значення виразу по об'єму 0.94464.

Для пошуку максимуму виразу $\frac{1+x+y}{1+e^{1-x-y-z}}$ виконайте наступні інструкції.

Results->**Derived Values**-More Derived Values->Volume Maximum

Derived Values – Volume Maximum 1:

Data – **Data set** = Mesh 2

Selection – **Selection** = Manual, 1 (*два поля*)

Expression – **Expression** = (1+x+y)/(1+exp(1-x-y-z))

LMB > Evaluate - New Table

В новій таблиці з'явиться максимальне значення функції 1.5422.

Для пошуку мінімуму виразу $\frac{1+x+y}{1+e^{1-x-y-z}}$ виконайте наступні інструкції.

Results->**Derived Values**-More Derived Values->Volume Minimum

Derived Values – Volume Minimum 1:

Data – **Data set** = Mesh 2

Selection – **Selection** = Manual, 1 (*два поля*)

Expression – **Expression** = (1+x+y)/(1+exp(1-x-y-z))

LMB > Evaluate - New Table

В новій таблиці з'явиться мінімальне значення функції 0.26894.

Обчислення середніх і екстремальних значень функцій по поверхні або ребрам нічим не відрізняється від розрахунків по об'єму за винятком назв відповідних команд і рівня їх дії.

Приклад 3. Обчислимо площу області еліптичного кільця з параболічним вирізом, координати точок якої задовольняють умовам $1 \le \frac{x^2}{9} + \frac{y^2}{4} \le 4$, $y \le x^2$.

Створіть шаблон порожньої моделі Blank Model.

File -> New -> Blank Model

Створіть функцію-заглушку.

Global Definitions : RMB > Functions – Analytic

Function name = f

Definition –**Expression, Arguments** = 1,x (*два поля*)

Створіть двовимірне джерело даних `Grid 2D`.

`Results` -> Data Set - More Data Sets -> Grid 2D

Data Sets – Grid 2D 1:

Data – **Source, Function** = `Function, All` (*два поля*)

Paramemter Bounds – First parameter –
Minimum, Maximum =`-6,6` (*два поля*)

Paramemter Bounds – **Second parameter –**
Minimum, Maximum =`-6,6` (*два поля*)

Resolution – x resolution, y resolution =`250,250` (*два поля*)

Побудуйте область, площу якої будемо обчислювати.

`Home` -> Add Plot Group -> 2D Plot Group

Results – 2D Plot Group 1:

Data– **Data set** = `Grid 2D 1`

`2D Plot Group 1` -> Contour

2D Plot Group 1 – Contour 1:

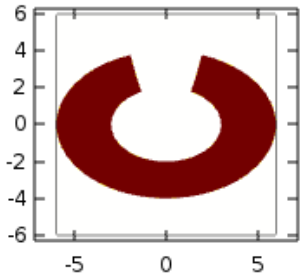
Data– **Data set** =`From Parent`;

Expression – **Expression** =
=`(x^2/9+y^2/4)>=1 && (x^2/9+y^2/4)<=4 && y<x^2`

Levels –**Entry method, Total levels** = `Number of levels, 2`

Coloring and Style – **Contour type** =`Filled`
– **Fill surface outside of contour levels** =
= `True`
– **Coloring** =`Color table`
– **Color table** =`ThermalLight`
– **Reverse color table** = `True`

LMB > Plot



Цей графік являє контурне зображення характеристичної функції області. Для контурів ми використали тільки два рівня значень, тому поза зони еліптичного кільця з вирізом колір джерела даних **Grid 2D 1** білий, а всередині - коричневий.

Тепер обчислимо площу області.

`Results`->More Derived Values->Surface Integration.

Derived Values – Surface Integration 1:

Data – **Data set** = `Grid 2D 1`

Expression – **Expression** =
=`(x^2/9+y^2/4)>=1 && (x^2/9+y^2/4)<=4 && y<x^2`

LMB > Evaluate

В клітині таблиці з'явиться результат 49.721. Оскільки інтегрування виконується по квадрату від характеристичної функції, яка поза областю дорівнює нулю а всередині – одиниці, то ми отримуємо площу відповідної ділянки квадрата, тобто обраної області.

Поверніться до вузла **Data Sets - Grid 2D 1** і змініть в його ПВ наступні поля:

Data Sets - Grid 2D 1:

Resolution – x resolution = 1000

Resolution – y resolution = 1000

Тепер знову обчисліть площу області.

Derived Values – Surface Integration 1: LMB > Evaluate – Table 1

В новій клітині таблиці **Table 1** з'явиться уточнений результат 49.739.

2.2.2. Геометричні застосування інтегрального числення.

Тут ми продемонструємо можливості «постпроцесорних інструментів» системи COMSOL Multiphysics (використовуваних в гілці Results) при розв'язанні класичних математичних задач, таких як обчислення довжин кривих, площ поверхонь і об'ємів тіл.

Довжина дуги плоскої кривої, яка задана в декартових координатах явним рівнянням $y = y(x)$, обчислюється за формулою

$$L = \int_{x_0}^{x_1} \sqrt{1 + (y'(x))^2}\, d\,x, \qquad (1)$$

де x_0 та x_1 визначають початкову та кінцеву точки дуги.

Приклад 1. Обчислимо довжину ланцюгової лінії $y = ch\,x - 1$ на ділянці $-1 \le x \le 1$.

Створіть порожню модель.

File->New->Blank Model

Створіть функцію.

Home -> Definitions – Functions -> Global – Analytic

Function name = f

Definition – Expression, Arguments = cosh(x)-1, x (*два поля*)

Створіть і налаштуйте одновимірне джерело даних Grid 1D.

Results -> Data Set - More Data Sets -> Grid 1D

Data Sets – Grid 1D 1:

Label = Grid 1D [-1,1]

Назва вузла в дереві моделі зміниться на Grid 1D [-1,1].

Data – Source, Function = Function, All (*два поля*)

Paramemter Bounds – Name= x

Paramemter Bounds – Minimum, Maximum = -1,1 (*два поля*)

Для обчислення інтегралу виконайте наступну послідовність інструкцій.

Results -> **Derived Values**-More Derived Values->Line Integration

Derived Values – Line Integration 1:

Label = Довжина

Назва вузла зміниться на «Довжина».

Data – **Data set** = Grid 1D [-1,1]

Expression – **Expression** = sqrt(1+d(f(x),x)^2)

LMB > Evaluate - New Table

Буде створена таблиця і в її клітині з'явиться результат: 2.3504.

Побудуйте графік кривої.

Home->Add Plot Group ->1D Plot Group

Results – 1 D Plot Group 1:

Data– **Data set**= Grid 1D [-1,1]

Поки цей вузол активний, на стрічці активна панель 1D Plot Group 1.

1D Plot Group 1-> Line Graph

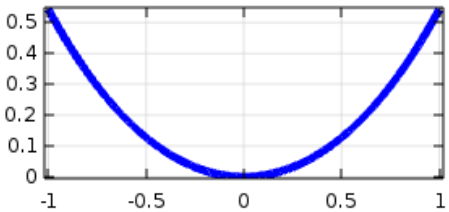
1D Plot Group 1 – Line Graph 1:

Data– **Data set** = From Parent

y-Axis Data – **Expression** = f(x)

x-Axis Data – **Parameter, Expression** = Expression, x (*два поля*)

LMB >Plot



Збережіть файл, наприклад, з ім'ям curves01.mph, але не закривайте його.

■

Довжина дуги плоскої кривої, яка задана в параметричному вигляді $x = x(t)$, $y = y(t)$, обчислюється за формулою

$$L = \int_{t_0}^{t_1} \sqrt{(x'(t))^2 + (y'(t))^2}\, dt, \qquad (2)$$

де t_0 та t_1 – значення параметра t в початковій і кінцевій точках дуги.

Приклад 2. Обчислимо довжину астроїди $x = \cos^3 t$, $y = \sin^3 t$, $0 \le t \le 2\pi$, і площу області, яку вона обмежує.

Продовжимо роботу з файлом curves01.mph. Створіть в ньому нове одновимірне джерело даних.

Results -> Data Set - More Data Sets -> Grid 1D

Data Sets – Grid 1D 2:

Label = Grid 1D [0,2*pi]

Назва вузла зміниться на Grid 1D [0,2*pi].

Data – **Source, Function** = Function, All (*два поля*)

Paramemter Bounds – **Name**= x

Paramemter Bounds – **Minimum, Maximum** = 0, 2*pi (*два поля*)

Побудуйте графік кривої.

Home -> Add Plot Group ->1D Plot Group

Results – 1 D Plot Group 2:

Data– Data set= Grid 1D [0,2*pi]

Поки вузол **1 D Plot Group 2** активний, на стрічці активна панель 1D Plot Group 2.

1D Plot Group 2 -> Line Graph

1D Plot Group 2 – Line Graph 1:

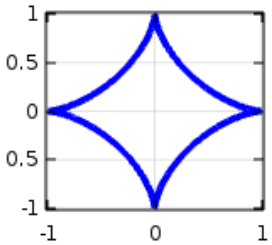
Data– Data set = From Parent

y-Axis Data – Expression = cos(x)^3

x-Axis Data – Parameter = Expression

x-Axis Data – Expression = sin(x)^3

LMB >Plot



Обчислимо довжину кривої.

Results-> **Derived Values**-More Derived Values ->Line Integration

Derived Values – Line Integration 2:

Data – Data set = Grid 1D [0,2*pi]

Expression – Expression =

= sqrt(d(cos(x)^3,x)^2+d(sin(x)^3,x)^2)

LMB > Evaluate -> Table 1- Довжина

В комірці попередньої таблиці з'явиться значення довжини 6.0000.

Площа області, яка обмежена кусково-гладкою замкнутою кривою L, може бути обчислена за допомогою криволінійного інтеграла 2-го роду:

$$S = \frac{1}{2}\oint_L x\,d\,y - y\,d\,x \qquad (3)$$

Якщо крива задана параметрично $x = x(t),\ y = y(t),\ t_{\min} \le t \le t_{\max}$, то формула (3) набуває вигляду

$$S = \frac{1}{2}\int_{t_{\min}}^{t_{\max}} \left(x(t)y'(t) - y(t)x'(t)\right)d\,t \qquad (4)$$

Обчислимо площу, яка обмежена астроїдою.

Results->**Derived Values**-More Derived Values -> Line Integration

Derived Values – Line Integration 3:

Label = Площа астроіди

Data – Data set = Grid 1D [0,2*pi]

Expression – Expression =

= (cos(x)^3*d(sin(x)^3,x)-sin(x)^3*d(cos(x)^3,x))/2

LMB > Evaluate -> Table 1- Довжина

В таблиці з'явиться значення площі 1.1781 (точна відповідь $3\pi/8 \approx 1.178097$).

■

Довжина дуги кривої, заданої в полярних координатах рівнянням $r = r(\varphi)$, обчислюється за формулою

$$L = \int_{\varphi_0}^{\varphi_1} \sqrt{r^2(\varphi) + (r'(\varphi))^2}\, d\varphi, \qquad (5)$$

де φ_0 та φ_1 – значення кутового параметра φ в початковій і кінцевій точках дуги.

Приклад 3. Обчислимо довжину логарифмічної спіралі $r = e^{\varphi/20}$ на відрізку $-4\pi \le \varphi \le 4\varphi$.

Продовжимо роботу з файлом curves01.mph. Створіть в ньому нове одновимірне джерело даних.

Results -> Data Set - More Data Sets -> Grid 1D

Data Sets – Grid 1D 3:

Label = Grid 1D [-4*pi,4*pi]

Назва вузла в дереві моделі зміниться на Grid 1D [-4*pi,4*pi].

Data – Source, Function = Function, All (*два поля*)

Paramemter Bounds – Name= theta

Paramemter Bounds – Minimum, Maximum = -4*pi,4*pi (*два поля*)

Побудуйте графік кривої.

Home -> Add Plot Group -> Polar Plot Group

Results – Polar Plot Group 3:

Data– Data set= Grid 1D [-4*pi,4*pi]

Axis – Manual axis limits = True

Axis – r minimum, r maximum =0,2 (*два поля*)

Polar Plot Group 3 -> **Add Plot** - Line Graph

Polar Plot Group 3 – Line Graph 1:

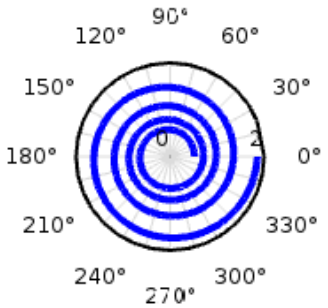
Data– Data set = From Parent

r-Axis Data – Expression = exp(theta/20)

θ-Axis Data – Parameter = Expression

θ-Axis Data – Expression = theta

LMB >Plot



Обчислимо довжину цієї кривої.

Results->**Derived Values**-More Derived Values -> Line Integration

Derived Values – Line Integration 4:

Data – **Data set** = Grid 1D [-4*pi,4*pi]

Expression – **Expression** =

= sqrt(exp(theta/20)^2+d(exp(theta/20),theta)^2)

LMB > Evaluate -> Table 1- Довжина

В комірці попередньої таблиці з'явиться значення довжини 26.853.

■

Довжина дуги просторової кривої, заданої параметрично $x = x(t), y = y(t), z = z(t)$, обчислюється за формулою

$$L = \int_{t_0}^{t_1} \sqrt{(x'(t))^2 + (y'(t))^2 + (z'(t))^2}\, d\,t, \qquad (6)$$

де t_0 та t_1 – значення параметра t в початковій і кінцевій точках дуги.

Приклад 4. Обчислимо довжину гвинтової лінії

$$x = \cos t,\ y = \sin t,\ z = t/12 - 1$$

від точки $t = 0$ до точки $t = 6\pi$.

Обчислення довжини кривої можна виконати по-різному.

Перший спосіб полягає в побудові одновимірного джерела даних Grid з подальшим обчисленням інтеграла (6). Для цього виконайте в файлі curves01.mph послідовність наступних інструкцій.

Results -> Data Set - More Data Sets -> Grid 1D

Data Sets – Grid 1D 4:

Label = Grid 1D [0,6*pi]

Назва вузла в дереві моделі зміниться на Grid 1D [0,6*pi].

Data – **Source, Function** = Function, All (*два поля*)

Parameter Bounds – **Name** = t

Parameemter Bounds – **Minimum, Maximum** = 0, 6*pi (*два поля*)

Зауваження. Тут використано погану практику називати параметр ДД ім'ям t. Зазвичай цим ідентифікатором в динамічних задачах позначають час. Оскільки у нас змінної часу немає, то складнощів не виникає.

Обчислимо довжину кривої.

Results->**Derived Values**-More Derived Values -> Line Integration

Derived Values – Line Integration 5:

Data – **Data set** = Grid 1D [0,6*pi]

Expression – **Expression** =

= sqrt(d(cos(t),t)^2+d(sin(t),t)^2+d(t/12-1,t)^2)

LMB > Evaluate -> Table 1- Довжина

В комірці таблиці з'явиться значення довжини 18.915.

Другий спосіб полягає в обчисленні криволінійного інтеграла першого роду

$$L = \int_{s_0}^{s_1} d\,s, \qquad (7)$$

де s - натуральний параметр (довжина дуги). Для цього потрібно мати джерело даних, яке представляє криву. Але спочатку потрібно створити тривимірне джерело даних `Grid 3D`, всередині якого буде розташована крива.

`Results` -> Data Set - More Data Sets -> Grid 3D

Data Sets – Grid 3D 5:

Label = `Grid 3D [-1,1][-1,1][-1,1]`

Data – **Source, Function** = `Function, All` (*два поля*)

В розділі **`Parameter Bounds`** в полях **Minimum** і **Maximum** всіх трьох параметрів x, y, z введіть `-1` і `1`.

Тепер створіть джерело даних, яке буде представляти криву.

`Results`-> Data Set - More Data Sets -> Parameterized Curve 3D.

Data Sets – Parameterized Curve 3D 1:

Data– **Data set**= `Grid 3D [-1,1][-1,1][-1,1]`

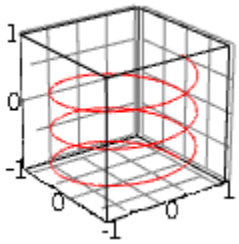
Parameter – **Name**=`s`

– **Minimum, Maximum** =`0, 6*pi` (*два поля*)

Expressions – **x, y, z** =`cos(s),sin(s),s/12-1` (*три поля*)

– **Only evaluate globally defined expressions**=`True`

Натисніть кнопку Plot і отримайте зображення джерела даних в формі гвинтової лінії (спіралі).



Тепер обчисліть довжину кривої за формулою (7).

`Results`->**`Derived Values`**-More Derived Values -> Line Integration

Derived Values – Line Integration 6:

Data – **Data set** = `Parameterized Curve 3D 1`

Expression – **Expression** = `1`

LMB > Evaluate -> Table 1- Довжина

В комірці таблиці з'явиться попередній результат `18.915`.

Звідсіля можна зробити висновок, що в `Line Integration` вузлах обчислюються криволінійні інтеграли першого роду.

Зауваження. В поточному прикладі у вузлі **Parameterized Curve 3D 1** діапазон зміни параметра $0 \le s \le 6\pi$ обраний так, щоб крива цілком потрапляла всередину куба $[-1,1]\times[-1,1]\times[-1,1]$, який представлено джерелом даних `Grid 3D`. Якщо діапазон зміни параметра s задати так, що крива буде виходити за межі куба, наприклад, $0 \le s \le 10\pi$, то встановлення в ПВ вузла **Parameterized Curve 3D 1** в розділі **`Expressions`** прапорця **Only evaluate globally defined expressions** дозволить враховувати всю криву. Зокрема, у відповідному вузлі `Line Integration` обчислення дасть повну довжину

кривої. Якщо цей прапорець зняти, то буде обчислена довжина тільки частини кривої, яка розташована всередині куба.

▪

В COMSOL Multiphysics можна використовувати кілька способів обчислення площ поверхонь і об'ємів тіл. Один з них полягає в створенні геометрії двовимірної області або тривимірного тіла з СкЕ сіткою. Потім, на основі цієї сітки, створюється джерело даних `Mesh`, яке використовується у вузлах `Surface Integration` або `Volume Integration`. Цей підхід зручно використовувати в тих задачах, де СкЕ сітка вже побудована. В прикладах попереднього параграфа ми вже обчислювали площу чверті кола, використовуючи двовимірне джерело даних `Mesh`. Там же ми розглянули приклад обчислення площі поверхні і об'єму частини кулі, використовуючи тривимірне джерело даних `Mesh`.

Інший спосіб полягає у використанні формул інтегрування. Наприклад, якщо площі поперечних перерізів тривимірного тіла, перпендикулярних до осі X, дорівнюють $S(x)$, то об'єм такого тіла обчислюється за формулою $V=\int_{x_1}^{x_2} S(x)\,d\,x$. Окремим випадком цього є формула обчислення об'єму тіла, утвореного обертанням явно заданої кривої $y=f(x)$ навколо осі X. Тоді $V=\pi\int_{x_1}^{x_2} f^2(x)\,d\,x$. Об'єм вертикального циліндричного тіла, яке має своєю основою область D на площині XY, і обмежене зверху поверхнею $z=f(x,y)$, обчислюється подвійним інтегралом $V=\iint_D f(x,y)\,d\,x\,d\,y$.

Третій спосіб полягає у використанні характеристичної функції області, розташованої всередині тривимірного джерела даних `Grid 3D`. Дійсно, об'єм V тривимірного тіла G можна обчислювати як інтеграл $V=\iiint_G d\,x\,d\,y\,d\,z$. Якщо створити характеристичну функцію області $\chi_G(x,y,z)=\begin{cases}1\,,\,(x,y,z)\in G\\ 0\,,\,(x,y,z)\notin G\end{cases}$, то останню формулу для об'єму можна записати у вигляді

$$V=\iiint_{\text{Grid 3D}} \chi_G(x,y,z)\,d\,x\,d\,y\,d\,z\,, \tag{8}$$

де `Grid 3D` являє область прямого паралелепіпеда, що цілком містить в собі тіло G.

У прикладах, які будуть розглядатися нижче, ми будемо обчислювати об'єми тіл, які утворені за допомогою теоретико-множинних операцій об'єднання, перетину і віднімання тіл (областей), що мають простішу форму. Часто, маючи характеристичні функції цих підобластей, нескладно записати характеристичну функцію $\chi(x,y,z)$ результівної області. Однак для прийнятного зображення поверхні тіла цього мало. Бажано мати неявне рівняння $F(x,y,z)=0$, бо в такому випадку за допомогою графічного вузла

типу `Isosurface` можна досить якісно побудувати «нульову» поверхню рівня функції $F(x,y,z)$. Але для застосування цього підходу потрібно вміти записувати неявне рівняння складної поверхні.

Нехай для кожної області Ω_i $(i=1,2,\ldots)$ відома функція $\omega_i(x,y,z)$, яка позитивна всередині області, негативна - зовні, і дорівнює нулю на її межі. Функцію $\omega_i(x,y,z)$, що має вказані властивості, ми будемо називати ідентифікуючої функцією (або просто ідентифікатором) області Ω_i. Очевидно, що неявне рівняння поверхні області Ω_i можна тоді записати у вигляді $\omega_i(x,y,z)=0$. Крім того, побудуємо функції

$$
\begin{aligned}
&ir(u,v)=\frac{1}{2}\left(u+v-|u-v|\right),\ \text{(region intersection);}\\
&ur(u,v)=\frac{1}{2}\left(u+v+|u-v|\right),\ \text{(region union);}\\
&dr(u,v)=\frac{1}{2}\left(u-v-|u+v|\right),\ \text{(region difference).}
\end{aligned}
\tag{9}
$$

Тоді справедливі наступні твердження:

1. Якщо $\Omega_{ir}=\Omega_1\bigcap\Omega_2$, то функція $\omega_{ir}(x,y,z)=ir(\omega_1(x,y,z),\omega_2(x,y,z))$ буде ідентифікуючої функцією області Ω_{ir};
2. Якщо $\Omega_{ur}=\Omega_1\bigcup\Omega_2$, то функція $\omega_{ur}(x,y,z)=ur(\omega_1(x,y,z),\omega_2(x,y,z))$ буде ідентифікуючої функцією області Ω_{ur};
3. Якщо $\Omega_{dr}=\Omega_1\setminus\Omega_2$, то функція $\omega_{dr}(x,y,z)=dr(\omega_1(x,y,z),\omega_2(x,y,z))$ буде ідентифікуючої функцією області Ω_{dr}.

Нехай тіло Ω, як множина точок в просторі, утворюється за допомогою деяких операцій (перетин, об'єднання, доповнення і т.д.) над підобластями Ω_i, що мають простішу форму ніж Ω. Припустимо що для кожної з підобластей Ω_i відома ідентифікуюча функція $\omega_i(x,y,z)$. Тоді суперпозиція функцій (9) і ідентифікаторів ω_i областей Ω_i дозволяє побудувати ідентифікуючу функцію результівної області Ω. Наприклад, якщо $\Omega=(\Omega_1\bigcap\Omega_2\bigcup\Omega_3)\bigcap\bar{\Omega}_4$, то $\omega_\Omega=ir(ur(ir(\omega_1,\omega_2),\omega_3),-\omega_4)$, де для стислості ми опустили аргументи x, y, z всіх функцій ω_i $(i=1,2,3,4)$. Як наслідок, неявне рівняння поверхні тіла Ω матиме вигляд $\omega_\Omega(x,y,z)=0$. Цей підхід працює також для плоских областей, ідентифікуючі функції яких залежать від двох просторових координат x, y. В такому випадку криві, побудовані по неявним рівнянням $\omega_\Omega(x,y)=0$, зображатимуть контури плоских областей.

Приклад 5. Обчислимо об’єм спільної частини циліндра $x^2+y^2\le 2x$ і еліптичного параболоїда $y^2+z^2\le 4x$.

Створіть порожню модель.

File->New->Blank Model

Створіть функцію $ir(u,v)$.

Home -> Definitions – Functions -> Global - Analytic

Function name = ir

Definition – Expression = (x+y-abs(x-y))/2

Definition – Arguments = x,y

Тут функція $ir(x,y)$ побудована по першій формулі (9).

Створіть джерело даних Grid 3D.

Results -> Data Set - More Data Sets -> Grid 3D

Data Sets – Grid 3D 1:

Label = Grid 3D [-1,3][-2,2][-3,3]

Назва вузла зміниться на Grid 3D [-1,3][- 2,2][- 3,3].

Data – Source, Function = Function, All (*два поля*)

В розділі **Parameter Bounds** в поля **Minimum** і **Maximum** введіть діапазони зміни параметрів $-1 \le x \le 3$, $-2 \le y \le 2$, $-3 \le z \le 3$.

Зобразіть тіло.

Home -> **Results** - Add Plot Group ->3D Plot Group

Results – 3 D Plot Group 1:

Data– Data set =Grid 3D [-1,3][-2,2][-3,3]

Plot Settings–Plot Data set edges = False

3D Plot Group 1-> Isosurface

3D Plot Group 1 – Isosurface 1:

Data– Data set = From Parent

y-Axis Data – Expression = ir(2*x-x^2-y^2,4*x-y^2-z^2)

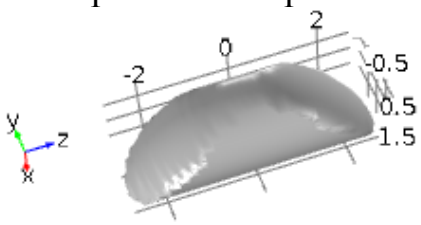
Levels – Entry method, Levels = Levels, 0 (*два поля*)

Coloring and Style – Coloring, Color =Uniform, Gray (*два поля*)

Coloring and Style – Color legend = False

В якості ідентифікуючої функції області $x^2+y^2 \le 2x$ використовується функція $\omega_1(x,y,z)=2x-x^2-y^2$, оскільки всередині цього циліндра вираз $2x-x^2-y^2$ позитивний. Аналогічно, ідентифікуючою функцією внутрішньої області еліптичного параболоїда є $\omega_2(x,y,z)=4x-y^2-z^2$. Тоді $\omega(x,y,z)=ir(\omega_1(x,y,z),\omega_2(x,y,z))$ буде ідентифікуючої функцією тіла, отриманого перетином циліндра і параболоїда.

Виконайте команду Plot і отримайте зображення тіла.



Характеристичну функцію області можна записати у вигляді

$$\chi(x,y,z)=(2x-x^2-y^2>0)\ \&\&\ (4x-y^2-z^2>0).$$

Обчисліть об'єм тіла за формулою (8).

Results -> **Derived Values** - More Derived Values ->
Volume Integration

Derived Values – Volume Integration 1:

Label = Volume

Data – Data set = Grid 3D [-1,3][-2,2][-3,3]

Expression – Expression = (2*x-x^2-y^2>0) && (4*x-y^2-z^2>0)

LMB > Evaluate -> New Table

Буде створена таблиця в якій з'явиться результат 11.588, що достатньо близько до точного значення $\frac{2}{3}(8+3\pi)\approx 11.6165$. Якщо в ПВ вузла **Data Sets – Grid 3D [-1,3][-2,2][-3,3]** в розділі **Resolution** збільшити розрізнення, то обчислення об'єму дасть точніше значення.

Збережіть файл з ім'ям volumes01.mph, але не закривайте його.

■

Приклад 6. Обчислимо об'єм спільної частини циліндрів $x^2+y^2\le 1$ і $y^2+z^2\le 1$.

Продовжимо роботу з файлом volumes01.mph. Спочатку зобразимо тіло.

Home -> Add Plot Group ->3D Plot Group

Results – 3 D Plot Group 2:

Data– Data set =Grid 3D [-1,3][-2,2][-3,3]

Plot Settings–Plot Data set edges = False

3D Plot Group 2->Isosurface

3D Plot Group 2 – Isosurface 1:

Data– Data set = From Parent

y-Axis Data – Expression = ir(1-x^2-y^2,1-y^2-z^2)

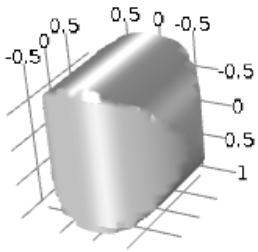
Тут $ir\left(1-x^2-y^2,1-y^2-z^2\right)$ є ідентифікуючої функцією тіла, отриманого перетином двох циліндрів.

Levels – Entry method, Levels = Levels, 0 (*два поля*)

Coloring and Style – Coloring, Color =Uniform,Gray (*два поля*)

Coloring and Style – Color legend = False

LMB >Plot



Характеристичну функцію області можна записати у вигляді

$$\chi(x,y,z)=\left(1-x^2-y^2>0\right)\ \&\&\ \left(1-y^2-z^2>0\right).$$

Обчисліть об'єм тіла за формулою (8).

Results -> **Derived Values** - More Derived Values ->
Volume Integration

Derived Values – Volume Integration 2:

Data – **Data set** = Grid 3D [-1,3][-2,2][-3,3]

Expression – **Expression** = (1-x^2-y^2>0) && (1-y^2-z^2>0)

LMB > Evaluate ->Table 1 - Volume

В новій комірці попередньої таблиці з'явиться результат 5.4320.

■

Приклад 7. Обчислимо об'єм тіла, що є об'єднанням циліндрів $x^2+y^2\le 1, |z|\le 2$ та $x^2+z^2\le 1, |y|\le 2$.

Продовжимо роботу з файлом volumes01.mph. Спочатку створіть функцію $ur(u,v)$ відповідно з другою формулою (9).

Home -> Definitions – Functions -> Global – Analytic

Function name = ur

Definition – **Expression** = (x+y+abs(x-y))/2

Definition – **Arguments** = x,y

Тепер зобразіть тіло.

Home -> Add Plot Group ->3D Plot Group

Results – 3 D Plot Group 3:

Data– **Data set** =Grid 3D [-1,3][-2,2][-3,3]

Plot Settings–**Plot Data set edges** = False

3D Plot Group 3->Isosurface

3D Plot Group 3 – Isosurface 1:

Data– **Data set** = From Parent

y-Axis Data – **Expression** =

= ur(ir(1-x^2-z^2,2- abs(y)),ir(1-x^2-y^2, 2-abs(z)))

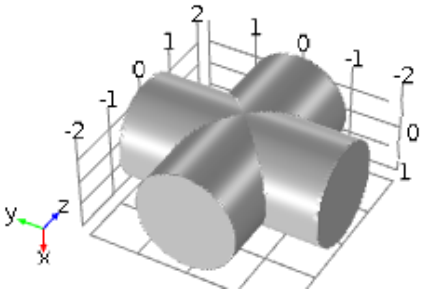
Levels – **Entry method, Levels** = Levels, 0 (*два поля*)

Coloring and Style – **Coloring, Color** =Uniform,Gray (*два поля*)

Coloring and Style – **Color legend** = False

Тут вираз $ir\left(1-x^2-z^2, 2-|y|\right)$ являє ідентифікуючу функцію циліндра, обмеженого уздовж осі Y площинами $y=\pm 2$. Аналогічно, вираз $ir\left(1-x^2-y^2, 2-|z|\right)$ являє ідентифікатор другого обмеженого циліндра. Використання цих виразів в якості аргументів функції $ur(\cdot,\cdot)$ створює функцію ідентифікації тіла, отриманого об'єднанням двох обмежених циліндрів.

В ПВ вузла **Grid 3D [-1,3][-2,2][-3,3]** в розділі **Resolution** збільште роздільність за всіма трьома координатами до 100. Виконайте команду Plot і отримайте зображення тіла.



Майте на увазі, що ДД **Grid 3D [-1,3][-2,2][-3,3]** тепер містить $100\times100\times100=1000000$ (мільйон) елементів і, якщо у вас малопотужний комп'ютер, то побудова зображення може затягнутися.

Характеристичну функцію тіла можна записати у вигляді

$$\chi(x,y,z)=\left(1-x^2-z^2\geq0\ \&\&\ 2-|y|\geq0\right)\ ||\ \left(1-x^2-y^2\geq0\ \&\&\ 2-|z|\geq0\right).$$

Обчисліть об'єм тіла за формулою (8).

Results -> **Derived Values** - More Derived Values -> Volume Integration

Derived Values – Volume Integration 3:

Data – **Data set** = Grid 3D [-1,3][-2,2][-3,3]

Expression – **Expression** = (1-x^2-z^2>=0 && 2- abs(y)>=0) || (1-x^2-y^2>=0 && 2-abs(z)>=0)

LMB > Evaluate ->Table 1 - Volume

В комірці попередньої таблиці з'явиться результат 19.728.

■

Для графічного зображення плоских областей в системі COMSOL Multiphysics, навпаки, зручно використовувати характеристичні функції (а не ідентифікуючі).

Приклад 8. Обчислення площі плоскої області, яка отримана об'єднанням двох одиничних кіл з центрами в точках (0,0) і (1,0), з подальшим видаленням зони одиничного кола з центром в точці (2,0).

Продовжимо роботу з файлом volumes01.mph. Спочатку створіть двовимірне джерело даних Grid 2D.

Results -> Data Set - More Data Sets -> Grid 2D

Data Sets – Grid 2D 2:

Label = Grid 2D [-1,3][-1,1]

Назва вузла зміниться на Grid 2D [-1,3][-1,1].

Data– **Source**, **Function** = Function, All (*два поля*)

В розділі **Parameter Bounds** в поля **Minimum** і **Maximum** введіть діапазони зміни параметрів $-1\leq x\leq3$, $-1\leq y\leq1$..

Resolution– **x resolution, y resolution** = 200,200 (*два поля*)

Тепер зобразіть область.

Home -> Add Plot Group ->2D Plot Group

Results – 2 D Plot Group 4:

Data– **Data set** = Grid 2D [-1,3][-1,1]

Plot Settings–**Plot Data set edges** = True

2D Plot Group 4 -> Surface

2D Plot Group 4 – Surface 1:

Data– **Data set** = From Parent

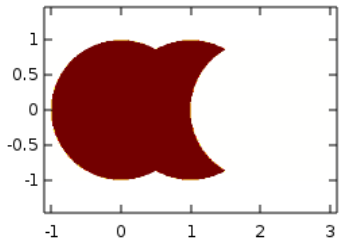
Expression – **Expression** = 1-x^2-y^2>=0 || 1-(x-1)^2-y^2>=0)&& (1-(x-2)^2-y^2<=0)

Coloring and Style – **Coloring** =Color table

– **Color** = ThermalLight

– **Color legend** = `True`
– **Reverse color table** = `True`

LMB > Plot



Обчисліть площу за формулою

$$S = \iint\limits_{\text{Grid2D}} \chi(x,y)\,d\,x\,d\,y, \qquad (10)$$

де $\chi(x,y)$ - характеристична функція досліджуваної області, а Grid 2D є прямокутником, що цілком містить в собі цю область.

Results -> **Derived Values** - More Derived Values -> Surface Integration

Derived Values – Surface Integration 4:

Data – **Data set** = `Grid 2D [-1,3][-1,1]`

Expression – **Expression** = `(1-x^2-y^2>=0||1-(x-1)^2-y^2>=0) && (1-(x-2)^2-y^2<=0)`

LMB > Evaluate ->Table 1 - Volume

В комірці попередньої таблиці з'явиться значення площі `3.8229`.

■

Нехай поверхня задана явним рівнянням $z = f(x,y)$. Площа ділянки цієї поверхні, яка проектується на площину *xOy* у вигляді області *D*, обчислюється за формулою

$$S = \iint\limits_{D} \sqrt{1 + f_x^2 + f_y^2}\,d\,x\,d\,y \qquad (11)$$

Якщо область D повністю розташована всередині прямокутника Grid 2D, то, використовуючи характеристичну функцію $\chi_D(x,y)$ області D, формулу (11) можна переписати у вигляді

$$S = \iint\limits_{\text{Grid 2D}} \chi_D(x,y)\cdot\sqrt{1 + f_x^2 + f_y^2}\,d\,x\,d\,y \qquad (12)$$

Приклад 9. Площа зрізу кругового циліндра $x^2 + y^2 \le 1$ площиною $x + y + z - 3 = 0$.

Продовжимо роботу з файлом `volumes01.mph`. Створіть нове джерело даних `Grid 3D`.

`Results` -> Data Set - More Data Sets -> Grid 3D

Data Sets – Grid 3D 3:

Label = `Grid 3D [-1,1][-1,1][0,5]`

Назва вузла зміниться на `Grid 3D [-1,1][-1,1][0,5]`.

Data– **Source**, **Function** = `Function, All` (*два поля*)

В розділі **Parameter Bounds** в поля **Minimum** і **Maximum** введіть діапазони зміни параметрів $-1 \le x \le 1,\ -1 \le y \le 1,\ 0 \le z \le 5$.

Тепер зобразіть тіло.

Home **-> Results -** Add Plot Group ->3D Plot Group

Results – 3 D Plot Group 5:

Data– Data set =Grid 3D [-1,1][-1,1][0,5]

Plot Settings – Plot Data set edges = False

3D Plot Group 5 -> Isosurface

3D Plot Group 5 – Isosurface 1:

Data– Data set = From Parent

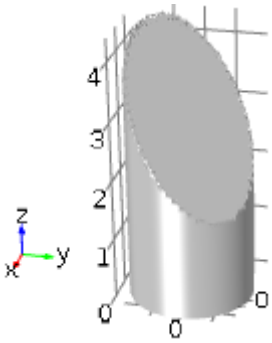
y-Axis Data – Expression = ir(1-x^2-y^2,3-x-y-z)

Levels – Entry method, Levels = Levels, 0 (*два поля*)

Coloring and Style – Coloring, Color =Uniform,Gray (*два поля*)

Coloring and Style – Color legend = False

LMB > Plot



Зверніть увагу на те, що вираз $ir\left(1-x^2-y^2,-x-y-z\right)$ являє ідентифікуючу функцію області «зрізаного» зверху, але необмеженого знизу циліндра. Обчислимо площу верхнього зрізу за формулою (12).

Results **-> Derived Values -** More Derived Values **->** Surface Integration

Derived Values – Surface Integration 5:

Data – Data set = Grid 2D [-1,3][-1,1]

Expression – Expression = sqrt(3)*(1-x^2-y^2>=0)

Тут ми використовуємо двовимірне джерело даних (інтегрування по прямокутнику), а характеристична функція плоскої області (одиничного кола) задається у вигляді $\chi_D(x,y)=\left(1-x^2-y^2 \ge 0\right)$. Рівняння плоского зрізу має вигляд $z=3-x-y$, і підкореневий вираз в (12) дорівнює $\sqrt{1+z_x'^2+z_y'^2}=\sqrt{3}$. Тому в поле **Expression** ми записали вираз $\sqrt{3}\left(1-x^2-y^2 \ge 0\right)$.

LMB > Evaluate ->Table 1 - Volume

В новій комірці таблиці **Table 1** з'явиться значення площі 5.4347, що близько до точного значення $\pi\sqrt{3} \approx 5.4414$.

■

Для обчислення площі поверхні S, заданої в параметричному вигляді $\mathbf{r}=\mathbf{r}(u,v)$, можна використовувати поверхневий інтеграл першого роду

$$A=\iint\limits_S d\,s=\iint\limits_D |\mathbf{r}'_u\times\mathbf{r}'_v|\,d\,u\,d\,v, \qquad (13)$$

де D – область в площині параметрів u, v. Якщо потрібно обчислити площу частини Ω поверхні S, то в системі COMSOL Multiphysics зручно використовувати формулу

$$A=\iint\limits_S \chi_\Omega(u,v)d\,s, \qquad (14)$$

де $\chi_\Omega(u,v)$ – характеристична функція ділянки поверхні Ω.

Приклад 10. На прямому гелікоїді $x=u\cos v$, $y=u\sin v$, $z=v/2$ обчислимо площу чотирикутника, обмеженого кривими $u=0, u=1, v=0, v=1$.

Продовжимо роботу з файлом `volumes01.mph`. Спочатку побудуємо поверхню прямого гелікоїда. Для цього створимо джерело даних `Parameterized Surface`.

`Results` -> Data Set - More Data Sets -> Parameterized Surface

Data Sets – Parameterized Surface 1:

Data– **Data set**= `Grid 3D [-1,3][-2,2][-3,3]`

Paramemter Bounds – **First parameter** –

– **Name, Minimum, Maximum** = `u,-1,1` (*три поля*)

Paramemter Bounds – **Second parameter** –

– **Name, Minimum, Maximum**= `v,-pi,pi` (*три поля*)

Діапазони зміни параметрів `u`, `v` взято ширшими, ніж в умові, щоб побудувати більшу частину гелікоїда.

Expressions – **x** =`u*cos(v)`

– **y** =`u*sin(v)`

– **z** =`v/2`

Resolution– **u resolution, v resolution** = `200,200` (*два поля*)

Виконайте команду Plot, і отримайте зображення джерела даних в формі поверхні прямого гелікоїда.

Тепер зобразимо ту частину поверхні, площа якої буде обчислюватися.

`Home` -> **Results** - Add Plot Group ->3D Plot Group

Results – 3 D Plot Group 6:

Data– **Data set** = `Parameterized Surface 1`

Plot Settings – **Plot Data set edges** = `False`

`3D Plot Group 6` -> Surface

3D Plot Group 6 – Surface 1:

Data – **Data set** = `From Parent`

Expression–**Expression** =`(0<=u && u<=1) && (0<=v && v<=0)`

Тут використана характеристична функція ділянки гелікоїда $0\le u\le 1, 0\le v\le 1$.

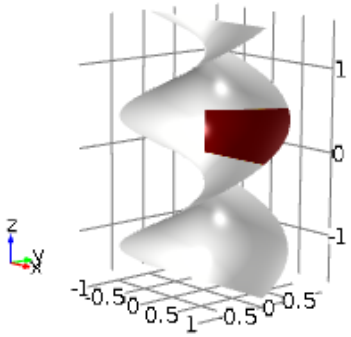
Coloring and Style – **Coloring** = `Color table`

– **Color** = `ThermalLight`

– **Color legend** = `False`

– **Reverse color table** = `True`

LMB > Plot



Визначимо площу ділянки $0 \le u \le 1, 0 \le v \le 1$ поверхні гелікоїда за формулою (14).

Results -> **Derived Values** - More Derived Values -> Surface Integration

Derived Values – Surface Integration 6:

Data – **Data set** = `Parameterized Surface 1`

Expression – **Expression** = `0<=u && u<=1 && 0<=v && v<=1`

LMB > Evaluate ->Table 1 – Volume

В комірці попередньої таблиці з'явиться значення площі `0.74708`, що досить близько до точного значення $\frac{1}{8}\left(2\sqrt{5} + \operatorname{arcsinh}(2)\right) \approx 0.7395$.

■

Зауваження 1. Коли тіло побудовано в геометрії моделі, то немає потреби в використанні характеристичних функцій областей, бо в якості ДД для інтегралів можна зразу обирати ДД `Mesh`, або `Study 1/Solution 1`, яке «містить» в собі СкЕ сітку.

Зауваження 2. Знаходити об'єми, площі поверхонь і довжини ребер тіл, створених в геометрії моделі, можна за допомогою змінних маси (Mass Properties), про які ми коротко розповіли в першій главі. В принципі, задаючи функцію густини ρ, в змінних маси можуть бути обчислені будь які інтеграли $\int_\Omega \rho \, d\Omega$.

2.3. Графічні ілюстрації до розв'язків прикладних задач.

Приклади попередніх параграфів показують, що гілка **Results** містить вузли, які використовуються для обробки і зображення результатів. В цьому параграфі ми ще не будемо займатися обчисленнями, а продовжимо знайомство з графічними можливостями системи на задачах, розв'язки яких відомі.

2.3.1. Центр ваги тіла.

Нехай $\rho = \rho(x, y)$ представляє густину матеріалу двовимірного тіла (масу одиниці площі в точці *x*, *y*). Тоді маса тіла, що займає на площині область D, обчислюється за формулою

$$M = \iint_D \rho(x, y)\, d\,x\, d\,y \qquad (1)$$

Положення (x_c, y_c) центра ваги тіла визначається за формулами

$$x_c = \frac{1}{M}\iint_D x\,\rho(x, y)\, d\,x\, d\,y,\ \ y_c = \frac{1}{M}\iint_D y\,\rho(x, y)\, d\,x\, d\,y \qquad (2)$$

Коли говорять про центр ваги геометричної фігури, то мають на увазі, що $\rho(x, y) = 1$.

Для обчислення величин M, x_c, y_c можна застосувати інструменти інтегрування, але простіше використати змінні маси (Mass Properties).

Приклад. Знайти і зобразити центр ваги чверті еліпса $\frac{x^2}{2^2} + \frac{y^2}{1^2} = 1, x \ge 0, y \ge 0$.

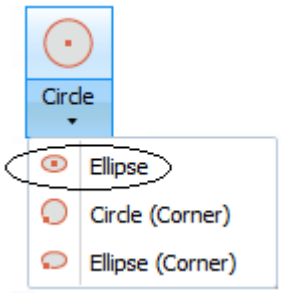
Створіть порожню модель.

File->New->Blank Model

Додайте в модель двовимірний компонент.

Home –> Model –> Add Component – 2D

З'явиться вузол **Component 1**. На стрічці перейдіть на закладку Geometry. На ній розкрийте кнопку-список Circle, виберіть команду Ellipse, і в графічному вікні мишею намалюйте еліпс.



Заповніть ПВ створеного вузла **Component 1 – Geometry 1 – Ellipse 1**:

Size and Shape – **a–semiaxis, b–semiaxis** = 2,1 (*два поля*)

Size and Shape – **Sector angle** = 90

Position – **Base, x, y** = Center,0,0 (*три поля*)

LMB > Build All Objects

Побудуйте СкЕ сітку.

Component 1 - Mesh 1 : Mesh Settings – **Element size** = Extra fine

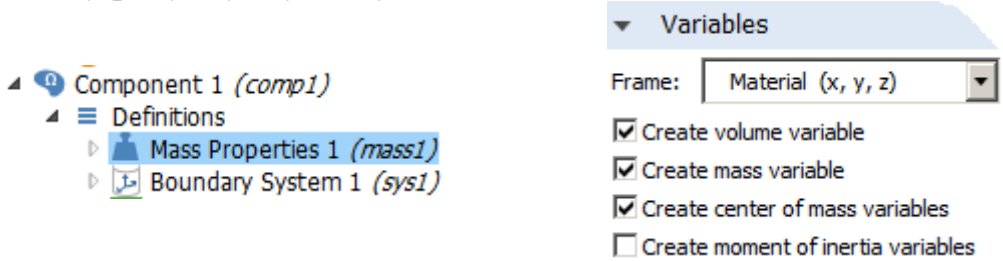
LMB > Build All

Для обчислення положення центру ваги створіть змінні маси. Для цього виконайте команду

Definitions -> **Mass properties** – Mass properties

В гілці **Component – Definitions** з'явиться вузол **Mass Properties 1 (mass1)**, який буде представляти групу механічних і геометричних властивостей компонента. В ПВ створеного вузла в розділі **Source Selection** вказується рівень геометрії і в секції вибору – множина, для якої будуть виконуватися обчислення. Для пошуку маси є розділ, в якому задається густина (за замовчуванням 1). Які характеристики обчислювати і зберігати в змінних моделі вказується відповідними прапорцями, що розташовані внизу ПВ вузла **Mass Properties 1(mass1)**.

Заповніть розділ **Variables** ПВ вузла **Mass Properties 1** так, як зображено на наступному рисунку. Густину залиште одиничною.



Додайте в модель вузол Study, обравши Stationary тип дослідження.

Home **-> Study** – Add Study = Add Study ->

Preset Studies - Stationary **-> x**

В ДМ з'являється вузол **Study 1** і кілька його підвузлів. Виконайте обчислення.

Study 1 : LMB > Compute

Ніяких реальних обчислень в моделі не виконуватиметься, але в гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**, яке буде «містити в собі» обчислені змінні маси.

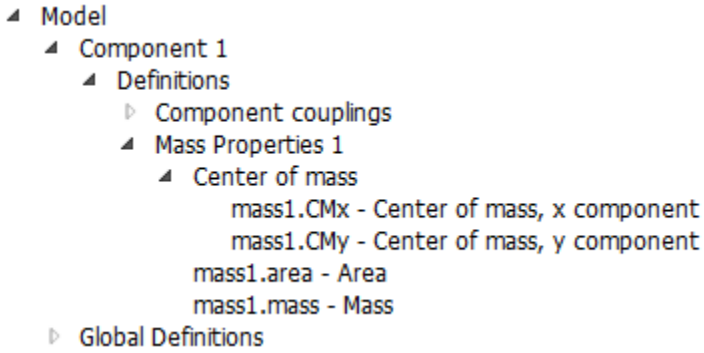
Надрукуйте координати центра ваги.

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

Derived Values – Global Evaluation 1:

Data – **Data set** = Study 1/Solution 1

Ім'я змінної, що представляє абсцису центра ваги, можна знайти в списку готових величин, який відкривається кнопкою Replace Expression, розташованою над полем **Expression**. Розкрийте його.



Вибрати змінну можна подвійним клацанням мишею по відповідному рядку цього списку. Отже введіть

Expression – **Expression** = mass1.CMx

LMB > Evaluate **->** New Table

Буде створена таблиця і в її комірці з'явиться результат обчислення 0.84883.

Ще раз виконайте послідовність інструкцій.

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

Derived Values – Global Evaluation 2:

Data – **Data set** = Study 1/Solution 1

Expression – **Expression** = mass1.CMy

LMB > Evaluate **->** Table - Global Evaluation 1

У першій таблиці в новій комірці з'явиться значення 0.42441.

Порівняйте отримані координати центру ваги з точними значеннями:

$$x_c = \frac{8}{3\pi} \approx 0.848826, \; y_c = \frac{4}{3\pi} \approx 0.424413.$$

Зобразіть область і її центр ваги.

Home -> **Results - Add Plot Group** -> 2D Plot Group

Results – 2 D Plot Group 1:

Data– Data set = Study 1/Solution 1

2D Plot Group 1-> Add Plot - Surface

2D Plot Group 1 – Surface 1:

Data– Data set = From parent

Expression – **Expression** = 1

Coloring and Style – **Color table** = GrayScale

– **Color legend** = False

Для зображення точки в графічну групу **2D Plot Group 1** додайте вузол Point Trajectories.

2D Plot Group 1 **-> Add Plot - More Plots** -> Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

Data– Data set = From Parent

Trajectory Data – **Plot data** = Coordinates

– **x-expression** = mass1.CMx

– **y-expression** = mass1.CMy

Зауваження. Якщо в полях **x-expression** і **y-expression** введені сталі значення, то відображається лише точка.

Налаштуйте стиль зображення точки.

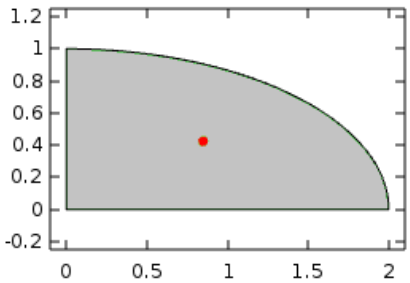
Coloring and Style – **Point style** – **Type** = Point

– **Point style** – **Point radius expression** =0.05

– **Point style** – **Radius scale factor** = True,1

– **Point style** – **Color** = Red

LMB >Plot



Вправа. Обчисліть координати центра ваги, використовуючи формули (1) і (2).

Вказівка. Для величин M, x_c, y_c створіть змінні моделі, а інтеграли в формулах (1), (2) запишіть у вигляді повторних інтегралів

$$M = \iint_D dx\,dy = \int_0^2 \left(\int_0^{\sqrt{1-x^2/4}} dy \right) dx,$$

$$M_x = \iint_D x \cdot dx\,dy = \int_0^2 \left(\int_0^{\sqrt{1-x^2/4}} x \cdot dy \right) dx,$$

$$M_y = \iint_D y \cdot dx\,dy = \int_0^2 \left(\int_0^{\sqrt{1-x^2/4}} y \cdot dy \right) dx.$$

Variables

Name	Expression	Unit
M	integrate(integrate(1,y,0,sqrt(1-x^2/4)),x,0,2)[m^2]	m²
Mx	integrate(integrate(x,y,0,sqrt(1-x^2/4)),x,0,2)	m³
My	integrate(integrate(y,y,0,sqrt(1-x^2/4)),x,0,2)	m³
xc	Mx/M	m
yc	My/M	m

2.3.2. Дія зосередженої сили на пружну півплощину.

Розглянемо плоске пружне середовище, обмежене зверху горизонтальною прямою. В деякій точці на її межі прикладена вертикально діюча вниз зосереджена сила P. Вісь X направимо вправо вздовж границі, а вісь Y - з точки прикладання сили вертикально вгору. Середовище розташовано в півплощині $y<0$. Вертикальне напруження Yy в півплощині обчислюються за формулою [8]

$$Y_y = \frac{2P}{\pi} \frac{y^3}{\left(x^2+y^2\right)^2}. \qquad (1)$$

Особливість цієї задачі полягає в тому, що напруження Y_y в точці прикладання сили дорівнює мінус нескінченності і, щоб апруження відобразити в кольорі, діапазон $(-\infty, 0]$ значень функції Y_y необхідно обмежити знизу.

Нагадаємо, що в Comsol Multiphysics можна розв'язувати задачі з параметрами. Для цього в дереві моделі створюється вузол **Parameters**. Параметри є ідентифікаторами, які можна використовувати в будь-яких місцях моделі, де можуть стояти числові значення. У нас параметрами будуть величина точкового навантаження P і константи «відсікання».

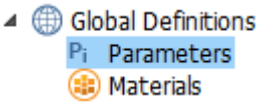
Приклад. Створіть порожню модель.

File->New->Blank Model

Задайте параметри моделі.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Parameters 1** в колонці **Name** вводьте імена параметрів, а в колонці **Expression** – значення. Поля в стовпці **Value** будуть заповнюватися автоматично. Заповніть таблицю параметрів так, як показано нижче.



Parameters

Name	Expression	Value	Description
P	1	1	
Yymin	-1.6	-1.6	
Srmin	-3	-3	
SYymin	-4	-4	

Тут *P* являє величину зосередженої сили, а `Yymin`, `Srmin`, `SYymin` представлятимуть константи відсікання, які будуть використані при побудові графіків.

Створіть функцію, яка обчислюється за формулою (1).

Global Definitions : RMB > Functions – Analytic

Function name = `Yy`

Definition – Expression = `2*P/pi*(y^3/(x^2+y^2)^2)`

Definition – Arguments = `x,y`

Створіть двовимірне джерело даних.

`Results` -> Data Set - More Data Sets -> Grid 2D

Data Sets – Grid 2D 1:

Data – Source, Function = `Function, All` (*два поля*)

В розділі **Parameter Bounds** в поля **Minimum** і **Maximum** введіть діапазони зміни параметрів $-1 \le x \le 1$, $-2 \le y \le 0$.

Почніть побудову графіка функції `Yy`.

`Home` -> **Results** - Add Plot Group ->2D Plot Group

Results – 2 D Plot Group 1:

Data– Data set = `Grid 2D 1`

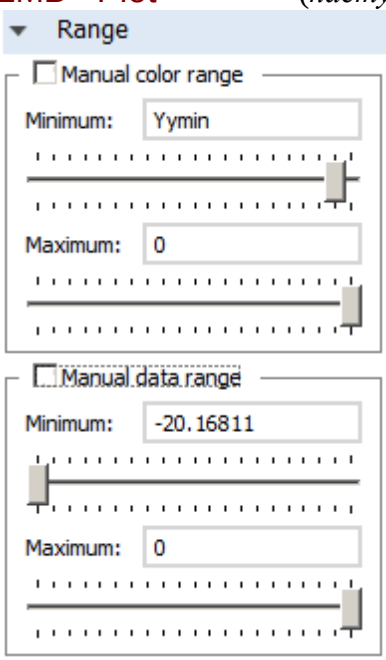
`2D Plot Group 1` -> **Add Plot** - Surface

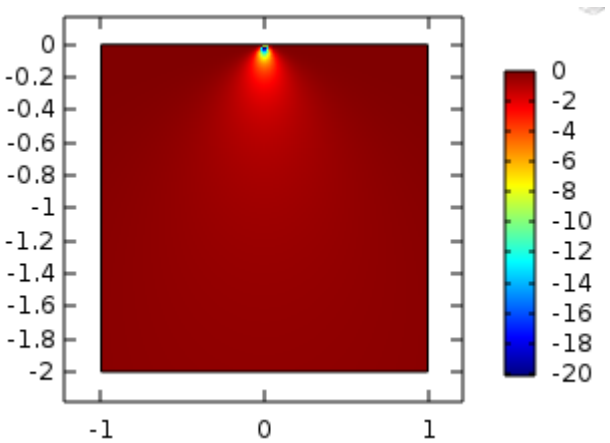
2D Plot Group 1 – Surface 1:

Data – Data set = `From Parent`

Expression–Expression = `Yy(x,y)`

LMB >Plot (*наступний рисунок праворуч*)





Отриманий графік не є дуже гарним, бо його суттєва частина «схована» в маленькій зоні біля точки прикладання сили, і відповідає великим від'ємним (стискаючим) напруженням. Це демонструє шкала кольорів праворуч від графіка. До того ж, математична модель теорії пружності, яка використана при виведенні формули (1), непридатна в околі зони прикладення зосередженого навантаження. Щоб побудувати зрозуміліше зображення в ПВ графіків в розділі **Range** (див. попередній рисунок ліворуч) можна задати діапазони зміни виразу (мінімальне і максимальне значення), що визначають кольори графіка. Експериментально ми визначили, що задовільним «відсікаючим знизу» значенням напруження `Yy` є `Ymin=-1.6`. Задайте його в полях **Minimum** обох секцій **`Manual color range`** і **`Manual data range`**.

2D Plot Group 1 – Surface 1:

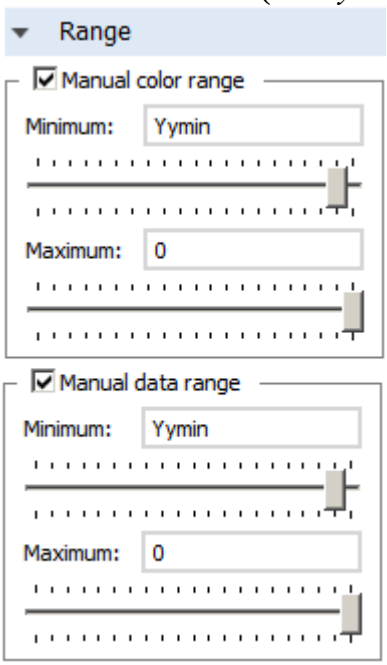
`Range` – **Manual color range** = `True`

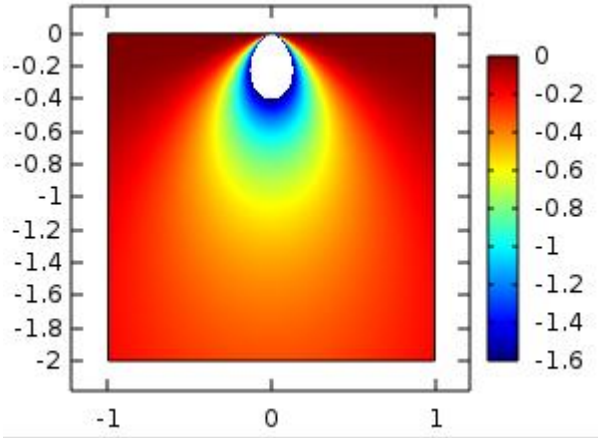
– **Manual color range** – **Minimum** = `Ymin`

– **Manual data range** = `True`

– **Manual data range** – **Minimum** = `Ymin`

LMB >Plot (*наступний рисунок праворуч*)





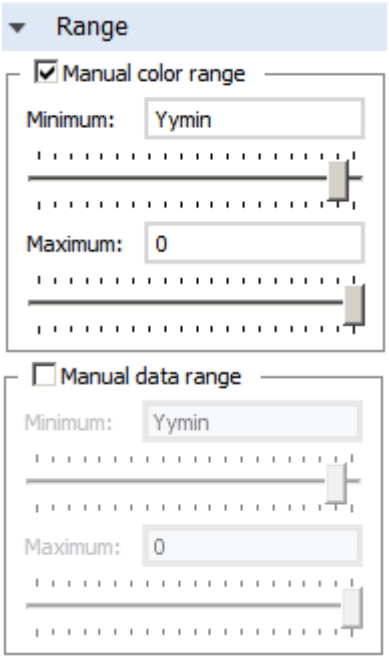
Встановлений прапорець **Manual data range** наказав системі не відображати на графіку зону, де напруження `Yy` менші за `Ymin`.

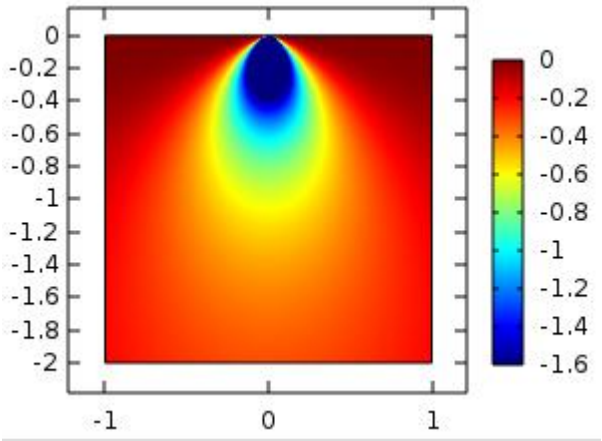
Тепер зніміть цей прапорець.

2D Plot Group 1 – Surface 1:

`Range` – **Manual data range** = `False`

LMB >Plot (*наступний рисунок праворуч*)





В цьому випадку темно синя зона під точкою прикладання навантаження відповідає великим від'ємним (стискаючим) напруженням (меншим за `Yymin`), а коричневий колір на поверхні вдалині від цієї точки відповідає близьким до нуля напруженням.

Для відображення ліній сталого значення в графічну групу **2D Plot Group 1** додайте вузол `Contour`, в якому лінії рівня відобразіть в тому ж діапазоні $Yy_{\min} \le Yy \le 0$.

`2D Plot Group 1` **`-> Add Plot -`** `Contour`

2D Plot Group 1 – Contour 1:

`Data`– Data set = `From parent`

`Expression`–Expression = `Yy(x,y)`

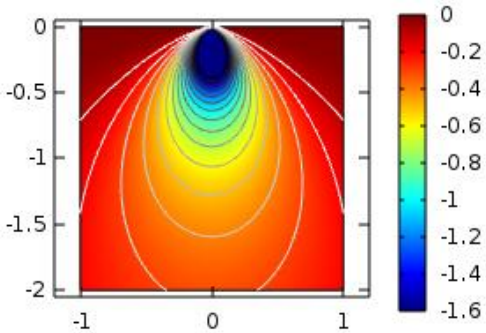
`Levels` –**Entry method** =`Levels`

`Levels` –**Levels** =`range(-0.1,-0.1,Yymin)`

`Coloring and Style` – **Color table** =`GrayScale`

`Coloring and Style` – **Color legend** = `False`

LMB >Plot



Відомо [7], що будь-який елемент середовища, розташований на відстані *r* від точки прикладання зосередженої сили, піддається простому стисканню в радіальному напрямку (від точки прикладання зусилля). При цьому компоненти радіального напруження визначаються за формулами

$\sigma_r = \frac{2P}{\pi}\frac{\cos\theta}{r}, \sigma_\theta = \tau_{r\theta} = 0$, де θ - кут, утворений радіус-вектором точки з вертикаллю. Виберемо окружність довільного діаметру *d*, яка дотична до поверхні півплощини, з центром на вертикальній осі. Для будь-якої точки цього кола маємо $d\cos\theta = r$. Звідси, відповідно до виразу для напруження σ_r одержуємо $\sigma_r = \frac{2P}{\pi d}$, тобто напруження σ_r у всіх точках кола буде однаковим (виключаючи точку прикладання сили). Проілюструємо цей факт графічно. Для цього зобразимо напруження

$$\sigma_r(x,y) = \frac{2P}{\pi}\frac{\cos\theta}{r} = \frac{2Py}{\pi\left(x^2+y^2\right)} \tag{2}$$

у вигляді контурного графіка.

Створіть функцію *Sr*, яка обчислюється за формулою (2).

Global Definitions : RMB > Functions – Analytic

Function name = `Sr`

Definition – Expression = `2*P/pi*(y/(x^2+y^2))`

Definition – Arguments = `x,y`

В гілці **Results** створіть графічну групу `2D Plot Group` і в ній вузол типу `Contour`.

`Home` **-> Results -** Add Plot Group ->2D Plot Group

2D Plot Group 2:

Data– Data set = `Grid 2D 1`

`2D Plot Group 2` **-> Add Plot -** Contour

2D Plot Group 2 – Contour 1:

Data– Data set = `From parent`

Expression – Expression = `Sr(x,y)`

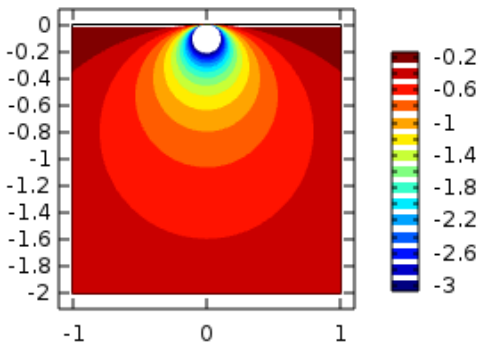
Levels –Entry method =`Levels`

Levels –Levels = `range(Srmin,0.2,0)`

Coloring and Style – Contour type =`Filled`

– Fill surfaces outside of contour levels =`False`

LMB >Plot



Як бачимо, лінії рівня функції $S_r(x,y)$ (напруження σ_r) є окружностями. Біле коло («дірка» на графіку) під точкою прикладання зусилля зображає зону відсікання, в якій $\sigma_r \le Sr_{\min} = -3$.

Продемонструємо дію кількох зосереджених сил на пружну півплощину. Оскільки рівняння теорії пружності лінійні, то напруження, які виникають від дії кількох сил, будуть сумою напружень (1), що виникають від дії кожної з них.

Нехай точки прикладання трьох одиничних зосереджених сил мають координати $x = -0.5,\ 0.0,\ 0.5$. Тоді сумарне вертикальне напруження буде обчислюватися за формулою

$$SY_y(x,y) = Y_y(x,y) + Y_y(x-0.5,y) + Y_y(x+0.5,y).$$

В гілці **Results** створіть нову графічну групу `2D Plot Group` і в ній вузол типу `Contour`.

`Home` **`-> Results -`** `Add Plot Group ->2D Plot Group`

2D Plot Group 3:

`Data`– **Data set** = `Grid 2D 1`

`2D Plot Group 3` **`-> Add Plot -`** `Contour`

2D Plot Group 3 – Contour 1:

`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `Yy(x,y)+Yy(x-0.5,y)+Yy(x+0.5,y)`

`Levels` –**Entry method** =`Levels`

`Levels` –**Levels** = `range(SYymin,0.2,0)`

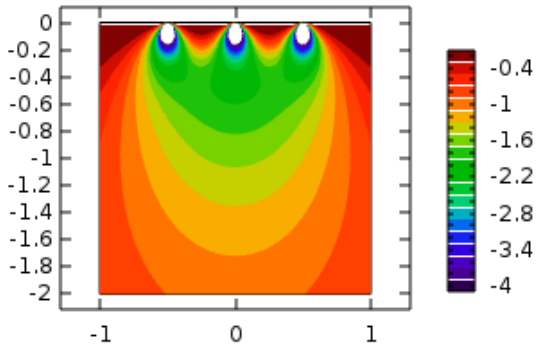
`Coloring and Style` – **Contour type** =`Filled`

– **Fill surfaces outside of contour levels** =`False`

– **Color table** = `Spectrum`

– **Color legend** = `True`

LMB >Plot



«Дірки» на графіку під точками прикладання зусиль зображають зони відсікання, в яких $SY_y(x,y) \le SYy_{\min}$.

2.3.3. Напруження в нескінченній платівці з круглим отвором.

Розглянемо задачу візуалізації напружень в нескінченній плоскій пластині з отвором, на яку в горизонтальному напрямку діє однорідне розтягувальне зусилля P (розраховане на одиницю довжини, прикладене на великій відстані від отвору). Введемо полярну систему координат, початок якої розташуємо в

центрі отвору радіуса a, а полярну вісь направимо горизонтально. Відомо [7], що радіальні напруження в такій платівці визначаються за формулою

$$\sigma_r = \frac{P}{2}\left(1-\frac{a^2}{r^2}\right)+\frac{P}{2}\left(1+\frac{3a^4}{r^4}-\frac{4a^2}{r^2}\right)\cos 2\theta, \qquad (1)$$

де r і θ - полярні координати..

Приклад. Створіть порожню модель.

File->New->Blank Model

Задайте параметри моделі.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Parameters** в таблиці в колонці **Name** введіть ім'я параметра `P`, а в колонці **Expression** введіть одиницю. Аналогічно задайте значення параметра a =1.

Створіть функцію $S_r(r,\theta)$, яка обчислюється за формулою (1).

Global Definitions : RMB > Functions – Analytic

Function name = `Sr`

Definition – Expression =

`P/2*(1-a^2/r^2)+P/2*(1+3*a^4/r^4-4*a^2/r^2)*cos(2*v)`

Definition – Arguments = `r,v`

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Геометрія компонента складатиметься з квадрата і кругового отвору.

`Geometry` **-> Primitives** –> Square

Component 1 - Geometry 1 – Square 1:

Size – Size length = 8

Position – Base, x, y = `Corner,-4,-4` (*три поля*)

Тут в розділі **Size** задається довжина сторони квадрата, а в розділі **Position** вводяться координати його вершини.

Накресліть коло.

`Geometry` **-> Primitives** – Circle

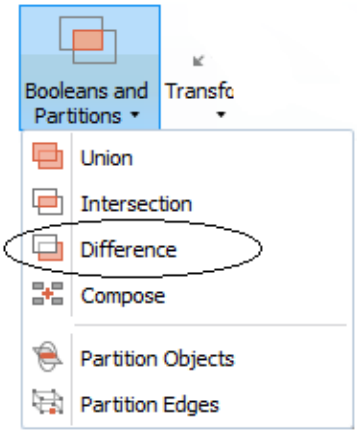
Component 1 - Geometry 1 - Circle 1:

Size and Shape – Radius: = `a`

Size and Shape – Sector angle = `360`

Position – Base, x, y = `Center,0,0` (*три поля*)

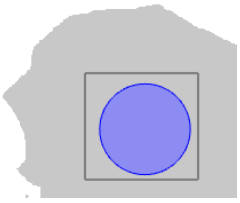
На закладці `Geometry` клацніть по кнопці-списку Boolean and Partitions і в випадному списку виберіть команду Difference.



В дереві моделі буде створено вузол **Geometry - Difference 1**. Він буде визначати логічну операцію віднімання (difference) над створеними областями квадрата і кола. В ПВ цього вузла в розділі **`Difference - Object to add`** активуйте кнопку Active. Вона повинна змінити свій вигляд.

У графічному вікні клацніть по квадрату, і в секцію **`Object to add`** додасться ім'я квадратної області `sq1`. Потім в секції **`Difference - Object to substruct`** активуйте кнопку Active. В графічному вікні спробуйте виділити коло. Якщо вам це не вдається, то використайте кнопку Select Box, розташовану над графічним вікном

Клацнувши по ній, виділіть мишею в графічному вікні область, яка містить коло,

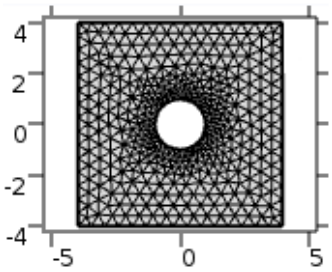


і його ім'я `c1` додасться в секцію **`Object to substruct`**.

LMB > Build All Objects

В графічній зоні з'явиться зображення квадрата з вирізаною з нього областю кола. На створеній геометрії побудуйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All



Функції потрапляють в робочій простір джерела даних Study/Solution і стають доступними в гілці обробки даних **Results** тільки після виконання яких-небудь обчислень. Зазвичай в моделі такі обчислення виконуються. В нашому прикладі ми повинні подбати про це окремо, створивши вузол типу Study.

Untitled.mph (root): RMB > Add Study = Add Study ->

Preset Studies - Stationary -> **x**

В результаті в ДМ з'являється гілка **Study 1** і кілька її підвузлів. Натисніть клавішу F8, виконавши команду Compute. Ніяких реальних обчислень в моделі не виконуватиметься, але в гілці **Results - Data Sets** буде створено вузол (джерело даних) **Study 1/Solution 1**. Він містить інформацію про геометрію області, а також про глобальну функцію Sr(x,y).

Тепер в гілку **Results** додайте гілку 2D Plot Group і створіть в ній вузол Surface.

Home **-> Results** - Add Plot Group ->2D Plot Group

2D Plot Group 1:

Data– **Data set** = Sudy 1/Solution 1

2D Plot Group 1 **-> Add Plot** - Surface

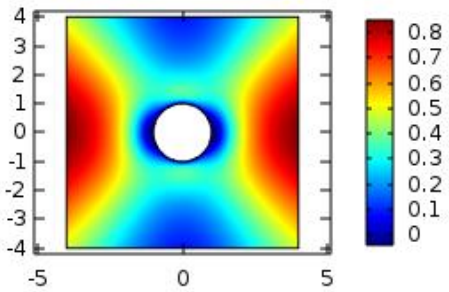
2D Plot Group 1 – Surface 1:

Data– **Data set** = From parent

Expression – **Expression** = Sr(sqrt(x^2+y^2),atan2(y,x))

Вираз в полі **Expression** відповідає функції радіальних напружень (1), яка переписана в декартових координатах. Використана тут функція atan2(y,x) повертає значення полярного кута φ точки (*x*,*y*), який належить діапазону $-\pi \le \varphi \le \pi$, і враховує знаки аргументів *x* і *y*.

LMB >Plot



Палітра кольорів, яка наведена праворуч, показує, що мінімальне значення радіального напруження σ_r близьке до нуля, а максимальне – більше за 0.8. Перевіримо це.

Results**->Derived Values**-More Derived Values**->** Surface Minimum

Derived Values – Surface Minimum 1:

Data – **Data set** = Study 1/Solution 1

Selection – **Selection** =Manual,1

Expression – **Expression** = Sr(sqrt(x^2+y^2),atan2(y,x))

LMB > Evaluate **->** New Table

В новій таблиці з'явиться результат обчислення мінімального значення введеного виразу `-0.041645`. Воно отримано по області квадрата з круговим отвором.

`Results->Derived Values-More Derived Values->Surface Maximum`

Derived Values – Surface Maximum 1:

`Data` – Data set = `Study 1/Solution 1`

`Selection` – Selection = `Manual,1`

`Expression` – Expression = `Sr(sqrt(x^2+y^2),atan2(y,x))`

LMB > Evaluate -> Table 1 – Surface Minimum 1

В новій комірці попередньої таблиці з'явиться максимальне значення введеного виразу: `0.84950`.

Побудуємо графіки залежності напруження σ_r вздовж перерізів області, які співпадають з координатними осями. Для цього в гілці **Data Sets** створіть нові джерелі даних `Cut Line 2D` (лінія перерізу двовимірної області).

Data Sets : RMB > Cut Line 2D

Data Sets - Cut Line 2D 1 :

`Data` – Data set = `Study 1/Solution 1`

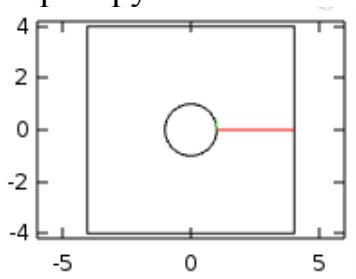
`Line Data` – Line entry method = `Two points` (*переріз по двом точкам*)

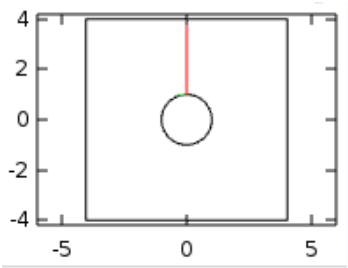
– **Point 1 : x, y**= `1,0` (*два поля*)

– **Point 2 : x,y**= `4,0` (*два поля*)

Цим було задано координати точок, через які проходить переріз.

Виконайте команду Plot, і отримайте графік лінії перерізу, тобто джерела даних **Cut Line 2D 1**, яке представлено червоним відрізком на наступному рисунку ліворуч. Аналогічно, по точкам (0,1) і (0,4) будується джерело даних **Cut Line 2D 2**, показане праворуч.





Тепер побудуйте графіки напруження σ_r вздовж цих перерізів.

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

1D Plot Group 2: **`Data Set` – Data** = `Cut Line 2D 1`

1D Plot Group 2 : RMB > Line Graph

1D Plot Group 2 - Line Graph 1:

`y-Axis Data` – Expression = `Sr(sqrt(x^2+y^2),atan2(y,x))`

`x-Axis Data` – Parameter, Expression = `Expression,x` (*два поля*)

Задайте позначку біля кривої (легенду).

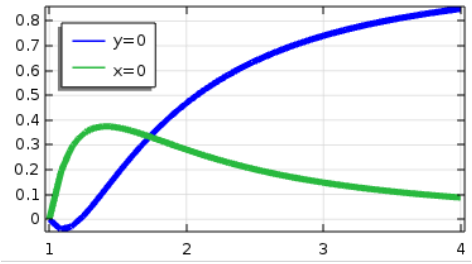
`Legends` – Show Legends, Legends = `True,Manual` (*два поля*)

`Legends` –Legends = `'y=0'` (*легенда, ведення в таблиці*)

1D Plot Group 2 : RMB > Line Graph

1D Plot Group 2 - Line Graph 2:

Data – **Dataset** = `Cut Line 2D 2`
y-Axis Data – **Expression** = `Sr(sqrt(x^2+y^2),atan2(y,x))`
x-Axis Data – **Parameter, Expression** = `Expression,y` (*два поля*)
Legends – **Show Legends, Legends** = `True,Manual` (*два поля*)
Legends – **Legends** = `'x=0'` (*легенда, ведення в таблиці*)
LMB > Plot

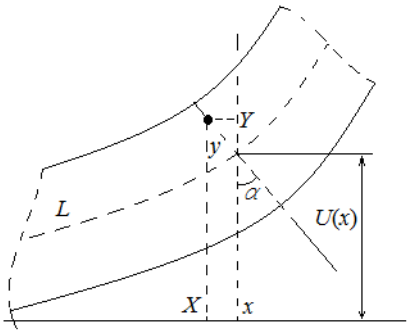


Нагадаємо, що розтяг площини виконується в горизонтальному напрямку вдалині від отвору.

2.3.4. Прогин балки.

Рівняння рівноваги балки, що знаходиться під дією зовнішніх сил, має вигляд $EJ\,u''''_{xxxx} = q(x)$, де q - зовнішнє зусилля, розраховане на одиницю довжини балки, E і J - модуль пружності матеріалу балки і момент інерції її перерізу. Ось X збігається з віссю балки. Функція $u(x)$ являє вертикальне зміщення точок осі балки, які мали до навантаження горизонтальну координату x. Для зосереджених навантажень функція $q(x)$ визначається виразом $q(x) = \sum_{i=1}^{n-1} f_i \cdot \delta(x - x_i)$, де $\delta(x)$ - дельта-функція Дірака, x_i - точки прикладання навантажень, f_i - величини навантажень. Для вільно опертої на краях балки граничні умови мають вигляд $u(0)=0, u(L)=0$, $u''(0)=0, u''(L)=0$, де L - довжина балки.

Відповідно до гіпотези плоских перерізів, всі поперечні перерізи, які до деформування були перпендикулярні осі балки, залишаються перпендикулярними до неї після деформування. Точки, що мали до деформування координати (x, y), переходять в точки з координатами (X,Y), де $X = x - y\sin\alpha$, $Y = u(x) + y\cos\alpha$ (див. рисунок).



Оскільки $tg\,\alpha = u'(x)$, то

$$X = x - \frac{u'_x(x)y}{\sqrt{1+(u'_x(x))^2}} \quad \text{і} \quad Y = u(x) + \frac{y}{\sqrt{1+(u'_x(x))^2}}.$$

Зміщення точок балки визначається вектором

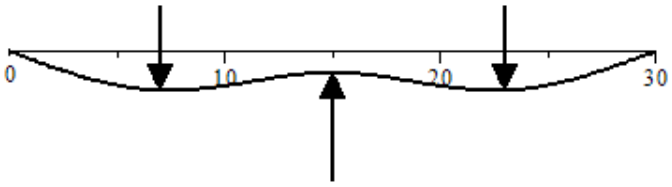
$$(\Delta x, \Delta y) = (X - x, Y - y) = \left(-\frac{u'_x(x)y}{\sqrt{1+(u'_x(x))^2}}, u(x) + \frac{y}{\sqrt{1+(u'_x(x))^2}} - y \right). \quad (1)$$

Якщо поперечний переріз балки прямокутний, то подовжні (розтягувальні) напруження визначаються за формулою

$$\sigma_x(x, y) = \frac{M(x) \cdot y}{J}, \quad (2)$$

де $M = -EJ\,u''(x)$ – момент інерції в перерізі x балки, y - вертикальна координата точки балки (y=0 відповідає осі). Величину $\sigma_x(x, y)$ ми будемо використовувати для функціонального забарвлення поверхні балки.

Нехай балка довжиною `L=30` і висотою `h=6` вільно оперта по краях, знаходиться під дією трьох зосереджених навантаг. Розв'яжемо модельну задачу, в якій покладемо E = 2, J = 54. Схема прикладання навантаг $f_i = -4, 5, -4$ в точках $x_i = 7, 15, 23$ (i=1,2,3) показана на наступному рисунку.



Функція $u(x)$ прогину точок осі балки для цієї задачі має вигляд

$$u(x) = \frac{-4|x-23|^3 + 5|x-15|^3 - 4|x-7|^3 + 135x^2 - 4050x + 33165}{1296}. \quad (3)$$

Її перша похідна дорівнює

$$u'(x) = \frac{-4|x-23|(x-23) + 5|x-15|(x-15) - 4|x-7|(x-7) + 90x + 1350}{432}. \quad (4)$$

Це випливає з співвідношення $\frac{d}{dx}\left(|x|^3\right) = 3|x| \cdot x$.

Згинальний момент в перерізі x дорівнює

$$M = -EJ\,u''(x) = -E\,J\,\frac{-4|x-23| + 5|x-15| - 4|x-7| + 45}{216}. \quad (5)$$

Вираз (5) може бути одержаний з (3) подвійним диференціюванням, якщо врахувати, що $\frac{d^2}{dx^2}\left(|x|^3\right) = 6|x|$.

Приклад. Створіть порожню модель.

File->New->Blank Model

Задайте параметри моделі.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions-Parameters** в таблиці задайте значення параметрів E=2 і J=54.

Створіть функцію $M(x)$, яка обчислюється за формулою (5).

Global Definitions : RMB > Functions – Analytic

Function name = M

Definition – Expression =
-E*J*(-4*abs(x-23)+5*abs(x-15)-4*abs(x-7)+45)/216

Definition – Arguments = x

Для обчислення по формулі (3) створіть функцію u(x).

Global Definitions : RMB > Functions – Analytic

Function name = u

Definition – Expression = (-4*abs(x-23)^3+5*abs(x-15)^3-4*abs(x-7)^3+135*x^2-4050*x+33165)/1296

Definition – Arguments = x

Для обчислення похідної за формулою (4) створіть функцію u1(x) .

Global Definitions : RMB > Functions – Analytic

Function name = u1

Definition – Expression = (-4*abs(x-23)*(x-23)+5*abs(x-15)*(x-15)-4*abs(x-7)*(x-7)+90*x-1350)/432

Definition – Arguments = x

Для обчислення напруження σ_x по формулі (2) створіть функцію Tx(x,y).

Global Definitions : RMB > Functions – Analytic

Function name = Tx

Definition – Expression = M(x)*y/J

Definition – Arguments = x,y

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Його геометрія складатиметься з одного прямокутника.

Geometry **-> Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

Size – Width, Height = 30,6 (*два поля*)

Position – Base, x, y = Corner,0,-3 (*три поля*)

LMB > Build All Objects

Побудуйте в області чотирикутну сітку.

Mesh **-> Generators** – Mapped

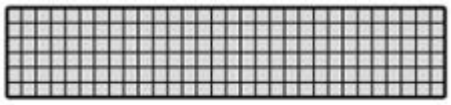
Задайте розміри клітинок сітки.

Mesh 1 – Size:

Element Size – Custom= True

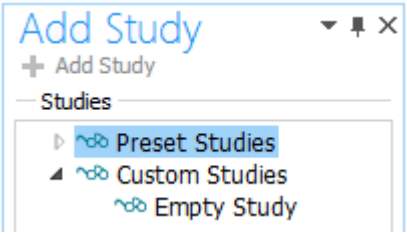
Element Size Parameters – Maximum element size= 1

LMB > Build All



Тепер нам знадобиться джерело даних, яке в своєму робочому просторі буде містити створені нами функції і геометрію області разом з сіткою. Їм буде джерело Study/Solution, якщо ми додамо в модель вузол типу Study.

Untitled.mph (root): RMB > Add Study = Add Study ->
Custom Studies - Empty Study -> **x**



У створеному вузлі **Study 1** виконайте команду = Compute. Ніяких реальних обчислень в моделі не виконуватиметься, але в гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**.

В гілку **Results** додайте гілку 2D Plot Group і створіть в ній вузол Surface.

Home -> **Results** - Add Plot Group ->2D Plot Group

2D Plot Group 1:

Data– **Data set** = Sudy 1/Solution 1

Plot Settings – **Plot data set edges** = False

2D Plot Group 1 -> **Add Plot** - Surface

2D Plot Group 1 – 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 1 – Surface 1:> RMB > Deformation

2D Plot Group 1 – Surface 1 – Deformation 1 :

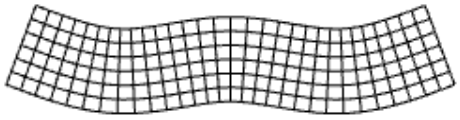
Expression – **x component** = -u1(x)/sqrt(1+u1(x)^2)*y

– **y component** = u(x)+y/sqrt(1+u1(x)^2)-y

Введені тут вирази відповідають вектору зсуву, що визначається за формулою (1).

LMB > Plot

В результаті графічна зона вузла **Surface 1** буде містити деформоване каркасне зображення балки.



В графічній грцпі **2D Plot Group 1** створіть ще один вузол Surface, який ми використаємо для функціонального забарвлення зони балки.

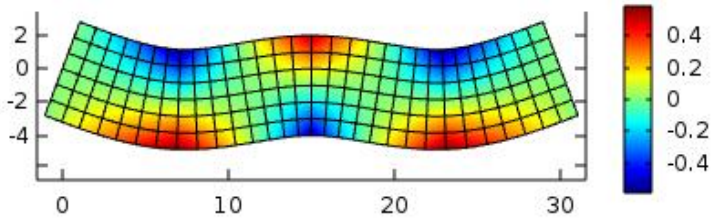
2D Plot Group 1 – Surface 1:> RMB > Duplicate

2D Plot Group 1 – Surface 2:

Expression – **Expression** = Tx(x,y)

Coloring and Style – **Coloring, Color** =Color table, Rainbow
– **Color legend** = True
– **Wireframe** = False

LMB > Plot



На останньому рисунку деформоване зображення балки з функціональним забарвленням, побудованим у вузлі **2D Plot Group 1 - Surface 2**, поєднане з каркасним зображенням, побудованим у вузлі **2D Plot Group 1 - Surface** 1.

Зауважимо, що опція **Coloring and Style**–**Wireframe** вузла 2**D Plot Group 1-Surface 1** працездатна тільки, якщо джерелом даних цього вузла є Sudy/Solution. Якщо СкЕ сітку будувати не потрібно, то графічне зображення зігнутої балки з функціональним забарвленням можна побудувати простіше. Для цього можна використати джерела даних Grid 2D або Grid 3D.

Створимо джерело даних Grid 3D, яке має розміри тривимірної балки.

Results –> Data Set - More Data Sets –> Grid 3D

Data Sets – Grid 3D 1:

Data – **Source, Function** = Function, All (*два поля*)

В розділі **Parameter Bounds** в полях **Minimum** і **Maximum** введіть діапазони зміни параметрів $0 \le x \le 30$, $-3 \le y \le 3$, $-1 \le z \le 1$.

Створіть групу тривимірних графіків і в ній вузол Surface.

Results –> Plot Group - 3D Plot Group

3D Plot Group 2 :

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 2 - Surface 1 :

Data – **Data set** = From parent

Expression – **Expression** = Tx(x,y)

Coloring and Style – **Coloring, Color** =Color table, Rainbow

– **Color legens** = True

У вузлі **Surface 1** створіть підвузол деформації.

2D Plot Group 2 – Surface 1: RMB > Deformation

2D Plot Group 2 – Surface 1 – Deformation 1 :

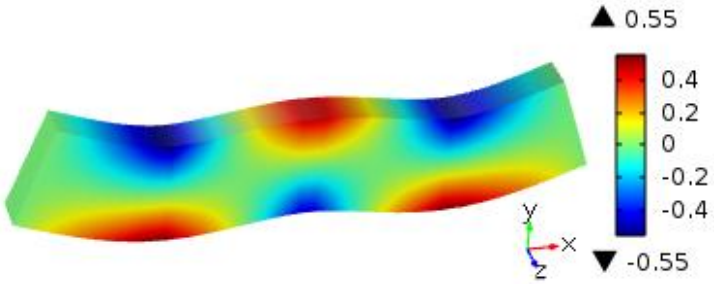
Expression – **x component** = -u1(x)/sqrt(1+u1(x)^2)*y

– **y component** = u(x)+y/sqrt(1+u1(x)^2)-y

– **z component** = 0

Scale – **Scale factor** = 1

LMB > Plot



■

Система COMSOL Multiphysics вміє будувати анімаційні зображення. Для цього з випадного меню вузла **Results - Export** вибирається команда Player, що створює вузол `Animation`. Також можна записувати анімацію в файл. Для цього в гілку **Results - Export** слід додати вузол `Animation` Animation . Він має схожі з «плеєром» налаштування, а також поля для введення формату і імені відеофайлу.

Продемонструємо можливості системи по створенню анімованих зображень на прикладах.

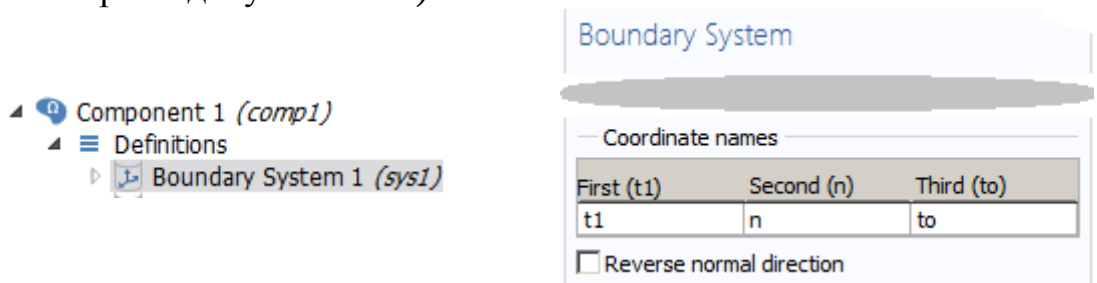
2.3.5. Дотична і нормаль до еліпса.

Одиничний дотичний вектор до плоскої кривої, заданої параметрично $\boldsymbol{r}(\tau)=(x(\tau),y(\tau))$, в точці $\tau=t$ дорівнює $\boldsymbol{L}=(L_x,L_y)=\boldsymbol{r}'(t)/\|\boldsymbol{r}'(t)\|$, де символи $\|\ \|$ позначають довжину вектора. Одиничний вектор нормалі в точці $\tau=t$ дорівнює $\mathbf{N}=(L_y,-L_x)$. Тоді векторні рівняння дотичної і нормальної прямих в тій же точці матимуть вигляд

$$\boldsymbol{r}_t(\tau)=\boldsymbol{r}(t)+\boldsymbol{L}(t)\cdot\tau,\ \boldsymbol{r}_n(t)=\boldsymbol{r}(t)+\boldsymbol{N}(t)\cdot\tau, \qquad (1)$$

де τ буде натуральним параметром уздовж цих прямих.

Якщо крива створена в геометрії 2D моделі, то одиничні дотичні і нормальні вектори до неї «будуються» у вузлі **Boundary System 1 (sys1)**, і позначаються за замовчуванням як `t1`, `n`, `to` (дотичний і нормальний вектори, а також їх векторний добуток $\mathbf{n}\times\mathbf{t1}$).



У виразах, які обчислюються на такій кривій, можна застосовувати координати цих векторів `t1x,t1y,nx,ny`. В двовимірному випадку замість `t1` можна використовувати ідентифікатор `t`, тобто координати дотичного вектора позначати як `tx` і `ty`.

Приклад. Побудуємо графіки дотичної і нормалі до еліпса з півосями a і b. Його параметричне рівняння можна записати у вигляді $x = a\cos\tau$, $y = b\sin\tau$, $0 \le \tau \le 2\pi$.

Створіть шаблон порожньої моделі.

File->New->Blank Model

Задайте параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри a=2 і b=1.

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Геометрія компонента складатиметься з прямокутної розрахункової області і еліпса.

Створіть прямокутник.

Geometry **-> Primitives** – Rectangle

Geometry 1 - Rectangle1:

Size – **Width, Height** = 6,6 (*два поля*)

Position – **Base, x, y** = Center,0,0 (*три поля*)

Побудуйте еліпс.

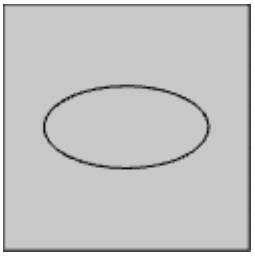
Geometry **-> Primitives** – Ellipse

Geometry 1 – Ellipse 1:

Size and Shape – **a–semiaxis, b–semiaxis** = a,b (*два поля*)

Position – **Base, x, y** = Center,0,0 (*три поля*)

LMB > Build All Objects



Використовуючи вузол **Mesh 1**, побудуйте стандартну сітку

Component 1 (comp1) - Mesh 1 : LMB > Build All

В нашій задачі будемо вважати, що положення точки дотику (x_0, y_0) визначається моментом часу *t*. Нагадаємо, що величини, які можуть залежати від часу, просторових координат або інших величин, називаються змінними (VARIABLES), і їм присвоюються імена (ідентифікатори). Отже x_0, y_0 повинні бути змінними моделі. Координати дотичного (L_x, L_y) і нормального (N_x, N_y) векторів до кривої залежать від точки (x_0, y_0) дотику, а значить, від часу. Тому вони також повинні бути визначені як змінні.

Змінні бувають глобальними і локальними. Локальні змінні створюються в таблиці ПВ вузла **Definitions - Variables** поточного компонента. Перед використанням створених змінних потрібно «розв'язати» модель, виконавши команду Compute або Update Solution.

Створіть вузол локальних змінних.

Definitions **->Variables** – Local Variables

В ПВ побудованого вузла **Variables 1** в таблиці в колонці **Name** введіть ім'я змінної x0, а в колонці **Expression** введіть вираз a*cos(t[1/s]). Зверніть увагу на аргумент t[1/s] функції cos(·). Більшість стандартних математичних функцій очікують безрозмірні аргументи і повертають безрозмірні значення. Але змінна *t* має розмірність часу, наприклад, секунди, якщо в моделі використовується система одиниць СІ. Щоб використати змінну *t* в якості безрозмірного аргументу, ми доповнили її розмірністю [1/s]. Розмірність [s] змінної *t* множиться на розмірність [1/s] і величина t[1/s] стає безрозмірною.

Задайте значення інших змінних y0,Lx,Ly,Nx,Ny так, як показано в таблиці.

Variables

Name	Expression	Unit	Description
x0	a*cos(t[1/s])		
y0	b*sin(t[1/s])		
Lx	at1(x0,y0,t1x)		
Ly	at1(x0,y0,t1y)		
Nx	at1(x0,y0,nx)		
Ny	at1(x0,y0,ny)		

Тут x0,y0 являють координати точки дотику в момент часу t.

```
x0= a*cos(t[1/s])
y0= b*sin(t[1/s])
```

Змінні Lx,Ly представляють координати дотичного вектора в точці x0,y0.

```
Lx= at1(x0,y0,t1x)
Ly= at1(x0,y0,t1y)
```

Зверніть увагу на те, що зоною дії змінних є вся модель (опція за замовчуванням **Geometric entity level** = Entire model). Але оскільки вектор t1 визначений лише на кривих (ми будемо використовувати його на еліпсі), то обчислювати значення потрібно з застосуванням одновимірного оператора at1(x0,y0,...). При чому точка x0,y0 повинна бути розташована на кривій.

Аналогічно, змінні Nx,Ny представляють координати нормального вектора до кривої в точці x0,y0.

```
Nx= at1(x0,y0,nx)
Ny= at1(x0,y0,ny)
```

Додайте в модель вузол обчислення.

Home -> Study - Add Study

В панелі Add Study, яка з'явиться збоку, виберіть Time Dependent тип дослідження, і закрийте панель.

Задайте діапазон і інтервал зміни параметра часу t.

Study 1 - Step 1: Time Dependent:

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

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**. Його «робочий простір» містить інформацію про геометрію моделі, діапазони змінної `t`, та про змінні, які створені у вузлі **Global Definitions - Variables 1**.

Створіть джерело даних типу `Edge 2D`, яке буде репрезентувати еліпс.

`Results` -> Data Set - More Data Sets -> Edge 2D

Data Sets - Edge 2D 1: **`Selection`** – **Selection,** <> = `Manual,5,6,7,8` (*еліпс*)

Створіть джерело даних типу `Parameterized Curve 2D`, яке буде репрезентувати дотичну.

`Results` -> Data Set - More Data Sets -> Parameterized Curve 2D

Data Sets - Parameterized Curve 2D 1:

`Data` – **Data set** = `Study 1/Solution 1`

`Parameter` – **Name, Minimum, Maximum** = `tau,-1, 1` (*три поля*)

`Expressions` – **x, y** = `x0+tau*Lx, y0+tau*Ly` (*два поля*)

Введені вирази відповідають першій формулі (1), бо змінні `x0,y0,Lx,Ly` залежать від змінної часу `t`.

Почнемо будувати зображення. Створіть двовимірну графічну групу.

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

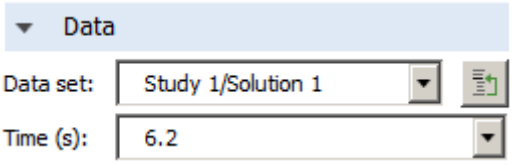
2D Plot Group 1 :

`Data` – **Data set** = `Study 1/Solution 1`

`Data` – **Time (s)** = `6.2`

`Plot Settings` – **Plot data set edges** = `False`

При виборі джерела даних, «що містить» час (у нас це `Study 1/Solution 1`), в розділі **`Data`** з'являється поле-список **Time** з моментом часу, в який будуватимуться графіки.



В створеній групі **2D Plot Group 1** побудуйте графіки еліпса, точки дотику, дотичної прямої та вектора нормалі, який зобразимо стрілкою.

Для зображення еліпса виконайте наступні інструкції.

`2D Plot Group 1`-> Add Plot – Line

Нагадаємо, що в двовимірному випадку вузли типу `Line` використовуються для графічного зображення скалярних функцій уздовж різноманітних ліній.

2D Plot Group 1 - Line 1 :

`Data` – **Data set** = `Edge 2D 1`

`Expression` – **Expression** =`1`

`Coloring and Style` – **Line type** =`Tube`

– **Tube radius expression** =`0.04`

– **Radius scale factor** =`True,1`

– **Coloring, Color** =`Uniform, Red` (*два поля*)

Створіть зображення точки дотику.

2D Plot Group 1 -> Add Plot - More Plots - Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

Data– Data set = From parent

Trajectory Data – Plot data = Coordinates

Trajectory Data – x-expression, y-expression = x0,y0 (*два поля*)

Сталі значення в полях **x-expression**, **y-expression** призводять до побудови зображення точки, а не кривої.

Coloring and Style – Point style –Type = Point

– Point style –Point radius expression = 0.1

– Point style – Radius scale factor = True,1

– Point style – Color = Blue

Зверніть увагу на те, що положення точки визначається координатами x0,y0 в момент часу, який задано в батьківському вузлі (в ПВ графічної групи) в полі **Data – Time (s)**.

Побудуйте дотичну до еліпса.

2D Plot Group 1-> Add Plot – Line

2D Plot Group 1 - Line 2 :

Data – Data set = Parameterized Curve 2D 1

Data– Time (s) = 6.2

Тут введено момент часу, який співпадає з часом батьківського вузла. Тому дотична буде накреслена в точці x0,y0.

Expression – Expression =1

Coloring and Style – Line type =Tube

– Tube radius expression =0.03

– Radius scale factor = True,1

– Coloring, Color =Uniform, Black (*два поля*)

Нормаль до кривої намалюємо у вигляді стрілки (графічне зображення вектора). Для цього в гілку **2D Plot Group 1** додайте вузол типу Arrow Line, який призначений для зображення векторного поля вздовж кривих.

2D Plot Group 1-> Add Plot – Arrow Line

2D Plot Group 1 - Arrow Line 1 :

Data – Data set = Parameterized Curve 2D 1

Стрілки векторного поля плануємо розподіляти вздовж дотичної.

Data– Time (s) = 6.2

Щоб стрілка була зображена в точці x0,y0, потрібно ввести момент часу, який співпадає з часом в батьківському вузлі.

Expression – x component, y component = Nx,Ny (*два поля*)

Coloring and Style – Scale factor = True,1

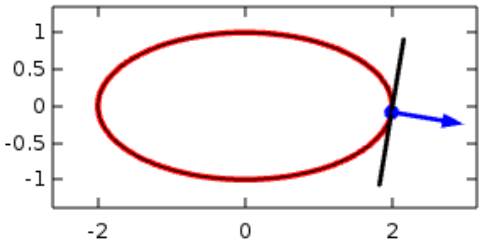
– Placement =Uniform

– Number of arrows =1

– Color =Blue

Опція **Placement**=`Uniform` визначає рівномірне розподілення стрілок векторного поля вздовж одновимірного джерела даних. В полі **Number of arrows** задається кількість стрілок (векторів). Якщо задається одна стрілка (**Number of arrows**=`1`), то вона розташовується «всередині» кривої. В нашому випадку ця середина співпадає з точкою дотику.

Виконайте команду Plot і отримайте наступне зображення.



Воно побудовано для моменту часу `t=6.2`, вказаного одночасно в полях **Time (s)** вузлів **2D Plot Group 1**, **Line 2** та **Arrow Line 1**. Час, вказаний в однойменному полі вузла **Line 1** не має значення.

Тепер побудуйте анімацію руху дотичної і нормалі. Для цього в гілку **Results - Export** командою

`Results` –> Animation –> Player

додайте вузол, який автоматично отримає ім'я **Animation 1**.

Export – Animation 1:

`Scene` – **Subject** = `2D Plot Group 1`

Останнє означає, що анімація буде містити кадри зображень, побудовані в графічній групі **2D Plot Group 1**.

`Frames` – **Frame selection** = `All`

Значення `All` в списку **Frame selection** означає генерування кадрів анімації для всіх моментів часу з діапазону `range(0,0.1,2*pi)`.

`Playing` – **Repeat** = `True`

LMB > Generate Frame .

Почнеться процес генерації кадрів. По завершенні стає доступною панель управління анімацією , яка розташована в лівому верхньому куті графічного вікна. Кнопка Play цієї панелі запускає анімацію, кнопка Stop зупиняє, а кнопки прокручують анімацію «покадрово».

Зауваження. Для визначення точки дотику можна було б використати який-небудь допоміжний параметр, наприклад, криволінійний параметр τ вздовж дуги еліпса. Тоді для побудови анімації потрібно було б застосовувати параметричний аналіз.

2.3.6. Траєкторія руху точки на ободі колеса.

Траєкторія фіксованої точки кола, яке котиться по прямій, називається циклоїдою. Якщо радіус кола дорівнює 1 і швидкість руху центру $v=1$, то параметричне рівняння циклоїди матиме вигляд $x(t)=t-\sin t$, $y(t)=1-\cos t$. Побудуємо анімацію цього руху. Крім рухомої разом з окружністю точки,

будуватимемо її траєкторію (ділянку циклоїди від початку до поточного положення точки).

Приклад. Створіть порожню модель.

File->New->Blank Model

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Геометрія компонента складатиметься з прямокутника.

Geometry **->** **Primitives** – Rectangle

Geometry 1 - Rectangle1:

Size – **Width, Height** = 15,3 (*два поля*)

Position – **Base, x, y** = Corner,-1,0 (*три поля*)

LMB > Build All Objects

Додайте в модель вузол обчислення.

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(0,0.25,4*pi)

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**.

Створіть джерело даних типу Parameterized Curve 2D, яке буде репрезентувати рухоме коло.

Results **->** Data Set - More Data Sets**->** Parameterized Curve 2D

Data Sets - Parameterized Curve 2D 1:

Data– **Data set** = Study 1/Solution 1

Parameter–**Name** = tau

–**Minimum, Maximum** =-3/2*pi+0.3, pi/2-0.3 (*два поля*)

Expressions – **x** = t[1/s]+cos(-t[1/s]+tau)

– **y** = 1+sin(-t[1/s]+tau)

Введені тут вирази і діапазон зміни параметра tau відповідають зображенню окружності з невеликим розривом. Положення центру кола і розриву буде залежати від моменту часу t, і їх зміщення буде імітувати котіння.

Виконайте команду Plot і отримайте зображення цього джерела даних у вигляді кола. Зазвичай для змінної t система використовує останній момент часу з діапазону, що задається у вузлі Study.

Для імітування обертання малюватимемо також радіус кола, проведений в «робочу» точку. Для нього (радіусу) створіть ще одне ДД параметричної кривої.

Results **->** Data Set - More Data Sets**->** Parameterized Curve 2D

Data Sets - Parameterized Curve 2D 2:

Data– **Data set** = Study 1/Solution 1

Parameter – **Name, Minimum, Maximum** = tau,0, 1 (*три поля*)

Expressions – **x** = t[1/s]-sin(t[1/s])*tau
– **y** = 1-cos(t[1/s])*tau

Тепер все готово для побудови зображення, яке буде складатися з ділянки циклоїди, точки, кола і радіусу.

Створіть двовимірну графічну групу.

Results -> Plot Group - 2D Plot Group

2D Plot Group 1 :

Data – **Data set** = Study 1/Solution 1

Data– **Time(s)** = 10

Тут поле-список **Time (s)** містить момент часу, для якого будуватимуться графіки.

Plot Settings – **Plot data set edges** = False

В графічній групі **2D Plot Group 1** створіть вузли, які будуть містити зображення ділянки циклоїди, рухому точку, коло і радіус, проведений в цю точку.

Для креслення ділянки циклоїди, яку описує рухома точка, використаємо Point Trajectories графік. Цей графік одночасно з траєкторією може зображати кінцеву точку кривої.

2D Plot Group 1 -> Add Plot - More Plots - Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

Data– **Data set** = From parent

Trajectory Data – **Plot data** = Coordinates
– **x-expression, y-expression** = t-sin(t)
– **x-expression, y-expression** = 1-cos(t)

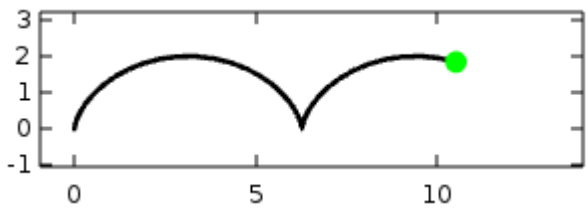
Налаштуйте стиль кривої.

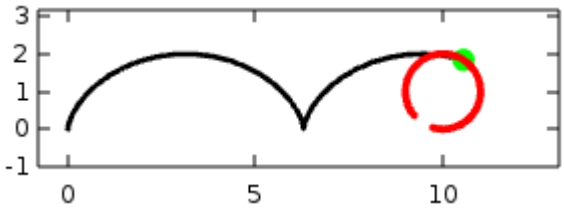
Coloring and Style – **Line style** –**Type** = Tube
– **Line style** – **Tube radius expression** = 0.05
– **Line style** – **Radius scale factor** = True,1
– **Line style** – **Color** = Black

Налаштуйте стиль зображення кінцевої точки кривої.

Coloring and Style – **Point style** –**Type** = Point
– **Point style** –**Point radius expression** = 0.25
– **Point style** – **Radius scale factor** = True,1
– **Point style** – **Color** = Green

LMB > Plot (*наступний рисунок ліворуч*)





В графічну групу додайте Line графік, який репрезентуватиме коло.

2D Plot Group 1-> Add Plot – Line

2D Plot Group 1 – Line 1:

Data – **Data set** = Parameterized Curve 2D 1

Data– **Time(s)** = 10

Цим визначається момент часу, в який будуватиметься коло.

Expression – **Expression** =1

Налаштуйте стиль зображення кола.

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

–**Tube radius expression** =0.05

– **Radius scale factor** =True,1

– **Coloring, Color** =Uniform, Red

LMB > Plot (*попередній рисунок праворуч*)

Додайте в групу ще один Line графік, який зображатиме рухомий радіус.

2D Plot Group 1 – Line 1: RMB > Duplicate

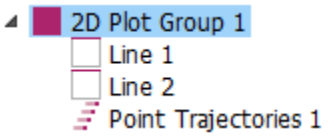
2D Plot Group 1 – Line 2:

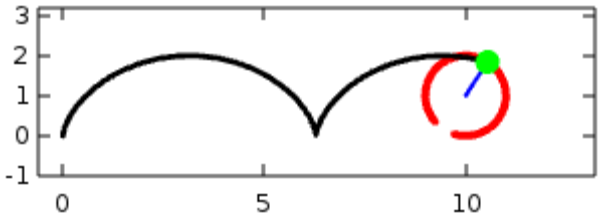
Data – **Data set** = Parameterized Curve 2D 2

Coloring and Style – **Coloring, Color** =Uniform, Blue

LMB > Plot

Графіки вузлів **Line 1** і **Line 2** малюються поверх графіка вузла **Point Trajectories 1** (вони розташовані нижче в ДМ), і частково закривають зображення точки. Щоб цього уникнути, перемістіть вузол **Point Trajectories 1** в ДМ нижче вузлів **Line 1** і **Line 2**, «захопивши» його лівою кнопкою миші і зсовуючи вниз.





Нагадаємо, що рисунок побудований для моментів часу, зазначених в полях **Time (s)** ПВ вузлів **2D Plot Group 1**, **Line 1** і **Line 2**, які для сумісності повинні бути однаковими. Якщо ви поміняєте значення поля **Time (s)** у всіх трьох вузлах на інше (однакове), то отримаєте зображення для нового моменту часу.

Побудуйте анімацію.

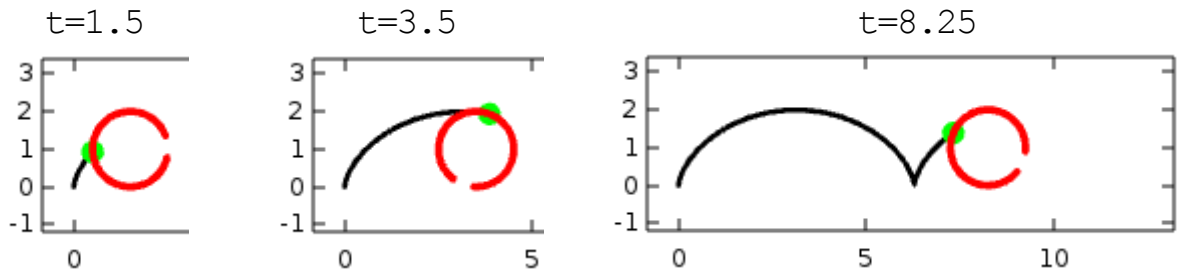
Results –> Animation –> Player

Animation 1:

Scene – **Subject** = 2D Plot Group 1

Frames – **Frame selection** = All

Згенеруйте кадри анімації, натиснувши на кнопку Generate Frame. Запустіть анімацію кнопкою Play. На наступному рисунку показані кадри, побудовані в моменти часу t=1.5,3.5,8.25.



2.3.7. Рух стичного кола.

Нехай задана параметрична крива $x = x(\tau), y = y(\tau)$. Стичним називається коло, що має з кривою порядок дотику не нижче 2.

Позначимо точку дотику через $(x_0, y_0) = (x(\tau_0), x(\tau_0))$. Координати центру (x_{ev}, y_{ev}) стичного кола та його радіус R обчислюються за формулами [6]

$$\begin{aligned} x_{ev} &= x(\tau_0) + y'(\tau_0)\cdot m(\tau_0), \\ y_{ev} &= y(\tau_0) - x'(\tau_0)\cdot m(\tau_0), \\ R &= \sqrt{x'(\tau_0)^2 + y'(\tau_0)^2}\cdot |m(\tau_0)|, \end{aligned} \tag{1}$$

де

$$m(\tau) = \frac{x'(\tau)^2 + y'(\tau)^2}{x''(\tau)y'(\tau) - y''(\tau)x'(\tau)}, \tag{2}$$

Центр стичного кола називають центром кривини, а радіус - радіусом кривини кривої. Геометричне місце центрів кривини для всіляких точок дотику називають еволютою.

Побудуємо анімацію руху стичного кола, коли точка дотику рухається вздовж еліпса. Крім еліпса та кола будемо малювати ділянку еволюти, яку описує рухомий центр кривини.

Приклад. Створіть порожню модель.

File->New->Blank Model

Задайте параметри еліпса.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions-Parameters** в таблиці задайте параметри a=2 і b=1.

Параметричне рівняння еліпса має вигляд $x_e = a\cos\tau$, $y_e = b\sin\tau$, $0 \le \tau \le 2\pi$. В цьому прикладі ми не будемо створювати еліпс в геометрії моделі. Для його зображення згодом буде створено ДД Parameterized Curve 2D. Але нам знадобляться перші і другі похідні виразів $x_e(\tau), y_e(\tau)$. Тому буде зручно відразу створити всі функції.

Спочатку створіть функції, які являють параметричні рівняння еліпса.

Global Definitions : RMB > Functions – Analytic

Function name = xe

Definition –Expression, Arguments = a*cos(tau),tau (*два поля*)

Global Definitions : RMB > Functions – Analytic

Function name = ye

Definition –**Expression, Arguments** = b*sin(tau),tau (*два поля*)

Аналогічно створіть функції, що представлятимуть перші і другі похідні виразів $x_e(\tau), y_e(\tau)$, а також функції (1) і (2). Імена функцій і формули їх обчислення наведені в наступній таблиці.

Ім'я функції	Аргументи	Формула
xe1	tau	d(xe(tau),tau)
ye1	tau	d(ye(tau),tau)
xe2	tau	d(xe1(tau),tau)
ye2	tau	d(ye1(tau),tau)
m	t	(xe1(t)^2+ye1(t)^2)/(xe2(t)*ye1(t)-ye2(t)*xe1(t))
R	t	sqrt(xe1(t)^2+ye1(t)^2)*abs(m(t))
evx	t	xe(t)+ye1(t)*m(t)
evy	t	ye(t)-xe1(t)*m(t)
xc	tau, t	evx(t)+R(t)*cos(tau)
yc	tau, t	evy(t)+R(t)*sin(tau)

Для стислості ми не наводимо інструкції створення функцій. Зробіть це самостійно.

Зерніть вашу увагу на використання операторів символьного диференціювання d(.,.).

Тут функції evx(t), evy(t), R(t) представляють параметричні рівняння еволюти і функцію радіуса кривини, обчислювані за формулами (1). Функції xc(tau,t) і yc(tau,t) представляють параметричні рівняння стичного кола в момент часу t. В цих рівняннях аргумент t визначає положення центра кола і його радіус (він також визначає точку дотику на еліпсі), а параметр tau описує положення точок на колі.

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Геометрія компонента складатиметься з прямокутної розрахункової області.

Geometry **-> Primitives** – Rectangle

Geometry 1 - Rectangle1:

Size – **Width, Height** = 14,14 (*два поля*)

Position – **Base, x, y** = Center,0,0 (*три поля*)

LMB > Build All Objects

Додайте в модель вузол дослідження.

Home -> Study - Add Study = Add Study ->

Preset Studies - Time Dependent **->x**

Задайте діапазон і інтервал зміни параметра часу t.

Study 1 - Step 1: Time Dependent:

Study Settings –**Times** = range(0,0.25,2*pi)

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**.

Створіть допоміжне джерело даних Parameterized Curve 2D, яке репрезентуватиме еліпс.

Results -> Data Set - More Data Sets-> Parameterized Curve 2D

Data Sets - Parameterized Curve 2D 1:

Data– Data set = Study 1/Solution 1

Parameter – Name, Minimum, Maximum = tau,0,2*pi (*три поля*)

Expressions – x, y = xe(tau), ye(tau) (*два поля*)

Виконайте команду **Plot** і отримайте зображення цього джерела даних в формі еліпса.

Створіть ще одне джерело даних Parameterized Curve 2D, яке репрезентуватиме стичне коло.

Results -> Data Set - More Data Sets-> Parameterized Curve 2D

Data Sets - Parameterized Curve 2D 2:

Data– Data set = Study 1/Solution 1

Parameter – Name, Minimum, Maximum = tau,0,2*pi (*три поля*)

Expressions – x, y = xc(tau,t[1/s]), yc(tau,t[1/s]) (*два поля*)

Це джерело являє стичне коло в момент часу t, який визначає точку дотику.

Для еволюти створювати ДД непотрібно, бо її ми зможемо відразу побудувати за допомогою вузла Point Trajectories. Тому переходимо безпосередньо до побудови зображення.

Створіть двовимірну графічну групу.

Results -> Plot Group - 2D Plot Group

2D Plot Group 1 :

Data – Data set = Study 1/Solution 1

Data– Time(s) = 5.25

Тут поле-список **Time (s)** містить момент часу, для якого будуватимуться графіки.

Plot Settings – Plot data set edges = False

В графічній групі **2D Plot Group 1** створіть два вузла Line, які будуть містити зображення еліпса і стичного кола.

2D Plot Group 1-> Add Plot – Line

2D Plot Group 1 – Line 1:

Data – Data set = Parameterized Curve 2D 1

Data– Time(s) = 5.25

Expression – Expression =1

Coloring and Style –Line Type =Tube

–Tube radius expression =0.05

– Radius scale factor =True,1

– Coloring, Color =Uniform, Red

LMB > Plot

В побудованому вузлі буде відображатися еліпс. Створіть ще один вузол Line.

2D Plot Group 1 – Line 1: RMB > Duplicate

2D Plot Group 1 – Line 2:

Data – **Data set** = Parameterized Curve 2D 2

Coloring and Style – **Coloring, Color** =Uniform, Blue

В цьому вузлі буде відображатися стичне коло. Поле-список **Time (s)**=5.25 успадковане від вузла **Line 1**, і містить момент часу, який визначає точку дотику.

Ділянку еволюти, яку описує центр стичного кола, намалюємо у вузлі Point Trajectories. Одночасно в ньому можна зобразити центр кривини.

2D Plot Group 1 –> Add Plot - More Plots - Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

Data– **Data set** = From parent

Trajectory Data – **Plot data** = Coordinates

– **x-expression, y-expression** = evx(t)

– **x-expression, y-expression** = evy(t)

Налаштуйте стиль кривої.

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

– **Line style** – **Tube radius expression** = 0.05

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

– **Line style** – **Color** = Black

Налаштуйте стиль зображення точки центру кривини.

Coloring and Style – **Point style** –**Type** = Point

– **Point style** –**Point radius expression** = 0.25

– **Point style** – **Radius scale factor** = True,1

– **Point style** – **Color** = Black

LMB > Plot (*наступний рисунок ліворуч*)

Створена крива являє відрізок еволюти. Він починається в точці, яка відповідає положенню центру стичного кола при t=0, і закінчується в центрі кола, побудованого в поточний момент часу.

Для наочності додамо на рисунок зображення точки дотику кола до еліпса. Це можна зробити за допомогою такого ж вузла Point Trajectories, відключивши в ньому відображення кривої і залишивши відображення точки.

2D Plot Group 1 - Point Trajectories 1 : RMB > Duplicate

2D Plot Group 1 - Point Trajectories 2 :

Trajectory Data – **x-expression, y-expression** = xe(t)

Trajectory Data – **x-expression, y-expression** = ye(t)

Coloring and Style – **Line style** –**Type** = None

Останнім відключається відображення кривої. Наступні інструкції налаштовують зображення точки дотику.

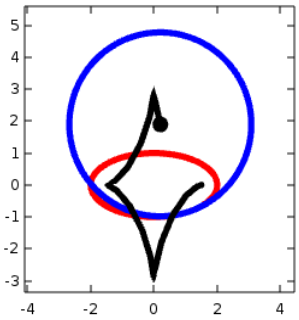
Coloring and Style – **Point style** –**Type** = Point

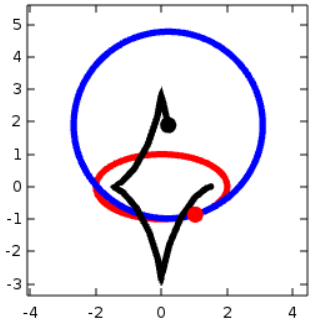
– **Point style** –**Point radius expression** = 0.25

– **Point style** – **Radius scale factor** = True,1

– **Point style** – **Color** = Red

LMB > Plot (*наступний рисунок праворуч*)





Побудуйте анімацію.

`Results` –> Animation –> Player

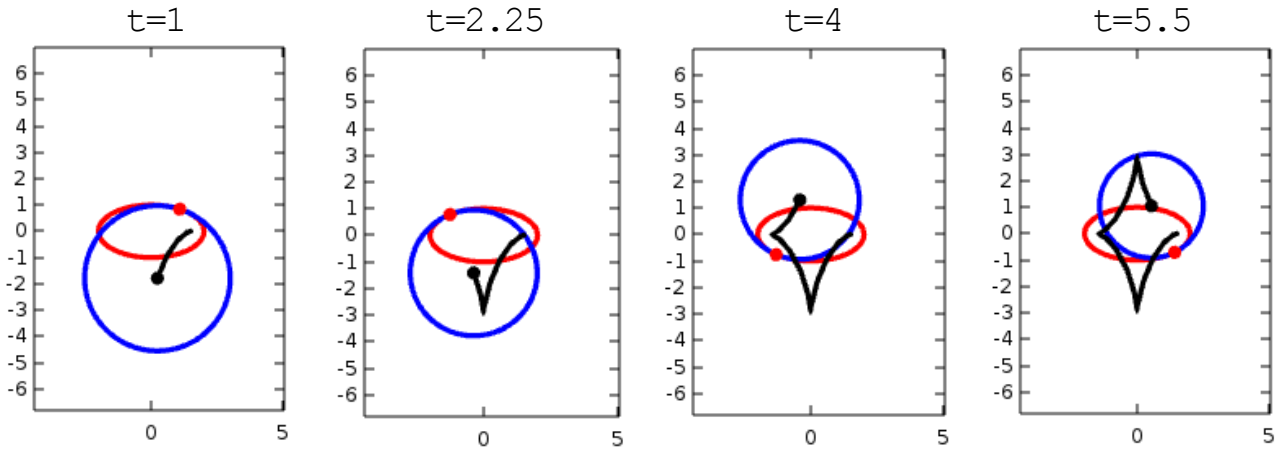
Animation 1:

`Scene` – **Subject** = `2D Plot Group 1`

`Frames` – **Frame selection** = `All`

`Playing` – **Repeat** = `True`

Згенеруйте кадри анімації, натиснувши на кнопку Generate Frame, і запустіть її кнопкою Play. На наступному рисунку показані кадри, побудовані в моменти часу `t=1,2.25,4,5.5`.



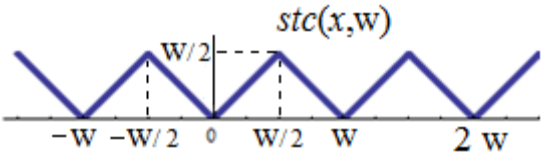
Вправа. Замість параметричних рівнянь еліпса використайте рівняння якоїсь іншої замкненої кривої. Наприклад, до виразів $a\cos\tau$, $b\sin\tau$ додайте невеликі тригонометричні доданки. Побудуйте рух стичного кола до такої кривої.

Вказівка. Якщо ділянка кривої матиме точки перегину, то радіус стичного кола в цих точках перетворюватиметься на нескінченність. Уникайте цього випадку.

2.3.8. Математичний більярд в прямокутнику.

Є горизонтальний більярдний стіл довільної форми без луз (плоска область). 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, що обмежена бортом Г (межею області Q), - називається математичним більярдом. Траєкторія точки в математичному більярді визначається її початковим положенням і вектором швидкості.

Для побудови траєкторії точки в прямокутному більярді нам знадобиться періодична пилкоподібна функція наступного вигляду.

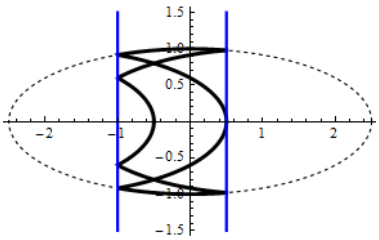


Позначимо її $stc(x,w)$. Ця функція є парною і періодичною по змінній x з періодом w. На відрізку $-w/2 \le x \le w/2$ вона збігається з $|x|$, а поза відрізком повторюється періодично.

Наведемо без доведення деякі факти, пов'язані з перетворенням відбиття кривих. Нехай крива C задана параметричними рівняннями $x = x(t), y = y(t)$. Рівняння кривої C_v, отриманої багаторазовим внутрішнім відбиттям від пари вертикальних прямих $x = x_l$, $x = x_r$ $(x_l < x_r)$, мають вигляд

$$x_v = x_l + stc\left(x(t) - x_l, 2(x_r - x_l)\right), \quad y_v = y(t),$$

На наступному рисунку показаний приклад кривої, отриманої відбиттям еліпса від двох вертикальних прямих.



Аналогічно, рівняння кривої C_h, отриманої багаторазовим внутрішнім відбиттям C від пари горизонтальних прямих $y = y_d$, $y = y_u$ $(y_d < y_u)$ має вигляд $x_h = x(t), y_h = y_d + stc\left(y(t) - y_d, 2(y_u - y_d)\right)$.

Параметричні рівняння кривої C_t, отриманої багаторазовим внутрішнім відбиттям лінії $C: x = x(t), y = y(t)$ від границь прямокутника $x_l \le x \le x_r$, $y_d \le y \le y_u$, має вигляд.

$$\begin{aligned} x_t(t) &= x_l + stc\left(x(t) - x_l, 2(x_r - x_l)\right) \\ y_t(t) &= y_d + stc\left(y(t) - y_d, 2(y_u - y_d)\right) \end{aligned} \qquad (1)$$

Траєкторія точки в прямокутному більярді є ламаною, отримуваною багаторазовим відбиттям прямолінійного променя від «бортів» прямокутника. Нехай з точки (x_0, y_0) вилітає нескінченно мала кулька в напрямку вектора $(\cos\alpha, \sin\alpha)$. Тертя відсутнє. Якби бортів області не було, то вона рухалася б вздовж прямої $x = x_0 + ct\cos\alpha$, $y = y_0 + ct\sin\alpha$, де t - час, c - швидкість, α - кут нахилу променя. Рівняння більярдної траєкторії буде мати вигляд (1), де в якості функцій $x(t), y(t)$ слід використовувати рівняння цієї прямої.

В моделі поточного прикладу ми побудуємо анімацію руху точки в прямокутному більярді, вважаючи c=1.

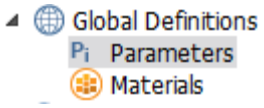
Приклад. Створіть порожню модель.

File->New->Blank Model

Створіть вузол параметрів.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** заповніть таблицю параметрів так, як показано нижче.



Parameters

Name	Expression	Value	Description
xl	-1	-1	Лівий борт
xr	1	1	Правий борт
wd	xr-xl	2	Ширина
yd	-2	-2	Нижній борт
yu	2	2	Верхній борт
ht	yu-yd	4	Висота
T	18	18	Максимальний час
x0	0.75	0.75	Абсциса початкової точки
y0	0.5	0.5	Ордината початкової точки
vx	2	2	Абсциса спрямувального вектора
vy	1	1	Ордината спрямувального вектора
cosa	vx/sqrt(vx^2+vy^2)	0.89443	
sina	vy/sqrt(vx^2+vy^2)	0.44721	

Тут параметри `x0,y0` є початковими координатами точки, а `vx,vy` представляють початковий вектор напрямку руху, який нормується до вектора одиничної довжини `cosa,sina`.

Аналітичні (`Analitic`) функції системи мають опцію періодичного продовження, яка не може бути аргументом. Але ділянку періоду можна задавати параметрами. Отже, щоб застосувати формули (1), потрібно побудувати дві періодичні пилкоподібні функції `stc` з періодами 2 wd і 2 ht.

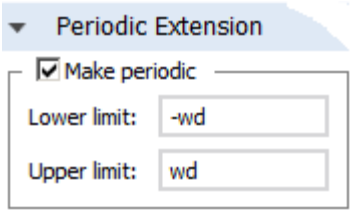
Global Definitions : RMB > Functions – Analytic

Function name = `stcw`

Definition –**Expression, Arguments** = `abs(x),x` (*два поля*)

Periodic Extension – **Make periodic** = `True`

– **Lower limit, Upper limit** = `-wd,wd` (*два поля*)



Global Definitions : RMB > Functions – Analytic

Function name = `stch`

Definition –**Expression, Arguments** = `abs(x),x` (*два поля*)

Periodic Extension – **Make periodic** = `True`

– **Lower limit, Upper limit** = `-ht,ht` (*два поля*)

Створіть ще дві функції, які представлятимуть параметричні рівняння (1) траєкторії точки.

Global Definitions : RMB > Functions – Analytic

Function name = `xt`

Definition – **Expression** = `xl+stcw(x0+tau*cosa-xl)`

Definition – **Arguments** = `tau`

Global Definitions : RMB > Functions – Analytic

Function name = `yt`

Definition – **Expression** = `yd+stch(y0+tau*sina-yd)`

Definition – **Arguments** = `tau`

Додайте в модель двовимірний компонент з прямокутною розрахунковою областю, яка буде репрезентувати більярдний стіл.

Untitled.mph : RMB > Add Component - 2D

`Geometry` **-> Primitives** – Rectangle

Geometry 1 - Rectangle1:

Size – **Width, Height** = `wd,ht` (*два поля*)

Position – **Base, x, y** = `Corner,xl,yd` (*три поля*)

Додайте в модель вузол 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)`

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**.

Для зображення траєкторій точок призначені графіки `Point Trajectories`. Створіть один такий вузол в двовимірній графічній групі.

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

2D Plot Group 1 :

Data – **Data set** = `Study 1/Solution 1`

Data– **Time(s)** = `16.3`

`2D Plot Group 1` -> Add Plot - More Plots - Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

Data– **Data set** = `From parent`

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

– **x-expression, y-expression** = `xt(t)`

– **x-expression, y-expression** = `yt(t)`

Налаштуйте стиль ламаної.

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

– **Line style** – **Tube radius expression** = `0.02`

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

– **Line style** – **Color** = `Blue`

Налаштуйте стиль зображення рухомої точки.

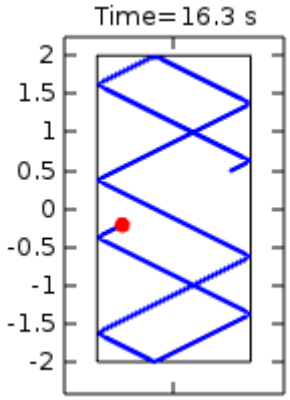
Coloring and Style – **Point style** –**Type** = `Point`

– **Point style** –**Point radius expression** = `0.07`

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

– **Point style** – **Color** = `Red`

LMB > Plot



Створена крива являє відрізок траєкторії, який починається в момент часу `t=0` в точці (x_0, y_0) і закінчується в момент `t`, заданий у вузлі **Time(s)** батьківської графічної групи.

Побудуйте анімацію.

`Results` –> Animation –> Player

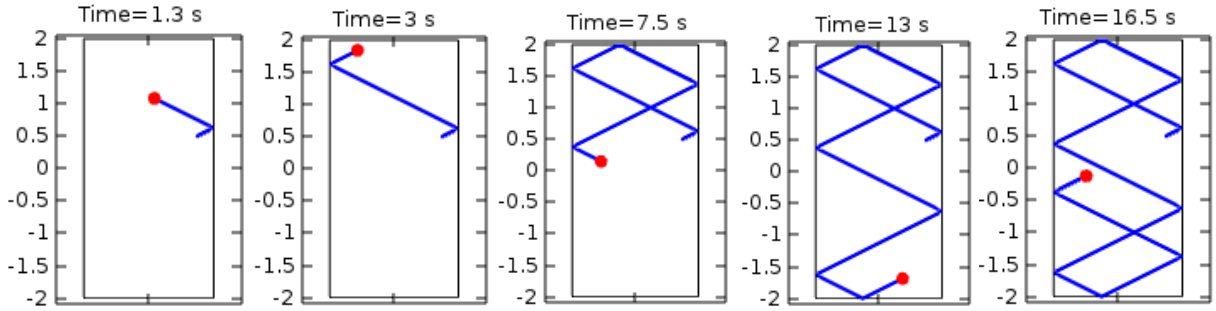
Animation 1:

`Scene` – **Subject** = `2D Plot Group 1`

`Frames` – **Frame selection** = `All`

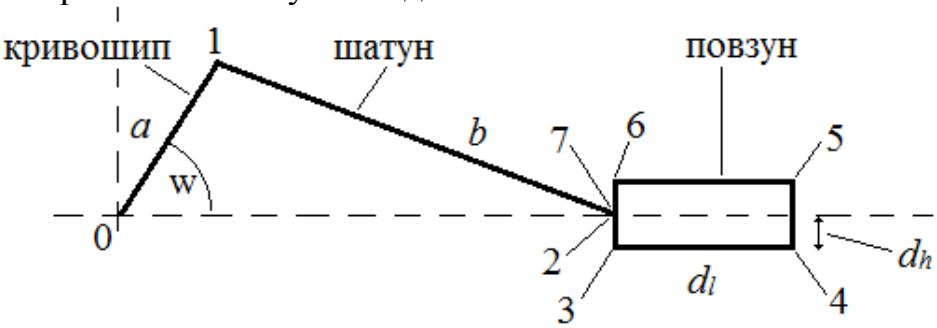
`Playing` – **Repeat** = `True`

Згенеруйте кадри анімації, натиснувши на кнопку Generate Frame, і запустіть її кнопкою Play. На наступному рисунку показані кадри, побудовані в моменти часу `t=1.3,3,7.5,13,16.5`.



2.3.9. Кривошипно-шатунний механізм.

Кривошипно-шатунний механізм є пристроєм, який обертальний рух перетворює на поступальний (або навпаки). Змоделюємо схематично рух цього механізму, зобразивши його у вигляді ламаної.



Позначимо кут повороту кривошипа через w, довжину – через *a*, довжину шатуна – через *b*, розміри прямокутного повзуна $-d_l \times 2d_h$. Занумеруємо вузли

ламаної індексами від 0 до 7, причому вершина 7 співпадатиме з вершиною 2. Шарнірами механізму будуть точки 1 і 2, а вершина 0 – точкою обертання. Координати вершин ламаної будуть визначатися наступними формулами.

Координати вершин x_i, y_i
$x_1 = a \cdot cos\, w$ $y_1 = a \cdot sin\, w$
$x_2 = x_1 + \sqrt{b^2 - a^2 sin^2 w}$ $y_2 = y_1 - a \cdot sin\, w$
$x_3 = x_2$ $y_3 = y_2 - d_h$
$x_4 = x_3 + d_l$ $y_4 = y_3$
$x_5 = x_4$ $y_5 = y_4 + 2d_h$
$x_6 = x_5 - d_l$ $y_6 = y_5$
$x_7 = x_6$ $y_7 = y_6 - d_h$

Позначимо кутову швидкість обертання кривошипа (лівого відрізка ламаної) через ω [рад/сек]. Кут його повороту `w` в момент `t[сек]` буде дорівнювати $w = \omega \cdot t$. В цій задачі кут `w` є визначальним для форми механізму, і ламану логічніше характеризувати цим кутом, а не моментом часу.

У попередніх прикладах величиною, що визначала положення рухомих об'єктів, був час. Однак в Comsol Multiphysics керуючим може бути будь-який параметр моделі, і в поточному прикладі їм буде `w`. Це означає, що для моделювання процесу руху механізму слід застосувати параметричний аналіз по куту `w`.

Таким чином, схема побудови моделі буде наступною:

- створюється шаблон порожньої моделі;
- створюються параметри розмірів моделі, кута `w`, і залежні від них параметри координат вузлів ламаної;
- створюється двовимірний компонент, в якому кресляться прямокутна розрахункова область і внутрішня полілінія (ламана) з координатами вузлів, заданими в таблиці параметрів;
- виконується параметричний аналіз по параметру `w`;
- будуються зображення механізму (ламаної) і анімація його руху в залежності від кута `w`.

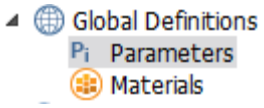
Приклад. Створіть порожню модель.

File->New->Blank Model

Створіть вузол параметрів моделі.

Global Definitions : RMB > Parameters

В ПВ вузла **Global Definitions-Parameters** задайте параметри розмірів *a*, *b*, *dl*, *dh*, кут w, а також координати вершин x_i, y_i відповідно до формул, наведених вище.



Parameters

Name	Expression	Value	Description
a	1	1	довжина кривошипа
b	3	3	довжина шатуна
dl	1	1	довжина повзуна
dh	0.25	0.25	піввисота повзуна
w	0.5	0.5	кут в радіанах
x1	a*cos(w)	0.87758	абсциса 1-го шарніра
y1	a*sin(w)	0.47943	ордината 1-го шарніра
x2	x1+sqrt(b^2-a^2*sin(w)^2)	3.839	
y2	y1-a*sin(w)	0	
x3	x2	3.839	
y3	y2-dh	-0.25	
x4	x3+dl	4.839	
y4	y3	-0.25	
x5	x4	4.839	
y5	y4+2*dh	0.25	
x6	x5-dl	3.839	
y6	y5	0.25	
x7	x6	3.839	
y7	y6-dh	0	

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Геометрія компонента складатиметься з прямокутника і ламаної в формі кривошипно-шатунного механізму.

Створіть прямокутник.

`Geometry` **-> `Primitives`** – Rectangle

Geometry 1 - Rectangle1:

`Size` – **Width, Height** = `6.2,2.2` (*два поля*)

`Position` – **Base, x, y** = `Corner,-1.1,-1.1` (*три поля*)

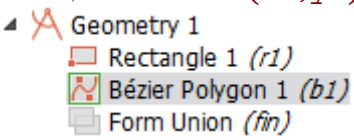
Побудуйте ламану, використовуючи графічний примітив `Bézier Polygon`.

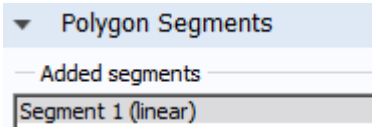
`Geometry` -> **`Pimitives`** – Bézier Polygon

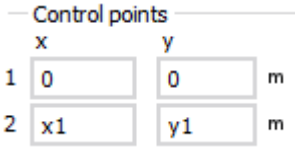
Bézier Polygon 1 : `General` – **Type** = `Open curve`

Bézier Polygon 1 : LMB > Add Linear

Для зображення відрізка кривошипа введіть координати його початкової (`0,0`) і кінцевої точок (`x1,y1`).

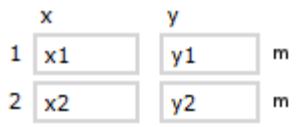


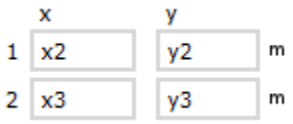




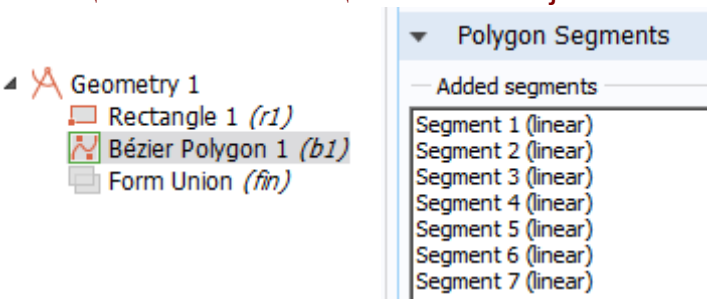
Потім додавайте відрізки до ламаної, клацаючи по кнопці Add Linear і вводячи координати початкової і кінцевої точок. Наприклад,

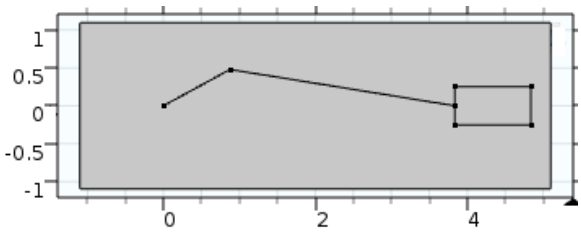
`Segment 2 (linear)(відрізок 1-2)` `Segment 3 (linear) (відрізок 2-3)`





Всього потрібно побудувати 7 сегментів. Завершіть будування ламаної клацанням по кнопці Build All Objects.





Додайте в модель `Stationary` тип дослідження.

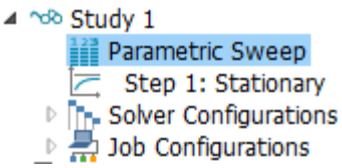
Untitled.mph (root): RMB > Add Study = `Add Study ->`

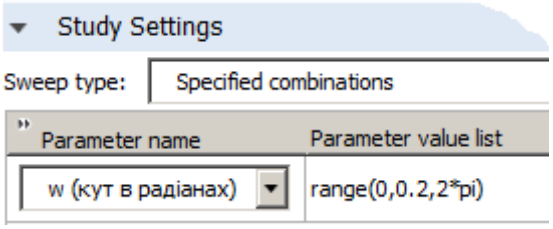
Preset Studies - Stationary **->x**

В ДМ з'являється гілка **Study 1**. Для параметричного аналізу в ній потрібно створити підвузол **Parametric Sweep**, в ПВ якого задати список розглядуваних значень параметра.

`Study` **-> Study Step** – Parametric Sweep

В ПВ вузла **Parametric Sweep** в розділі **Study Settings** є таблиця для введення значень параметрів. Клацніть по таблиці правою кнопкою миші і виберіть команду Add. Із випадного списку виберіть параметр `w`. В другому стовпці таблиці задайте діапазон його зміни `range(0,0.2,2*pi)`.





Виконайте обчислення.

`Home` **-> Study** – Compute

В гілці **Results - Data Sets** буде створено два джерела даних **Sudy 1/Solution 1** і **Study 1/Parametric Solutions 1**. Якщо планується побудова анімації, то для графічних вузлів слід обирати друге ДД.

Будувати графік ламаної (механізму) можна двома способами. В обох з них в секціях `Selection` потрібно буде вибирати ламану, а точніше всі її відрізки. Якщо вибір одних і тих же елементів геометрії потрібно повторювати кілька разів, то в компоненті в гілці **Definitions** можна створити вузол вибірки, обрати в ньому потрібні геометричні елементи, дати набору/ім'я, і використовувати його в тих місцях, де потрібні всі елементів вибірки.

Виконайте команду.

`Definitions` **-> Selections** – Explicit

З'явиться вузол **Explicit 1**, який ми зараз перейменуємо.

Definitions - Explicit 1:

Label = `Polyline_1`

Input Entities – **Geometric entity level** = Boundary

Цим обирається рівень геометрії, елементи якого складатимуть вибірку. В графічному вікні поклацайте мишею по всіх відрізках ламаної. Їх індикатори з'являться в секції вибору.

Selection – **Selection, <>** = Manual;4,5,6,7,8,9,10

(*відрізки ламаної*)

Як наслідок, вибірка з назвою Polyline_1 буде з'являтися як окремий елемент у всіх списках Selection.

Побудуємо графік ламаної (механізму).

Перший спосіб полягає в використанні в графічній групі 1D Plot Group вузла Line Graph.

Results -> Plot Group - 1D Plot Group

1D Plot Group 1 :

Data – **Data set** = Study 1/Parametric Solution 1

Data – **Parameter selection (w)** = From list

При виборі в полі **Parameter selection (w)** режиму ручного введення моментів часу з'являється поле-список **Parameter value (w)**, в якому слід вибрати значення параметра w, для якого будуватимуться графіки в дочірніх вузлах групи.

Data– **Parameter value (w)** = 0.6 (*вибір зі списку*)

Axis – **Manual axis limits** = True

– **x minimum, x maximum** = -1.1,6

– **y minimum, y maximum** = -1.1,1.1

Оскільки далі ми збираємося використовувати поточну графічну групу для побудови анімації, то в розділі **Axis** ми визначили спільний для всіх моментів часу діапазон зміни просторових координат $-1.1 \le x \le 6, -1.1 \le y \le 1.1$.

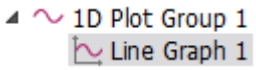
1D Plot Group 1-> Add Plot – Line Graph

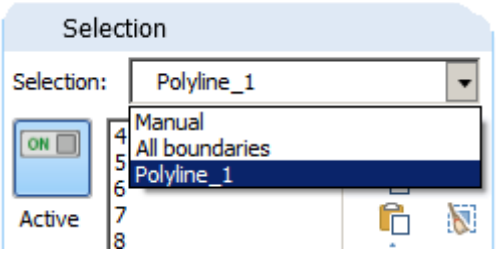
1D Plot Group 1 - Line Graph 1:

Data – **Data set** = From parent

Selection – **Selection** = Polyline_1

При виборі елементів геометрії ми використали ім'я створеної раніше вибірки.



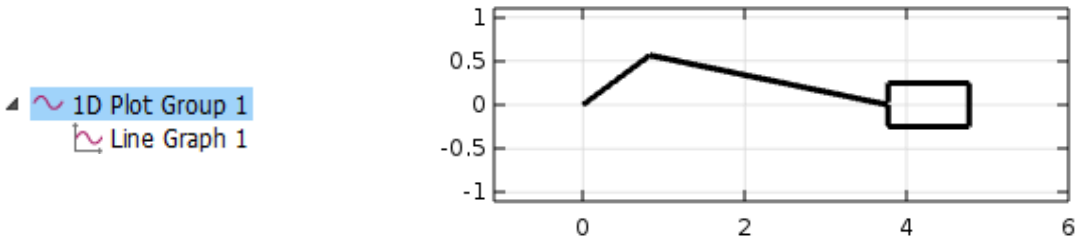


y-Axis Data – **Expression** =y

x-Axis Data – **Parameter, Expression** = Expression,x (*два поля*)

Coloring and Style – **Color, Width** = Black,3 (*два поля*)

LMB > Plot



Нагадаємо, що зображення побудовано для кута w, зазначеного в списку **Parameter value (w)** батьківського вузла **1D Plot Group 1**.

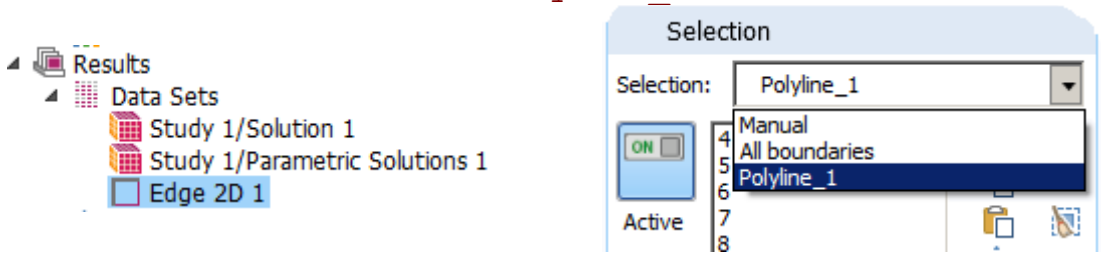
Другий спосіб побудови ламаної полягає в використанні в графічній групі `2D Plot Group` вузла типу `Line`, який спирається на ДД `Edge 2D`. Воно буде репрезентувати ламану і підключене до ДД **Study 1/Parametric Solution 1**.

`Results` -> Data Set - More Data Sets -> Edge 2D

Edge 2D 1:

Data – **Data set** = `Study 1/Parametric Solution 1`

Selection – **Selection** = `Polyline_1`



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

2D Plot Group 2 :

Data – **Data set** = `Edge 2D 1`

Data – **Parameter value(w)** = `1`

`2D Plot Group 2`-> Add Plot – Line

2D Plot Group 2 – Line 1:

Data – **Data set** = `From parent`

Expression – **Expression** =`1`

Налаштуйте стиль ламаної.

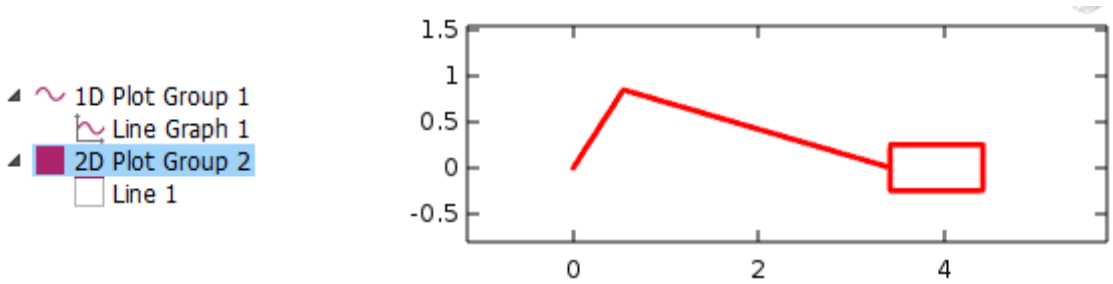
Coloring and Style – **Line Type** =`Tube`

–**Tube radius expression** =`0.03`

– **Radius scale factor** =`True,1`

– **Coloring, Color** =`Uniform,Red`

LMB > Plot



На основі будь-якої з цих графічних груп можна побудувати анімацію руху «механізму». Побудуємо її, використовуючи **2D Plot Group 1** .

`Results` –> Animation –> Player

Animation 1:

Scene – **Subject** = 2D Plot Group 1

Frames – **Frame selection** = All

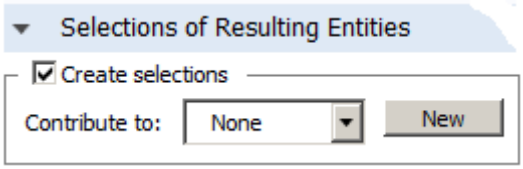
Playing – **Repeat** = True

Вгорі ПВ **Animation 1** натисніть кнопку Generate Frame. Почнеться процес генерації кадрів. По його завершенні натисніть кнопку Play на панелі управління анімацією, яка розташована в лівому верхньому куту графічного вікна. Цим ви запустите анімацію.

Перед побудовою анімації підберіть мишею пропорції графічного вікна так, щоб на екрані горизонтальні і вертикальні одиниці довжини були приблизно однаковими.

Зауваження. Якщо ви плануєте будувати лише графік механізму при деякому значенні кута w, то виконувати параметричний аналіз непотрібно. Він знадобиться, якщо ви збираєтесь створювати анімацію.

Зауваження. Для кривої Без'є створити вибірку можна простіше. В ПВ вузла **Bézier Polygon 1** в розділі **Selections of Resulting Entities** треба встановити прапорець **Create selections**. В цьому випадку окремий вузол вибірки в ДМ не створюється, але вибірка з ім'ям Bézier Polygon 1 буде присутня в усіх списках вибору.



2.3.10. Стрічка Мебіуса.

Стрічка Мебіуса - найпростіша неорієнтована поверхня. Вона утворюється рухом і обертанням відрізка прямої вздовж замкнутої просторової кривої. Нехай ця крива буде плоскою окружністю радіуса R, а n позначає кількість напівобертів відрізка при обході кривої. Тоді рівняння поверхні можна записати в параметричному вигляді

$$x(u,v)=\left(R+v\cos\left(\frac{nu}{2}\right)\right)\cos u,$$
$$y(u,v)=\left(R+v\cos\left(\frac{nu}{2}\right)\right)\sin u, \qquad \begin{matrix}0\le u\le 2\pi\\ -H/2\le v\le H/2\end{matrix}, \qquad (1)$$
$$z(u,v)=v\sin\left(\frac{nu}{2}\right),$$

де H позначає довжину рухомого відрізка (ширину стрічки).

В моделі поточного прикладу ми побудуємо поверхню стрічки Мебіуса, а також змоделюємо рух відрізка, що замітає цю поверхню.

Приклад. Створіть порожню модель.

File->New->Blank Model

Задайте параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions-Parameters** в таблиці задайте параметри `R=3,H=2,n=1,maxU=2*pi`. Тут `H` позначає ширину стрічки, а `n` - кількість напівобертів відрізка, і при непарному `n` вийде неорієнтована поверхня.

Створіть функцію `X(u,v)` відповідно до формул (1).

Global Definitions : RMB > Functions – Analytic

Function name = `X`

`Definition` – **Expression** = `(R+v*cos(n*u/2))*cos(u)`

`Definition` – **Arguments** = `u,v`

Аналогічно створіть дві інші функції.

`Y(u,v)= (R+v*cos(n*u/2))*sin(u)`

`Z(u,v)=v*sin(n*u/2)`

Додайте в модель тривимірний компонент.

`Home` **–>** Model **–>** Add Component – 3D

Його геометрія складатиметься з поверхні стрічки.

`Geometry` **-> `Primitives` – `More Primitives`** – Parametric Surface

Geometry 1 – Parametric Surface 1 (ps1):

`Parameter` – **First parameter** – **Name** = `u`

– **First parameter** – **Minimum, Maximum** = `0,maxU`

`Parameter` – **Second parameter** – **Name** = `v`

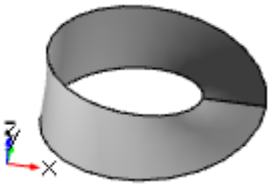
– **Second parameter** – **Minimum, Maximum** = `-H/2,H/2`

`Expression` – **x, y, z** = `X(u,v),Y(u,v),Z(u,v)` (*три поля*)

`Advanced Settings` – **Relative tolerance** = `1E-3`

Зверніть увагу на те, що ми зменшили точність «внутрішнього» зображення поверхні (опція **Relative tolerance** = `1E-3`).

LMB > Build Selected



Зауваження. Ми не створюємо в геометрії додатково тривимірну розрахункову область, бо алгоритм побудови СкЕ сітки при наявності неорієнтовної поверхні може дати збій. Якщо будувати частину неорієнтовної поверхні, то проблем не виникає (бо вона буде орієнтовною).

Додайте в модель стаціонарний тип дослідження.

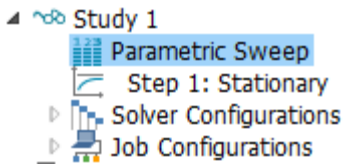
Untitled.mph (root): RMB > Add Study = `Add Study ->`

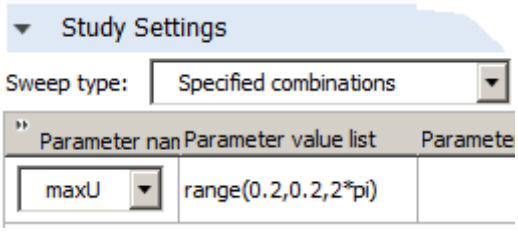
Preset Studies - Stationary **->x**

При моделювання руху твірного відрізка і процесу замітання ділянки поверхні потрібно будувати частину стрічки Мебіуса, яка обмежена діапазоном параметра $0 \le u \le \max U$. Для варіювання величини `maxU` в гілку **Study 1** слід додати вузол параметричного аналізу **Parametric Sweep**.

Study -> **Study Step** – Parametric Sweep

В ПВ вузла **Parametric Sweep** в розділі **Study Settings** клацніть по таблиці параметрів (вона спочатку порожня) правою кнопкою миші і виберіть команду Add. Зі списку параметрів, що з'явиться, виберіть maxU. В другому стовпці таблиці задайте діапазон зміни цього параметра range(0.2,0.2,2*pi).





Виконайте обчислення.

Home -> **Study** – Compute

В гілці **Results - Data Sets** з'являться два джерела даних **Sudy 1/Solution 1** і **Study 1/Parametric Solutions 1**. Для параметричного аналізу слід використовувати друге.

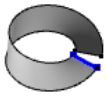
Окрім ділянки поверхні ми будемо зображати твірний відрізок (той, що замітає поверхню). Для цього нам знадобиться ДД Edge 3D.

Results -> Data Set - More Data Sets-> Edge 3D

Edge 3D 1:

Data – **Data set** = Study 1/Parametric Solutions 1

Selection – **Selection,** <> = Manual,1



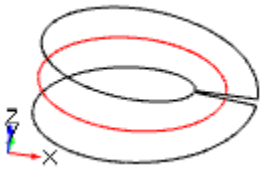
Для зображення на графіку напрямного кола теж потрібно джерело даних. Його можна створити як перетин площини z=0 і поверхні повної стрічки Мебіуса. Інформація про геометрію повної стрічки зберігається в джерелі даних **Sudy 1/Solution 1**.

Results -> Data Set -> Cut Plane

Data – **Data set** = Study 1/Solutions 1

Plane Data – **Plane** = xy-planes

LMB > Plot



Почнемо побудову зображення. Створіть тривимірну графічну групу і в ній графіки ділянки стрічки, твірного відрізка і напрямного кола.

Results -> Plot Group - 3D Plot Group

3D Plot Group 1 :

Data – **Data set** = Study 1/Parametric Solutions 1

Data– **Parameter value (maxU)** = 5.6

Plot Settings – **Plot data set edges** = False

Спочатку будуємо ділянку поверхні стрічки.

3D Plot Group 1-> Add Plot – Surface

3D Plot Group 1 - Surface 1 :

Data – **Data set** = Study 1/Parametric Solutions 1
Data– **Parameter value (maxU)** = 5.6
Expression – **Expression** = 1
Coloring and Style – **Coloring** = Color table
– **Color table** = Wave
– **Color legend** = False

Потім будуємо твірний відрізок.

3D Plot Group 1–> Add Plot – Line

3D Plot Group 1 - Line 1 :

Data – **Data set** = Edge 3D 1
Data– **Parameter value (maxU)** = 5.6
Expression – **Expression** =1
Coloring and Style – **Line type** =Tube
– **Tube radius expression** =0.07
– **Radius scale factor** =True,1
– **Coloring** = Color table
– **Color table** = DiacoLight
– **Color legend** = False

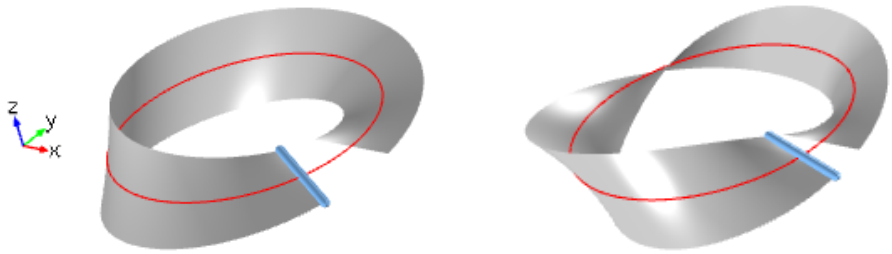
Нарешті будуємо напрямне коло.

3D Plot Group 1 - Line 1: RMB > Duplicate

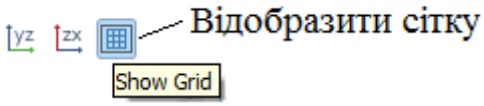
3D Plot Group 1 - Line 2:

Data – **Data set** = Cut Plane 1
Coloring and Style – **Coloring, Color** =Uniform,Red

Виконайте команду Plot і отримайте графік, показаний на наступному рисунку ліворуч (n=1). На рисунку праворуч зображена поверхня, отримана при n=2.



Щоб на графіку не відображалася координатна сітка достатньо натиснути на кнопку Show Grid, розташовану над графічної зоною.



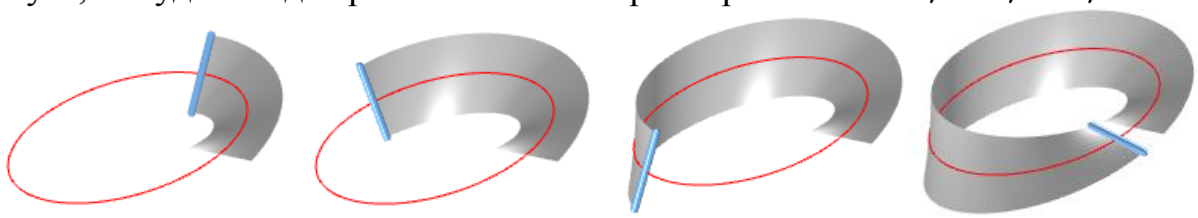
Тепер побудуйте анімацію.

Results –> Animation –> Player

Animation 1:

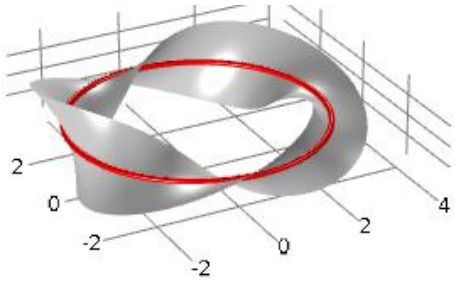
Scene – **Subject** = 3D Plot Group 1
Frames – **Frame selection** = All
Playing – **Repeat** = True

Згенеруйте кадри анімації, натиснувши на кнопку Generate Frame, і запустіть її кнопкою Play. На наступному рисунку показані ділянки поверхні стрічки Мебіуса, побудовані для різних значень параметра `maxU=1.4,2.4,4.4,6.0`.



Зауваження. Для відтворення поверхні цілком достатньо було б створити джерело даних **Sudy 1/Solution 1**. Однак, для побудови анімації потрібно мати джерело даних **Study 1/Parametric Solutions 1**, яке містить інформацію про всі значення sweep-параметра `maxU` і відповідні йому геометрії (ділянки стрічки).

Вправа. Змініть значення параметра `n` на `3` і переобчисліть модель. Ви повинні отримати наступне зображення.



2.3.11. Коливання струни з закріпленими кінцями.

Поперечні коливання струни, поздовжні коливання пружного стрижня, і багато інших явищ описуються одновимірним хвильовим рівнянням

$$\frac{\partial^2 u(x,t)}{\partial t^2} = a^2 \frac{\partial^2 u(x,t)}{\partial x^2}, \qquad (1)$$

де $u(x,t)$ являє зміщення з положення рівноваги точки x одновимірного середовища в момент часу t. Для однозначного визначення розв'язку рівняння (1) необхідно задати початкові умови

$$u(x,0) = \varphi(x),\ u'_t(x,t) = \psi(x). \qquad (2)$$

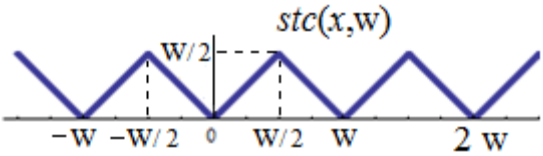
і граничні умови, якщо у коливного об'єкта є межа.

Для необмеженої прямої відомий [2] загальний розв'язок задачі (1), (2) в формі Д'Аламбера

$$u(x,t) = \frac{\varphi(x+at)+\varphi(x-at)}{2} + \frac{1}{2a}\int_{x-at}^{x+at}\psi(\alpha)d\alpha \qquad (-\infty < x < \infty, t \ge 0). \quad (3)$$

При цьому вираз (3) при будь-яких функціях $\varphi(x)$, $\psi(x)$ дає розв'язок рівняння (1).

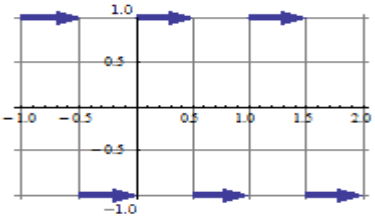
Визначимо парну «пилкоподібну» періодичну по змінній x функцію $\mathrm{stc}(x,w)$ з періодом w, яка на відрізку $-w/2 \le x \le w/2$ збігається з функцією $|x|$, а поза відрізком повторюється періодично.



Існує кілька формул для її побудови [3], наприклад:

$$\mathrm{stc}(x,w)=\left|x-w\left[\frac{x}{w}+\frac{1}{2}\right]\right| \quad \text{або} \quad \mathrm{stc}(x,w)=\frac{w}{2\pi}\arccos\left(\cos\frac{2\pi x}{w}\right).$$

Також визначимо періодичну знакову функцію $\mathrm{psign}(x,w)$, яка на півінтервалі $[-w/2,0)$ дорівнює -1, на ділянці $[0,w/2)$ дорівнює 1, і далі повторює свої значення періодично з періодом w. Можна записати кілька різних формул, що реалізують цю функцію, наприклад, $(-1)^{[2x/w]}$, де квадратні дужки $[z]$ позначають цілу частину числа z. На наступному рисунку зображено графік функції $\mathrm{psign}(x,1)$.



Основна ідея розв'язання крайової задачі для однорідного одновимірного хвильового рівняння (1) на скінченному відрізку $0\le x\le L$ при початкових умовах (2) і нульових граничних умовах $u(0,t)=0$, $u(L,t)=0$, $t\ge 0$ в роботі [3] сформульована в наступному вигляді. Відповідно до основних уявлень методу продовження виконується непарне періодичне продовження початкових функцій $\varphi(x)$ і $\psi(x)$ на всю вісь (з відрізка $[0,L]$ непарно на відрізок $[-L,0]$ і далі на всю вісь з періодом $2L$) за формулами

$$\Phi(x)=\mathrm{psign}(x,2L)\cdot\varphi(\mathrm{stc}(x,2L)) \quad \text{і} \quad \Psi(x)=\mathrm{psign}(x,2L)\cdot\psi(\mathrm{stc}(x,2L)). \qquad (4)$$

Отримані функції $\Phi(x)$ і $\Psi(x)$ на відрізку $0\le x\le L$ збігаються з $\varphi(x)$ і $\psi(x)$ відповідно. Відома лема [2], якщо початкові дані задачі про поширення коливань на необмеженій прямій є непарними функціями відносно деякої точки x_0, то відповідне рішення рівняння (1) в цій точці дорівнює нулю. Функції $\Phi(x)$ і $\Psi(x)$ є непарними відносно точок $x=0$ і $x=L$. Їх використання у формулі Д'Аламбера (3) дає розв'язок рівняння (1). Відповідно до леми, воно буде задовольняти нульовим граничним умовам. Також на відрізку $[0, L]$ воно буде задовольняти початковим умовам (2).

Якщо початкова швидкість дорівнює нулю $\psi(x)=0$, то формула Д'Аламбера набуде вигляду

$$u(x,t)=\frac{\Phi(x+at)+\Phi(x-at)}{2}, \qquad (5)$$

Розглянемо задачу про коливання скінченної струни довжини L з закріпленими кінцями, початковим зміщенням $u(x,0)=\varphi(x)$ $(0\le x\le L)$ і нульовою початковою швидкістю $u_t'(x,0)=0$. Якщо функцію $\varphi(x)$ з відрізка $[0,L]$

продовжити непарно і періодично на всю дійсну вісь за формулою (4) і одержану функцію $\Phi(x)$ підставити в (5), то отримаємо розв'язок задачі.

Нехай L=1 і $\varphi(x)=0.5-|x-0.5|$. Не забувайте, що функція $\varphi(x)$ початкового профілю повинна обертатися в нуль на кінцях відрізка $[0, L]$. Покладемо також a=1. Створимо модель, в якій креслиться графік струни в деякий момент часу, а також будується анімація коливань.

Приклад. Створіть порожню модель.

File->New->Blank Model

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `L=1, w=2*L, a=1 і T= 2*L/a` (`T` період коливань).

Ми вже зазначали, що функції системи мають опцію періодичного продовження. При чому ділянку періоду можна задавати параметрами. Отже, щоб застосувати формули (4), потрібно побудувати періодичну пилкоподібну функцію `stc` та періодичну знакову функцію `psign`, які матимуть період `w`.

Global Definitions : RMB > Functions – Analytic

Function name = `stc`

Definition –**Expression, Arguments** = `abs(x),x` (*два поля*)

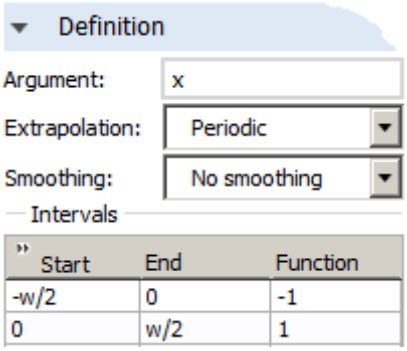
Periodic Extension – **Make periodic** = `True`

– **Lower limit, Upper limit** = `-w/2,w/2` (*два поля*)

Global Definitions : RMB > Functions – Piecewise

Function name = `psign`

Заповніть розділ **Definition** ПВ вузла функції так, як зображено на наступному рисунку.



Задайте функцію $\varphi(x)$, яка буде представляти початковий профіль струни.

Global Definitions : RMB > Functions – Analytic

Function name = `phi`

Definition –**Expression, Arguments** = `0.5-abs(x-0.5),x` (*два поля*)

Використовуючи формулу (4), створіть функцію $F(x)$, яка представлятиме непарне періодичне продовження функції $\varphi(x)$.

Global Definitions : RMB > Functions – Analytic

Function name = `F`

Definition – **Expression, Arguments** = `phi(stc(x))*psign(x),x`

За формулою (5) побудуйте функцію розв'язку $u(x,t)$.

Global Definitions : RMB > Functions – Analytic

Function name = `u`

`Definition` – **Expression, Arguments** = `(F(x+a*t)+F(x-a*t))/2,x,t`

Тепер 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 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(0,0.05,T)`

Виконайте обчислення.

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 всі функції.

Перейдемо до побудови графіків. Створіть одновимірну графічну групу.

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

1D Plot Group 1:

`Data` – **Data set** = `Study 1/Solution 1`

– **Time selection** = `From list`

– **Time(s)** = `0.25`

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

– **x minimum, x maximum** = `0,1`

– **y minimum, y maximum** = `-0.5,0.5`

`Legend` – **Show legends** = `True`

Зверніть увагу на те, що при виборі джерела даних `Study 1/Solution 1` і режиму введення моментів часу зі списку (**Time selection**=`From list`), в розділі `Data` з'являється список **Time (s)** з моментами часу, для яких будуватиметься графік . Ви можете вибрати один або декілька моментів, утримуючи клавішу CTRL і клацаючи мишею по елементам списку. Включення «легенди» (**Show legends**=`True`) дозволить відобразити на графіку обраний момент часу.

Оскільки графічна група **1D Plot Group 1** буде використовуватися для побудови анімації, то в розділі `Axis` ми визначили спільний для всіх моментів часу діапазон зміни просторових координат $0 \le x \le 1, -0.5 \le y \le 0.5$.

В графічній групі **1D Plot Group 1** створіть вузол лінійного графіка.

1D Plot Group 1-> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

Data – Data set = From Parent

Selection – Selection, <> = Manual,1 (*відрізок*)

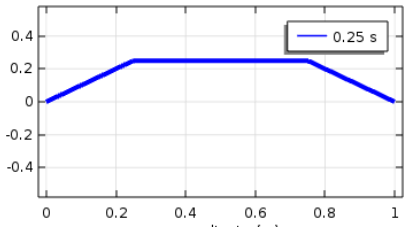
y-Axis Data – Expression = u(x,t)

x-Axis Data – Parameter, Expression = Expression,x (*два поля*)

Coloring and Style – Color, Width =Blue,3 (*два поля*)

Legends – Show legends = True

LMB > Plot



Нагадаємо, що графік побудований для моменту часу 0.25, обраного в списку **Time (s)** ПВ вузла **1D Plot Group 1**.

Існує можливість відкласти на графіку параметр часу вздовж просторової вісі. Це робиться за допомогою джерела даних Parametric Extrusion 1D.

Results -> Data Set - More Data Sets-> Parametric Extrusion 1D

Data Sets - Parametric Extrusion 1D 1:

Data – Data set = Study 1/Solution 1

Data – Time selection, Time (s) = Manual; range(1,2,41) (*два поля*)

Тепер побудуйте графік, який використовує це джерело даних. В гілку **Results** додайте графічну групу 2D Plot Group з вузлом типу Line.

Results -> Plot Group - 2D Plot Group

2D Plot Group 2:

Data– Data set = Parametric Extrusion 1D 1

Plot Settings – Plot data set edges = False

2D Plot Group 2-> Add Plot – Line

2D Plot Group 2 - Line 1 :

Data – Data set = From Parent

Expression – Expression =u(x,t)

Coloring and Style – Line type =Tube

– Tube radius expression =0.01

– Radius scale factor =True,1

– Coloring, Color =Uniform, Blue (*два поля*)

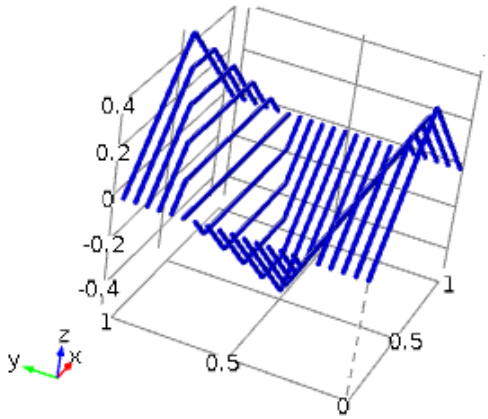
У вузол **Line 1** додайте підвузол Height Expression.

2D Plot Group 2-> **Attributes -** Height Expression

2D Plot Group 2 – Line 1 – Height Expression 1 :

Axis – Scale factor = True, 1 (*два поля*)

LMB > Plot



На цьому рисунку напрям 'y' імітує вісь часу. Ламані, що побудовані при різних значеннях $y \equiv t$, зображають форму струни в різні моменти часу.

Тепер побудуйте анімацію коливань струни.

`Results` –> Animation –> Player

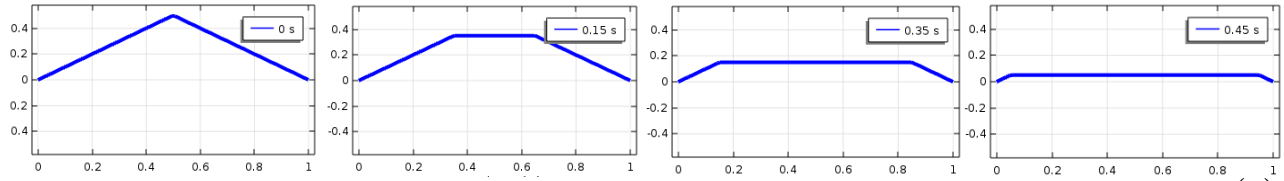
Animation 1:

`Scene` – **Subject** = `1D Plot Group 1`

`Frames` – **Frame selection** = `All`

`Playing` – **Repeat** = `True`

Згенеруйте кадри, натиснувши на кнопку Generate Frame, і запустіть анімацію кнопкою Play. На наступному рисунку показано форми струни в початковий і декілька подальших моментів часу.



Наведемо кілька прикладів коливань, вибираючи різні початкові функції $\varphi(x)$. Для цього не потрібно створювати нову модель. Достатньо змінити функцію $\varphi(x)$. Можна не видаляти колишній вираз функції, а створити новий вузол з новим виразом. Але тоді старий вузол слід деактивувати.

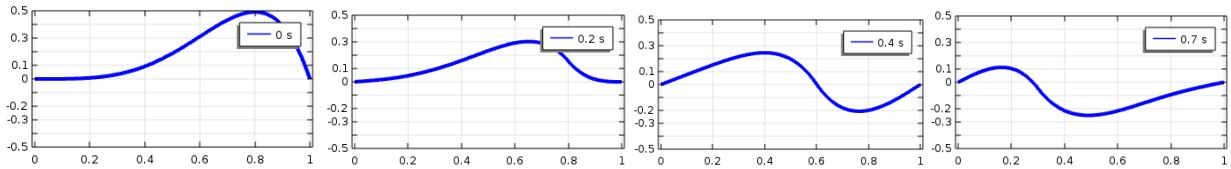
Деактивуйте вузол **Global Definitions - Analitic 2 (phi)**, натиснувши клавішу `F3`, і створіть новий вузол функції `phi`.

`Home` -> **`Definitions`** - `Finctions` -> **`Global`** - `Analitic`

Function name = `phi`

`Definition` – **Expression, Arguments** = `6*x^4*(1-x),x`

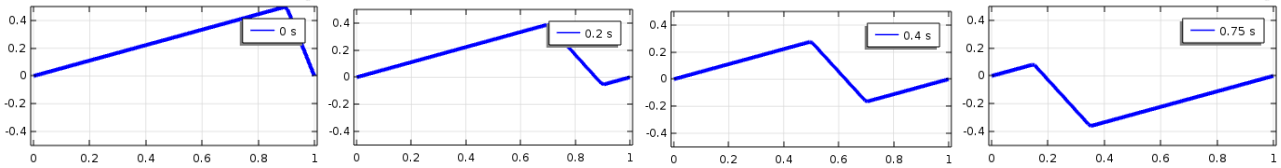
Потім виконайте команду Compute. Створена раніше модель буде повністю переобчислена з новою функцією `phi`. Як наслідок, зміняться графіки у вузлах **1D Plot Group 1** і **2D Plot Group 2**. Якщо запустити анімацію, то ви побачите рух струни, який визначається новим початковим профілем. На наступному рисунку зображена форма струни в різні моменти часу (час вказано на легендах).



Наведемо ще один приклад з іншою початковою умовою. Деактивуйте вузол, що містить функцію `phi`, натиснувши клавішу `F3`. Створіть новий вузол для цієї функції, в якому задайте вираз

```
phi(x)= 2.5-20/9*x-25/9*abs(x-0.9)
```

Виконайте команду Compute і запустіть анімацію. Нижче показана форма струни в моменти часу, вказані на графіках..



На попередніх рисунках ліві графіки наведені для моменту часу t=0, тобто на них показані початкові профілі струн.

Вправа. Обчисліть задачу для початкових профілів струни

```
phi(x)= 0.5*abs(x-1/3)-abs(x-2/3)+0.5*abs(x-1);
phi(x)= sqrt(x)*(1-x);
phi(x)= 1-2/3*x-4/3*abs(x-3/4).
```

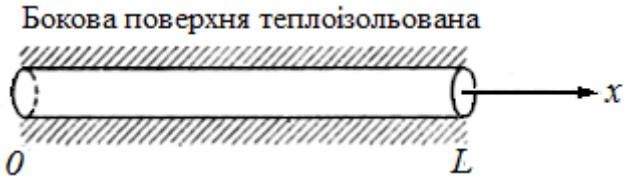
Ви можете створити інші початкові профілі і для них протестувати рішення. Але пам'ятайте, що наша модель побудована тільки для початкових функцій $\varphi(x)$, які обертаються в нуль на кінцях відрізка $[0, L]$, тобто повинно бути $\varphi(0)=0, \varphi(L)=0$.

2.3.12. Одновимірна теплопровідність.

Задача поширення тепла в тонкому однорідному необмеженому стрижні, бокова поверхня якого теплоізольована (тепло може поширюватися тільки уздовж осі), математично формулюється наступним чином [2,10]: знайти обмежену функцію $u(x,t)$, яка задовольняє рівнянню теплопровідності

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} \quad (t>0, -\infty<x<\infty) \tag{1}$$

і початковій умові $u(x,0)=\varphi(x)$ $(-\infty<x<\infty)$, де $u(x,t)$ температура точки x стрижня в момент часу t (стрижень тонкий, і температура точок будь-якого поперечного перерізу x однакова).



Відомо, що розв'язок цієї задачі репрезентується інтегралом Пуассона

$$u(x,t) = \frac{1}{2a\sqrt{\pi t}} \int_{-\infty}^{\infty} \varphi(\xi) e^{-\frac{(\xi-x)^2}{4a^2 t}} d\xi \tag{2}$$

Якщо в цьому інтегралі зробити заміну $\tau = \dfrac{\xi - x}{2a\sqrt{t}}$, то отримаємо

$$u(x,t) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{\infty} \varphi\left(x + 2a\sqrt{t}\,\tau\right) e^{-\tau^2} d\tau . \qquad (3)$$

Розглянемо нескінченний стрижень з початковою температурою $\varphi(x)$ всюди рівною нулю, за винятком відрізка [-1,1], де вона дорівнює одиниці, тобто

$$u(x,0) = \varphi(x) = \begin{cases} 1, -1 \le x \le 1 \\ 0, x < -1 \vee x > 1 \end{cases}.$$

При чисельних розрахунках ми не можемо безпосередньо використати формулу (3) тому, що вона містить нескінченні межі інтегрування. Але оскільки функція $\varphi(x)$ обмежена і фінітна, то в (3) можна взяти досить широкий, але обмежений відрізок інтегрування. Відрізка `[-5,5]` буде достатньо. Дійсно

$$\int_{-\infty}^{\infty} \varphi\left(x + 2a\sqrt{t}\,\tau\right) e^{-\tau^2} d\tau = \int_{-\infty}^{-A} + \int_{-A}^{A} + \int_{A}^{+\infty} .$$

Оскільки $|\varphi(x)| \le 1$, то

$$\left| \int_{A}^{+\infty} \varphi\left(x + 2a\sqrt{t}\,\tau\right) e^{-\tau^2} d\tau \right| \le \int_{A}^{+\infty} \left| \varphi\left(x + 2a\sqrt{t}\,\tau\right) \right| e^{-\tau^2} d\tau \le \int_{A}^{+\infty} e^{-\tau^2} d\tau = \frac{1}{2}\sqrt{\pi}\,\mathrm{Erfc}(A)$$

де $\mathrm{Erfc}(x)$ - додаткова функція помилок. Аналогічна нерівність має місце для інтеграла $\int_{-\infty}^{-A}$. Оскільки $\mathrm{Erfc}(5) < 10^{-11}$, то заміна нескінченної ділянки інтегрування на скінченний відрізок [-5,5]

$$\int_{-\infty}^{\infty} \varphi\left(x + 2a\sqrt{t}\,\tau\right) e^{-\tau^2} d\tau \approx \int_{-5}^{5} \varphi\left(x + 2a\sqrt{t}\,\tau\right) e^{-\tau^2} d\tau$$

призводить до похибки обчислень менше за 10^{-11}. Таким чином, в якості розв'язку можна розглядати наступну функцію

$$u(x,t) \approx \frac{1}{\sqrt{\pi}} \int_{-A}^{A} \varphi\left(x + 2a\sqrt{t}\,\tau\right) e^{-\tau^2} d\tau , \qquad (4)$$

де `A = 5`. Покладемо також a=1.

Побудуємо модель, в якій креслиться графік розподілу температури всередині стрижня в деякий момент часу, а також будується анімація зміни цієї температури.

Приклад 1. Створіть порожню модель.

File->New->Blank Model

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `A=5, a=1 i hw=3` (півширина відрізка спостереження).

Задайте функцію $\varphi(x)$, яка буде представляти початкову температуру стрижня.

Global Definitions : RMB > Functions – Analytic

Function name = phi

Definition –**Expression, Arguments** = x>=-1 && x<=1,x (*два поля*)

Створіть змінну моделі, яка буде обчислюватися за формулою (4).

Home -> Definitions – Variables -> Global Variables

Global Definitions – Variables 1:

Variables – **Name** = val

Variables – **Expression** =

```
integrate(phi(x[1/m]+2*sqrt(t[1/s])*tau)*
                exp(-tau^2),tau,-A,A)/sqrt(pi)
```

Variables

Name	Expression	Unit	Description
val	integrate(phi(x[1/m]+2*sqrt(t[1/s])*tau)*exp(-tau^2),tau,-A,A)/sqrt(pi)		

Нагадаємо, що ім'я val вводиться в таблиці змінних в колонці **Name**, а в колонці **Expression** вводиться вираз. Одиниці вимірювання, які стоять після імен просторової *x* і часової *t* змінних потрібні для того, щоб аргументи використовуваних функцій були безрозмірними. Зверніть також увагу на те, що відразу після введення формула стає помаранчевою. Це пов'язано з тим, що ми ще не створили просторову «компоненту» моделі і змінна *x* системі ще невідома. Також системі невідома часова змінна *t*, оскільки ми ще не обрали тип дослідження (study).

Створіть в моделі одновимірний компонент.

Untitled.mph : RMB > Add Component - 1D

В результаті з'явиться гілка **Component 1** з вузлом **Geometry 1**.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1:

Interval – **Left endpoint, Right endpoint** = -hw,0

Аналогічно створіть ще один інтервал [0,hw].

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1:

Interval – **Left endpoint, Right endpoint** = 0,hw

LMB > Build All Objects

Насправді робочим відрізком у нас буде [-hw,hw]. На ньому будуватиметься графік розподілу температури в нескінченному стрижні. Однак для подальших цілей нам зручно створити цей відрізок з двох половинок. Половинки з'єднуються в один відрізок. Така поведінка визначається значенням поля **Action** в створеному автоматично вузлі **Geometry 1 - Form Union**. Якщо не передбачено інше, то геометричні об'єкти вузла **Geometry 1** об'єднуватимуться в єдиний об'єкт.

Генеруйте СкЕ сітку зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Додайте в модель вузол 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** =

0 0.001 0.005 0.01 0.05 range(0.1,0.1,2)

Тут для більш гладкої анімації початок діапазону містить нерівномірно розподілені відліки часу.

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** буде створено джерело даних **Study 1/Solution 1**.

Створіть одновимірну графічну групу.

Results –> Plot Group - 1D Plot Group

1D Plot Group 1 :

Data – **Data set** = Study 1/Solution 1

– **Time selection** = From list

– **Time(s)** = 0.05 (*вибір зі списку*)

Axis –**Manual axis limits**= True

–**x minimum, x maximum** = -3,3

–**y minimum, y maximum** = -0.1,1.1

При виборі в списку **Time selection** значення From list в розділі **Data** з'являється список **Time (s)** вибору моменту часу, для якого будуватиметься графік. Оскільки графічна група **1D Plot Group 1** буде використовуватися для побудови анімації, то в розділі **Axis** ми визначили загальний для всіх моментів часу діапазон просторових змінних $-3 \le x \le 3, -0.1 \le y \le 1.1$.

В графічної групі **1D Plot Group 1** створіть вузол Line Graph.

1D Plot Group 1-> Add Plot – Line Graph

1D Plot Group 1 - Line Graph 1:

Data – **Data set** = From parent

Selection – **Selection, <>** = Manual;1,2 (*обидва відрізки*)

y-Axis Data – **Expression** = val

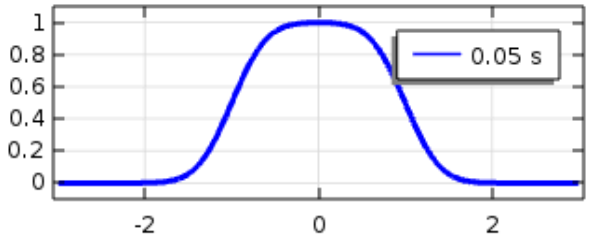
Тут ми ввели ім'я змінної val, для якої у вузлі **Global Definitions-Variables 1** створено вираз, що реалізує обчислення за формулою (4).

x-Axis Data – **Parameter, Expression** = Expression,x (*два поля*)

Coloring and Style – **Color, Width** = Blue,3 (*два поля*)

Legends – **Show legends** = True

LMB > Plot



Побудуйте анімацію зміни температури в стрижні.

Results –> Animation –> Player

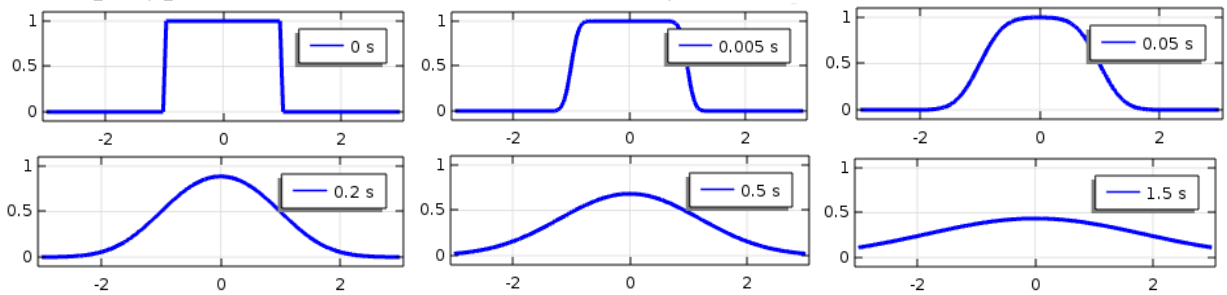
Animation 1:

Scene – **Subject** = `1D Plot Group 1`

Frames – **Frame selection** = `All`

Playing – **Repeat** = `True`

Згенеруйте кадри, натиснувши на кнопку Generate Frame, і запустіть анімацію кнопкою Play. На наступному рисунку показані графіки температури в початковий і декілька наступних моментів часу.

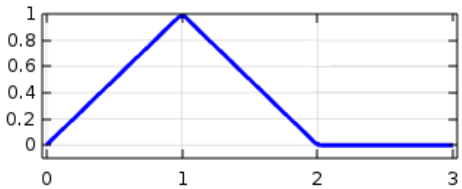


Збережіть файл, але не закривайте його. ■

Розв'яжемо задачу поширення тепла в напівнескінченному стрижні $0 \le x < \infty$ з нульовою температурою на лівому кінці $u(0,t)=0$ і початковою температурою $u(x,0)=\varphi(x)$ $(x \ge 0)$. Для цього можна взяти початкову функцію $\varphi(x)$, задану тільки на додатній півосі, і продовжити її непарно на всю вісь, отримавши функцію $\Phi(x)$. Підстановка $\Phi(x)$ в (2) або (3) дає розв'язок рівняння теплопровідності (1), яке при $x>0$ має задане початкове значення $\varphi(x)$, і на лівому кінці в будь-який момент часу дорівнює нулю [2]. Заміна нескінченних меж інтегрування на скінченні так, як в формулі (4), і використання лише додатних значень *x*, дасть наближений розв'язок задачі теплопровідності для напівнескінченного відрізка.

Розглянемо приклад, коли початкова температура $\varphi(x)$ має вигляд

$$\varphi(x)=\begin{cases} x & ,0 \le x \le 1 \\ 2-x, & 1 \le x \le 2 \\ 0 & , \ x \ge 2 \end{cases}$$



Приклад 2. Створіть в поточному файлі функцію `PHI(x)`, яка є непарним продовженням $\varphi(x)$ на від'ємну піввісь.

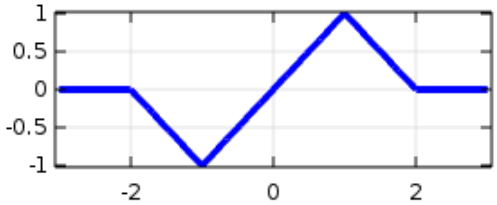
`Home` –> Definitions – Functions –> Global – Analytic

Function name = `PHI`

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

Definition – **Arguments** = `x`

В розділі **Plot Parameters** використайте відрізок `[-hw,hw]`, і командою Plot побудуйте графік функції `PHI(x)`, яка є непарною і для додатних аргументів *x* збігається з функцією $\varphi(x)$.



Відкрийте ПВ вузла **Global Definitions-Variables 1**, і в таблицю додайте нову змінну VAL з наступною формулою:

```
VAL= integrate(PHI(x[1/m]+2*a*sqrt(t[1/s])*tau)*
                    exp(-tau^2),tau,-A,A)/sqrt(pi)
```

Від val вона відрізняється тим, що використовує в інтегралі непарну функцію PHI(x). Потім виконайте обчислення.

Study 1 : LMB > Compute

Створіть нову одновимірну графічну групу, продублювавши першу.

1D Plot Group 1 : RMB > Duplicate

1D Plot Group 2 :

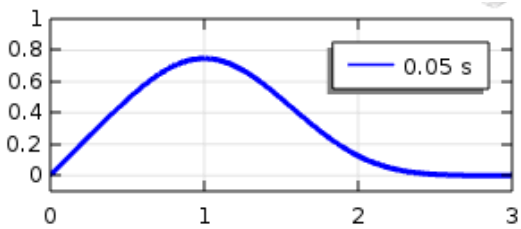
Axis–x minimum, x maximum = 0,3

1D Plot Group 2 – Line Graph 1:

Selection – **Selection,** <> = Manual;2 (*правий відрізок*)

y-Axis Data – **Expression** = VAL

LMB > Plot

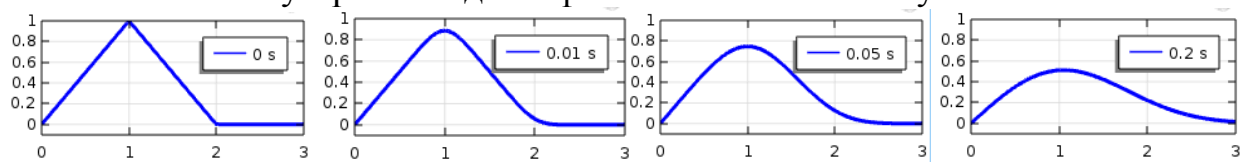


Побудуйте анімацію зміни температури в стрижні.

Results –> Animation –> Player

Заповніть ПВ створеного вузла **Animation 2** так само, як і раніше, використовуючи для зображень графічну групу **1D Plot Group 2**. Згенеруйте кадри анімації командою Generate Frame і запустіть процес кнопкою Play .

На наступному рисунку показані графіки розподілу температури в напівнескінченномустрижні в деякі фіксовані моменти часу.



Збережіть файл, але не закривайте його. ■

Розв'яжемо задачу поширення тепла в стрижні довжиною L, кінці якого підтримуються при нульовій температурі. Для цього можна використати формули (3), (4) і методику непарного періодичного продовження початкової функції, викладену в попередньому прикладі.

Нехай початкова температура $\varphi(x)$ стрижня задана на відрізку $[0, L]$, і дорівнює нулю на його кінцях. Продовжимо функцію $\varphi(x)$ непарно відносно точки x=0 на відрізок $[-L,0]$, а потім періодично на всю вісь з періодом $2L$.

Якщо отриману функцію $\Phi(x)$ підставити в формулу (3), то результуюча функція $u(x,t)$ буде задовольняти рівнянню (1). При t=0 і $0<x<L$ вона буде збігатися з початковою функцією $\varphi(x)$, а також обертатиметься на нуль в точках $x=0$ і $x=L$ при будь-якому $t>0$ [2]. Іншими словами, функція $u(x,t)$ буде представляти розв'язок крайової задачі для рівняння (1) з початковою умовою $u(x,0)=\varphi(x)$ $(0<x<L)$ і граничними умовами $u(0,t)=0, u(L,t)=0$ $\forall t>0$.

Непарне періодичне продовження $\Phi(x)$ функції $\varphi(x)$ з відрізка $[0,L]$ на всю дійсну вісь можна побудувати за формулою (4) попереднього параграфа 2.3.11

$$\Phi(x)=psign(x,2L)\cdot\varphi(stc(x,2L)).$$

Тут $stc(x,w)$ – пилкоподібна періодична по змінній x функція з періодом w, яка на відрізку $-w/2\le x\le w/2$ збігається з функцією $|x|$, а поза відрізком повторюється періодично. Функція $psign(x,w)$ є періодичною знаковою функцією, яка на півінтервалі $[-w/2,0)$ дорівнює -1, на ділянці $[0,w/2)$ дорівнює 1, і далі повторює свої значення періодично з періодом w.

Підставляючи $\Phi(x)$ в (3), здобуваємо функцію

$$u(x,t)=\frac{1}{\sqrt{\pi}}\int_{-\infty}^{\infty} psign\left(x+2a\sqrt{t}\,\tau,2L\right)\cdot\varphi\left(stc\left(x+2a\sqrt{t}\,\tau,2L\right)\right)e^{-\tau^2}d\tau\,.$$

В результаті ми отримуємо розв'язок поставленої задачі, розглядаючи $u(x,t)$ тільки на відрізку $0\le x\le L$. Дійсно $u(x,t)$ задовольняє рівнянню теплопровідності (1), обертається на нуль на кінцях відрізка $[0, L]$, і на цьому відрізку в нульовий момент часу збігається з функцією $\varphi(x)$.

В припущенні, що функція $\varphi(x)$ обмежена при $0\le x\le L$ і, враховуючи (4), можна записати розв'язок у вигляді

$$u(x,t)\approx\frac{1}{\sqrt{\pi}}\int_{-A}^{A} psign\left(x+2a\sqrt{t}\,\tau,2L\right)\cdot\varphi\left(stc\left(x+2a\sqrt{t}\,\tau,2L\right)\right)e^{-\tau^2}d\tau\,,$$

де `A` досить велике додатне число, залежне від бажаної точності.

Нехай L=3, a=1 і $u(x,0)=\varphi(x)=L/2-|L/2-x|$. З огляду на попереднє обговорення, достатню точність гарантує `A=5`.

Приклад 3. В поточному файлі в ПВ вузла **Global Definitions - Parameters** додайте параметр `L=hw`.

Потім створіть функцію $\mathrm{psign}(x)$. В попередньому параграфі ми реалізували її за допомогою Piecewise періодичної функції. Тут, для різноманітності, створимо її як аналітичну функцію.

`Home`-> Definitions – Finctions -> Global - Analitic

Function name = `psign`

Definition – **Expression** = `(x<L)-(x>=L)`

– **Arguments** = `x`

Periodic Extension – **Make periodic** = `True`

– **Lower limit, Upper limit** = `0,2*L` (*два поля*)

Останні дві опції визначають, що створюється періодична функція, і її значення з відрізка [0, 2*L*] на всю вісь повторюються з періодом 2*L*.

Створіть функцію $stc(x, w)$.

Global Definitions : RMB > Functions – Analytic

Function name = `stc`

Definition –**Expression, Arguments** = `abs(x),x` (*два поля*)

Periodic Extension – **Make periodic** = `True`

– **Lower limit, Upper limit** = `-L,L` (*два поля*)

Аналогічно створіть функцію `phif(x)`, яка представлятиме початковий профіль температури, і функцію `PHIF(x)` – її непарне періодичне продовження.

```
phif(x)= L/2-abs(L/2-x)
PHIF(x)= psign(x)*phif(stc(x))
```

Для стислості ми не наводимо вміст полів в панелях властивостей цих функцій.

Відкрийте ПВ вузла **Global Definitions - Variables 1**, і в таблицю додайте нову змінну `F` з наступною формулою:

```
F= integrate(PHIF(x[1/m]+2*a*sqrt(t[1/s])*tau)*
                    exp(-tau^2),tau,-A,A)/sqrt(pi)
```

Від змінних `val` і `VAL` вона відрізняється тим, що використовує в інтегралі непарну періодичну функцію `PHIF(x)`.

Виконайте обчислення.

Study 1 : LMB > Compute

Створіть нову одновимірну графічну групу, продублювавши вузол **1D Plot Group 2**.

1D Plot Group 2 : RMB > Duplicate

1D Plot Group 3 :

Data – **Time selection** = `Manual`

Data – **Time(s) (1 - 25)** = `1 4 5 6 7 8 9 11 15`

Axis – **x minimum, x maximum** = `0,3`

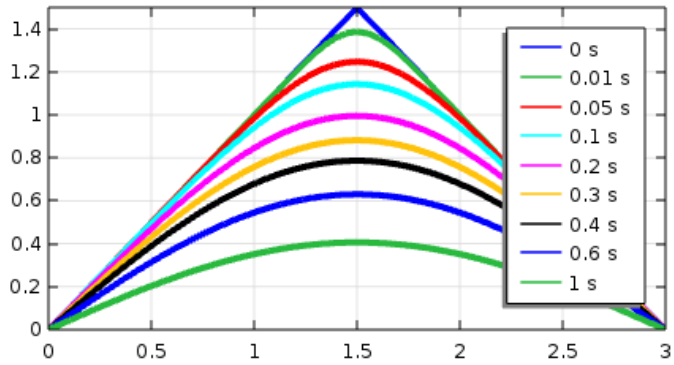
– **y minimum, y maximum** = `0,L/2`

1D Plot Group 3 – Line Graph 1:

y-Axis Data – **Expression** = `F`

Coloring and Style – **Color, Width** = `Cycle,3` (*два поля*)

LMB > Plot



Аналогічно до попереднього створіть анімацію.

2.3.13. Остигання квадратної платівки і куба.

Розглянемо тонку однорідну прямокутну пластину, контур якої підтримується при нульовій температурі. Теплообмін між плоскими гранями пластини і навколишнім середовищем відсутній. Математична задача поширення тепла в такій пластині зводиться до розв'язання двовимірного рівняння теплопровідності з відповідними початковою і граничними умовами.

Відомо [10], що розв'язання задачі теплопровідності в необмеженій на всі боки пластині, яка має початкову температуру $u(x,y,0)=\psi(x,y)$, репрезентується інтегралом Пуассона

$$u(x,y,t)=\frac{1}{4\pi a^2 t}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\psi(\xi,\eta)e^{-\frac{(x-\xi)^2+(y-\eta)^2}{4\pi a^2 t}}d\xi\, d\eta\,.$$

Коли початкова температура може бути представлена у вигляді добутку $\psi(x,y)=\varphi(x)\cdot\phi(y)$, подвійний інтеграл розпадається на добуток двох інтегралів

$$u(x,y,t)=\frac{1}{4\pi a^2 t}\int_{-\infty}^{\infty}\varphi(\xi)e^{-\frac{(x-\xi)^2}{4\pi a^2 t}}d\xi\ \times\ \int_{-\infty}^{\infty}\phi(\eta)e^{-\frac{(y-\eta)^2}{4\pi a^2 t}}d\eta\,.$$

Якщо в них зробити заміни $\tau=\dfrac{\xi-x}{2a\sqrt{t}}$ і $\nu=\dfrac{\eta-y}{2a\sqrt{t}}$, то отримаємо

$$u(x,y,t)=\frac{1}{\pi}\int_{-\infty}^{\infty}\varphi\left(x+2a\sqrt{t}\,\tau\right)e^{-\tau^2}d\tau\ \times\ \int_{-\infty}^{\infty}\phi\left(y+2a\sqrt{t}\,\nu\right)e^{-\nu^2}d\nu\,.$$

Відповідно до формули (4) попереднього параграфа 2.3.12, ці інтеграли з нескінченними межами з достатньою точністю можна перетворити до визначених скінченних інтегралів. У разі прямокутної пластинки з нульовою температурою на контурі кожну з функцій $\varphi(x)$, $\phi(y)$ слід продовжити непарно і періодично на всю вісь по формулі (4) параграфа 2.3.11. В результаті, розв'язок задачі теплопровідності в прямокутній пластині розміром $L_1\times L_2$, початковою температурою $u(x,y,0)=\varphi(x)\cdot\phi(y)$, і нульовою температурою на контурі прямокутника $u\big|_{\Gamma}=0$, може бути отримано за формулою

$$\begin{aligned}u(x,y,t)\approx\frac{1}{\sqrt{\pi}}\int_{-A}^{A}\mathrm{psign}\left(x+2a\sqrt{t}\,\tau,2L_1\right)\cdot\varphi\left(stc\left(x+2a\sqrt{t}\,\tau,2L_1\right)\right)e^{-\tau^2}d\tau\ \times\\ \times\frac{1}{\sqrt{\pi}}\int_{-A}^{A}\mathrm{psign}\left(y+2a\sqrt{t}\,\tau,2L_2\right)\cdot\phi\left(stc\left(y+2a\sqrt{t}\,\tau,2L_2\right)\right)e^{-\tau^2}d\tau\end{aligned}\qquad(1)$$

Аналогічно може бути розв'язана тривимірна задача охолодження паралелепіпеда розміром $L_1\times L_2\times L_3$. Якщо його початкова температура $\psi(x,y,z)$ представлена у вигляді добутку $\psi(x,y,z)=\varphi(x)\cdot\phi(y)\cdot\omega(z)$, а бокові грані підтримуються при нульовій температурі, то розв'язок $u(x,y,z,t)$ можна обчислювати за формулою

$$u(x,y,z,t)\approx\frac{1}{\sqrt{\pi}}\int_{-A}^{A}\mathrm{psign}\left(x+2a\sqrt{t}\,\tau,2L_1\right)\cdot\varphi\left(stc\left(x+2a\sqrt{t}\,\tau,2L_1\right)\right)e^{-\tau^2}d\tau\times$$

$$\times\frac{1}{\sqrt{\pi}}\int_{-A}^{A}\mathrm{psign}\left(y+2a\sqrt{t}\,\tau,2L_2\right)\cdot\phi\left(stc\left(y+2a\sqrt{t}\,\tau,2L_2\right)\right)e^{-\tau^2}d\tau\times \qquad (2)$$

$$\times\frac{1}{\sqrt{\pi}}\int_{-A}^{A}\mathrm{psign}\left(z+2a\sqrt{t}\,\tau,2L_3\right)\cdot\omega\left(stc\left(z+2a\sqrt{t}\,\tau,2L_3\right)\right)e^{-\tau^2}d\tau$$

Розв'яжемо задачу поширення тепла в квадратній пластині $[0,1]\times[0,1]$, у якої границя підтримуються при нульовій температурі. Нехай початкова температура $\psi(x,y)=1$. Це означає, що $\psi(x,y)=\varphi(x)\cdot\phi(y)$ при $\varphi(x)=1$ і $\phi(y)=1$. Їх непарні продовження з відрізка [0,1] на всю вісь, відповідно до формул (4) параграфа 2.3.11, дорівнюватимуть `psign(x,2)` і `psign(y,2)`.

Приклад 1. Створіть порожню модель.

File->New->Blank Model

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `A=5,L=1` (довжина сторони квадрата),`a=1` і `w=1[m]`. Тут `w=1[m]` (з розмірністю довжини) є допоміжним параметром.

Зауваження. Далі в формулах параметр `w` буде замінятися на змінні *x*, *y* або *z*, які мають розмірність довжини. Тому для сумісності ми зразу створили цей параметр з такою ж розмірністю.

Створіть функцію $\mathrm{psign}(x)$.

Home-> Definitions – Finctions -> Global - Analitic

Function name = `psign`

Definition – **Expression** = `(x<L)-(x>=L)`

– **Arguments** = `x`

Periodic Extension – **Make periodic** = `True`

– **Lower limit, Upper limit** = `0,2*L` (*два поля*)

Останні дві опції визначають, що створюється періодична функція, і її значення з відрізка [0, 2*L*] на всю вісь повторюються з періодом 2*L*.

Для наочності створіть функцію `PHI(x)` за формулою:

`PHI(x)= psign(x)`

Створіть вузол глобальних змінних моделі.

Home -> Definitions – Variables -> Global Variables

В ПВ вузла **Global Definitions – Variables 1** створіть змінну `WW`.

Global Definitions – Variables 1:

Variables – **Name** = `WW`

Variables – **Expression** =

```
integrate(PHI(w[1/m]+2*a*sqrt(t[1/s])*tau)*
                exp(-tau^2),tau,-A,A,1e-2)/sqrt(pi)
```

Тут використовується необов'язковий п'ятий аргумент оператора `integrate`, який визначає відносну похибку обчислення інтеграла.

Також створіть наступні змінні.

```
WX= subst(WW,w,x)
WY= subst(WW,w,y)
WZ= subst(WW,w,z)
```

У виразі змінної `WW` ми застосували допоміжний параметр `w`. Він ніде не буде використовуватися, оскільки у виразах будуть стояти тільки змінні `WX`, `WY` і `WZ`, які отримуються з `WW` заміною `w` на «справжні» змінні *x*, *y* і *z* відповідно. Змінна `WZ` нам знадобиться пізніше під час розв'язання задачі теплопровідності в кубі.

Оператор підстановки `subst` має наступний синтаксис

```
subst(expr, exprorig1, exprsubst1,...).
```

Оператор `subst` в виразі `expr` замінює вираз `exprorig1` виразом `exprsubst1`. Замість трьох крапок можуть стояти ще пари замінних виразів.

Створіть в моделі двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

В ДМ з'явиться гілка **Component 1** з вузлом **Geometry 1**. Геометрія компонента складатиметься з одиничного квадрата.

`Geometry` **-> Primitives** – Square

Geometry 1 – Square 1:

Size – Side length = `1`

Position – Base, x, y = `Corner,0,0` (*три поля*)

Для прискорення подальших обчислень на квадраті побудуйте грубу СкЕ сітку.

Component 2 - Mesh 1 : Mesh Settings – Element size = `Coarser`

LMB > Build All

Додайте в модель `Time Dependent` вузол дослідження.

Untitled.mph (root): RMB > Add Study = `Add Study` ->

Preset Studies - Time Dependent ->X

Задайте діапазон і інтервал зміни параметра часу `t`.

Study 1 - Step 1: Time Dependent:

Study Settings –Times = `0.001 0.005 range(0.01,0.01,0.1)`

Для подальшої більш гладкої анімації початок діапазону містить нерівномірно розподілені часові відліки.

Виконайте обчислення.

Study 1 : LMB > Compute

Створіть двовимірну графічну групу з вузлом `Surface`.

`Home` **-> Results -** Add Plot Group ->2D Plot Group

2D Plot Group 1:

Data– Data set = `Sudy 1/Solution 1`

Data– Time(s) = `0.02`

Plot Settings – Plot data set edges = `False`

`2D Plot Group 1` **-> Add Plot -** Surface

2D Plot Group 1 – Surface 1:

Data – Data set = `From parent`

Expression – Expression = `WX*WY`

Range – Manual color range = `True`

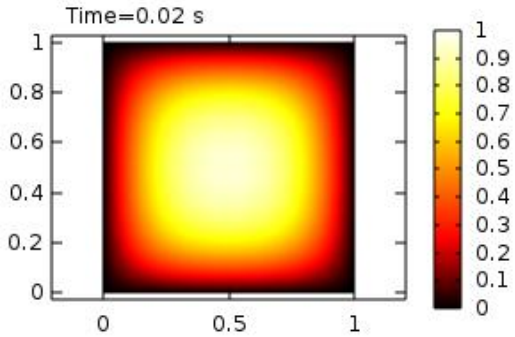
Range – Minimum, Maximum = `0,1` (*два поля*)

Coloring and Style – Coloring =`Color table`

– Color = `Thermal`

Зверніть увагу на вираз `WX*WY`, який відповідає формулі (1). Оскільки ми не можемо використовувати оператор `integrate` при створенні функцій, у вузлі **Variables 1** нами були створені змінні `WX` і `WY`, які можуть містити оператор інтегрування, і використовуються замість функцій.

Виконайте команду Plot і отримайте зображення зони пластини з функціональним забарвленням, відповідним температурі в момент часу `t=0.02`.

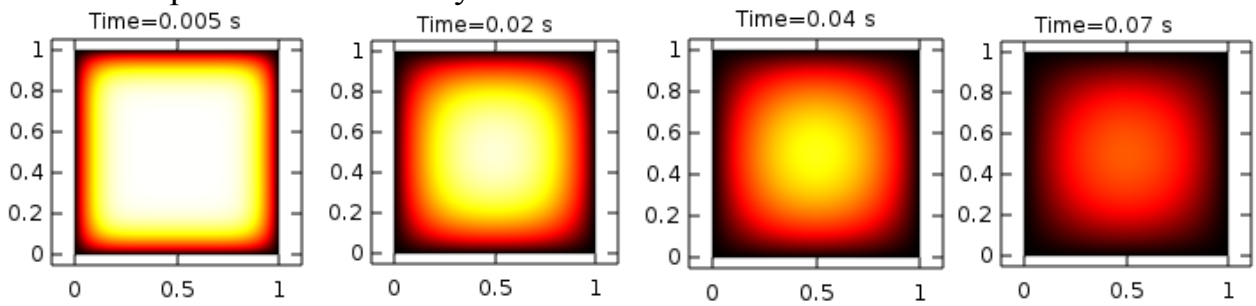


Побудуйте анімацію.

`Results` –> Animation –> Player

Заповніть ПВ створеного вузла **Animation 1** так само, як і раніше. Згенеруйте кадри анімації командою Generate Frame і запустіть процес кнопкою Play .

На наступному рисунку показані графіки розподілу температури в квадратній пластині в різні моменти часу.



Збережіть файл, але не закривайте його. ■

Розв'яжемо задачу поширення тепла в одиничному кубі, поверхня якого підтримується при нульовій температурі. Нехай початкова температура $u(x,y,z,0)=\psi(x,y,z)=1$. Це можна записати у вигляді $\psi(x,y,z)=\varphi(x)\cdot\phi(y)\cdot\omega(z)$, де $\varphi(x)=1$, $\phi(y)=1$, $\omega(z)=1$. Непарні періодичні продовження цих функцій з відрізка [0,1] на всю дійсну вісь, відповідно до формули (4) параграфа 2.3.11, будуть дорівнювати `psign(x,2)`, `psign(y,2)`,`psign(z,2)`.

Приклад 2. Продовжить роботу з поточним файлом, в який додайте тривимірний компонент.

Untitled.mph : RMB > Add Component - 3D

В ДМ з'явиться гілка **Component 2** з вузлом **Geometry 2**. Геометрія компонента складатиметься з одиничного куба.

Component 2 (comp2):

Geometry **-> Primitives** – Block

Component 2 - Geometry 2 – Block 1:

Size – Width, Depth, Height = 1,1,1 (*три поля*)

Position – Base= Corner

– x, y, z = 0,0,0 (*три поля*)

Для прискорення подальших обчислень побудуйте грубу СкЕ сітку.

Component 2 - Mesh 1 : Mesh Settings – Element size = Coarser

LMB > Build All

Переобчисліть модель.

Study 1 : LMB > Compute

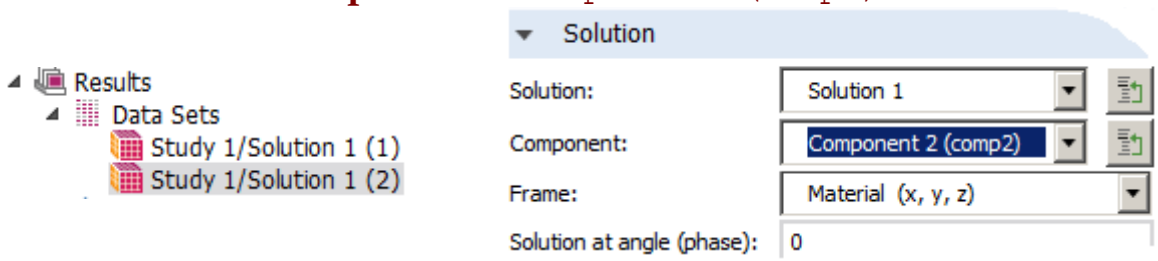
В гілці **Results – Data Sets** вузол джерела даних **Study 1/Solution 1** не зміниться. Тепер це ДД може бути використано для побудови розв'язку тривимірної задачі теплопровідності. Але для цього в його ПВ треба обрати відповідний компонент (перший або другий). Щоб не зашкодити попереднім побудовам, створимо копію цього ДД, в якій виберемо другий компонент.

Data Sets – Study 1/Solution 1 (1) : RMB > Duplicate

Одиниця в дужках автоматично додається до назви ДД **Study 1/Solution 1** в усіх вузлах, де воно використовувалося.

Data Sets – Study 1/Solution 1 (2) :

Solution – Component = Component 2(comp2)



Перейдемо до побудови зображення температурного поля в кубі. Створіть тривимірну графічну групу з вузлом Multislice.

Home **-> Results -** Add Plot Group ->3D Plot Group

3D Plot Group 2:

Data– Data set = Sudy 1/Solution 1

Data– Time(s) = 0.005

Plot Settings – Plot data set edges = False

3D Plot Group 2 : RMB > More Plots - Multislice

3D Plot Group 2 - Multislice 1 :

Data – Data set = From parent

Expression – Expression = WX*WY*WZ

Зверніть увагу на вираз WX*WY*WZ, який відповідає формулі (2).

В розділі **Multiplane Data** для **x-planes**, **y-planes**, **z-planes** вкажіть кількість площин - по одній площини кожної орієнтації.

Range – **Manual color range** = True

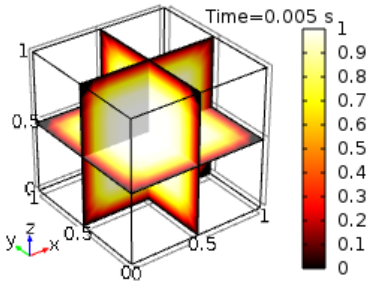
– **Minimum, Maximum** = 0,1 (*два поля*)

Coloring and Style – **Coloring** =Color table

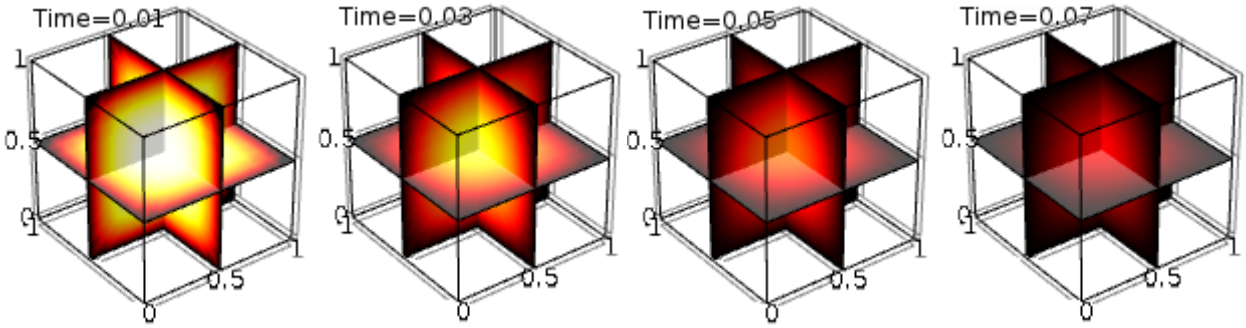
– **Color** = Thermal

LMB > Plot

Виконайте команду Plot і отримайте зображення трьох взаємно ортогональних площин, що проходять через центр куба, з функціональним забарвленням, відповідним температурі куба в момент часу t=0.005



Побудуйте анімацію процесу остигання куба. Майте на увазі, що обчислення триватимуть деякий час (на комп'ютері автора півтори хвилини). На наступному рисунку показані графіки розподілу температури в кубі в різні моменти часу.



2.3.14. Хвиля Релея.

Англійський фізик Релей показав, що в пружному півпросторі можуть існувати спеціальні хвилі, які він назвав поверхневими. У таких хвиль амплітуда коливань швидко зменшується з віддаленням від поверхні. Одне з таких плоских хвильових рухів може бути описано за допомогою наступних функцій [9]

$$U(x,y,t)=C\left(-e^{-\alpha y}+c\,e^{-\beta y}\right)\sin\left(\xi x+pt\right)$$
$$V(x,y,t)=C\,a\left(-e^{-\alpha y}-d\,e^{-\beta y}\right)\cos\left(\xi x+pt\right) \qquad (1)$$

де U і V представляють подовжнє і поперечне зміщення точок середовища відносно положення рівноваги. Тут $\alpha=a\xi$, $\beta=b\xi$ і a,b,c,d деякі сталі, залежні від властивостей середовища, C - довільна стала. Ось Y спрямована вглиб середовища. Якщо вісь Y направити вертикально вгору (назовні), то в формулах (1) слід поміняти знак перед змінною y і перед функцією V.

В моделі поточного прикладу ми побудуємо двовимірну область з внутрішньої координатної сіткою, деформованою в момент часу *t* відповідно до формул (1), а також змоделюємо рух цієї сітки, який імітуватиме рух середовища. Для спрощення відразу покладемо $\xi = 1$.

Приклад. Створіть порожню модель.

File->New->Blank Model

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `c=0.5, d=2, p=1, C=0.25, a=1, b=0.4`.

Створіть функцію `U(x,y,t)`..

`Home`-> Definitions – Finctions -> Global - Analitic

Function name = `U`

Definition–Expression=`C*(-exp(-a*y)+c*exp(-b*y))*sin(x+p*t)`

Definition – **Arguments** = `x,y,t`

Аналогічно створіть другу функцію (1) `V`:

`V(x,y,t)=C*a*(-exp(-a*y)-d*exp(-b*y))*cos(x+p*t)`

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Його геометрія складатиметься з одного прямокутника.

`Geometry` -> **Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

Size – **Width, Height** = `10,5` (*два поля*)

Position – **Base, x, y** = `Corner,-5,-5` (*три поля*)

LMB > Build All Objects

Побудуйте в геометрії чотирикутну сітку.

`Mesh` -> **Generators** – Mapped

Mesh 1 – Mapped 1:

Domain Selection – **Geometric entity level** = `Domain`

Domain Selection – **Selection**, <> = `Manual,1` (*прямокутник*)

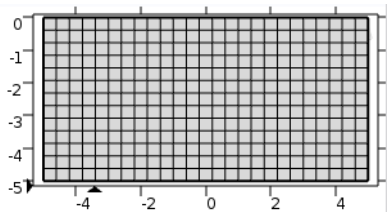
Задайте розміри клітинок сітки.

Mesh 1 – Size:

Element Size – **Custom**= `True`

Element Size Parameters – **Maximum element size**= `0.4`

LMB > Build All



Додайте в модель `Time Dependent` вузол дослідження.

Untitled.mph (root): RMB > Add Study = `Add Study` ->

Preset Studies - Time Dependent ->X

Задайте діапазон і інтервал зміни параметра часу `t`.

Study 1 - Step 1: Time Dependent:

Study Settings –**Times** = `range(0,0.2,2*pi)`

Виконайте обчислення.

Study 1 : LMB > Compute

Створіть двовимірну графічну групу з вузлом `Surface`.

`Home` **-> Results -** Add Plot Group ->2D Plot Group

2D Plot Group 1:

Data– **Data set** = `Sudy 1/Solution 1`

Data– **Time(s)** = `0.6`

Plot Settings – **Plot data set edges** = `False`

`2D Plot Group 1` **-> Add Plot -** Surface

2D Plot Group 1 – Surface 1:

Data– **Data set** = `From parent`

Expression – **Expression** = `1`

Coloring and Style – **Coloring, Color** = `Uniform, Black` (*два поля*)

– **Wireframe** = `True`

У вузлі **Surface 1** створіть підвузол `Deformation`.

2D Plot Group 1 – Surface 1 : RMB > Deformation

2D Plot Group 1 – Surface 1 – Deformation 1:

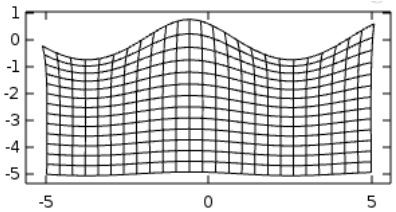
Expression – **x component** = `U(x,-y,t)`

– **y component** = `-V(x,-y,t)`

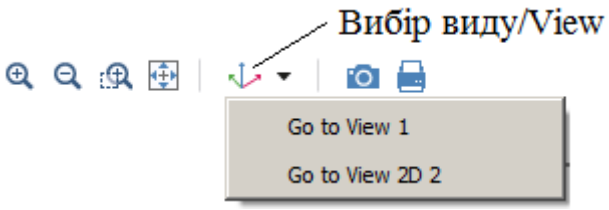
Scale – **Scale factor** = `1`

LMB > Plot

Введені тут вирази відповідають вектору зсувів, який визначається формулами (1) з урахуванням зміни знака, як про це казалося раніше. В результаті графічна група **2D Plot Group 1** буде містити деформоване каркасне зображення прямокутної області.



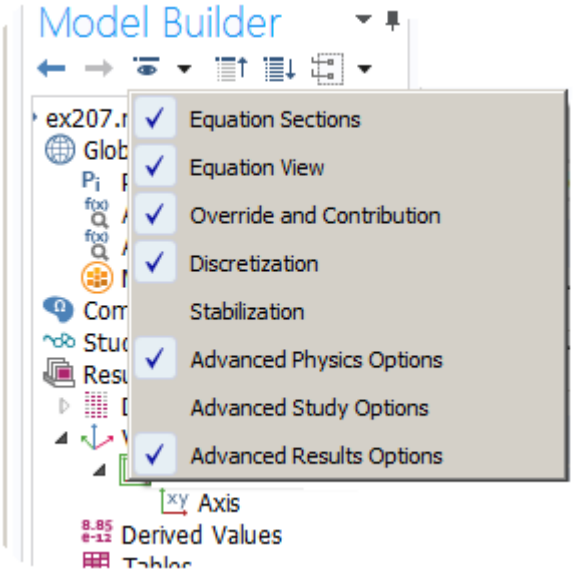
Прямокутник, який нами використовується, зображає невелику частину деформованого середовища. Щоб це показати графічно, можна обмежити видиму зону. Для цього використовується «інструмент» View/Вид. Він керує параметрами відображення в графічному вікні. Ви можете уявляти його як аналог видошукача в фотоапараті або кінокамери, який використовується для спостереження і визначення меж кадру. Кожен «вид» має свій вузол в дереві моделі. Вид (View) діє в вікні графічної групи. В графічної зоні «вид» вибирається зі списку, що розкривається кнопкою .



Крім іншого, у вузлі типу `View` задаються видимі розміри зображення. Саме ця особливість нам знадобиться в поточному прикладі.

Один вид **View 1** зазвичай створюється автоматично. Інші види, які використовуються для відображення результатів, користувач створює самостійно в гілці **Results - Views**. При створенні вузла `View` також створюється підвузол `Axis`.

Щоб гілка **Views** була помітна в дереві моделі, над деревом слід розкрити список **Show** кнопкою і встановити прапорець **Advanced Results Options** (можливо у вас він вже вибраний).



Додавання нового виду виконується клацанням правої кнопки миші по вузлу **Views**, і вибором з випадного списку команди View 3D або View 2D.

Results-Views : RMB > View 2D

Буде створено вузол **View 2D 2** з підвузлом **Axis**. Налаштуйте їх.

Results – Views - View 2D 2:

`View` – Lock axis = `True`

Ця опція захищає налаштування вузла **Axis** від змін, які можуть відбутися, якщо в графічному вікні використовуються інструменти zoom (будь-яка з кнопок Zoom In, Zoom Out, Zoom Box, Zoom Extents).

Results – Views - View 2D 2 - Axis :

`Axis` – x minimum, x maximum=`-4,4` (*два поля*)

– y minimum, y maximum=`-4,2` (*два поля*)

– View scale = `Automatic`

Опція **View scale**=`Automatic` використовується для збереження масштабу зображення у «виді». Якщо ця опція не включена і масштаб в графічному вікні змінювався, то при повторному виборі «виду» його первісний масштаб не повертається. Опції ПВ вузла **Axis**, що розташовані в розділі **`Grid`**,

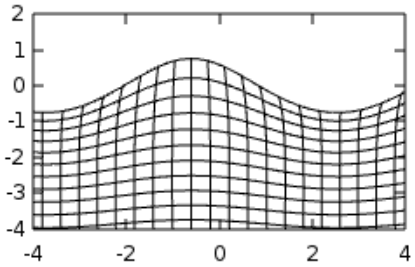
застосовуються для управління зарубками на осях координат і в поточному прикладі не використовуються.

Після створення і налаштування вузла **View 2D 2** активуйте вузол **2D Plot Group 1**. В його ПВ внесіть зміну.

2D Plot Group 1: `Plot Settings` – View = `View 2D 2`

Цим ви встановлюєте «вид» за замовчуванням для поточної графічної групи. Він буде вибиратися кожний раз, коли група активується, а також при створенні анімації з цієї групи.

Над графічним вікном розкрийте список видів кнопкою і виберіть команду Go to View 2D 2. В результаті ви отримаєте наступне зображення



Побудуйте анімацію.

`Results` –> Animation –> Player

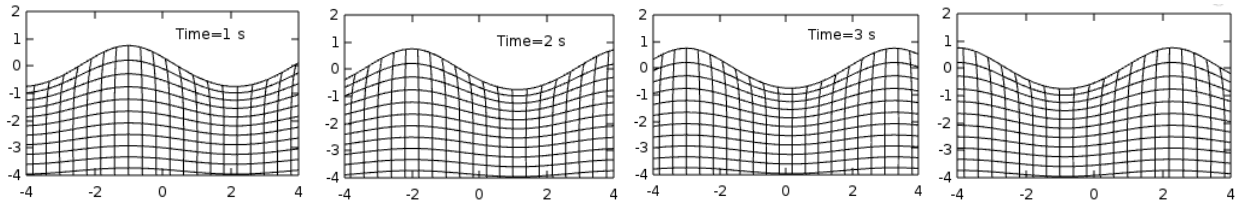
Animation 1:

`Scene` – **Subject** = `2D Plot Group 1`

`Frames` – **Frame selection** = `All`

`Playing` – **Repeat** = `True`

Згенеруйте кадри анімації командою Generate Frame і запустіть процес кнопкою Play . На наступному рисунку зображені кадри в моменти часу `t=1,2,3,4`.



2.3.15. Біжні хвилі в водоймищі скінченної глибини.

Водоймище скінченної глибини d, необмежене в горизонтальному напрямку, в стані спокою можна представляти у вигляді смуги $-\infty < x < \infty, -d \le y \le 0$. Біжні (рухомі) хвилі в ньому можна змоделювати рівняннями [1]

$$U = a\sin(wt - kx)\left(e^{ky} + e^{-2kd}e^{-ky}\right)$$
$$V = a\cos(wt - kx)\left(e^{ky} - e^{-2kd}e^{-ky}\right) \qquad (1)$$

де U і V миттєві зсуви частинок рідини, які в стані спокою мали координати (x, y), в напрямку осей X і Y (вісь Y спрямована вертикально вгору, і водоймище розташовано в області $y \le 0$).

Точки середовища в момент часу t матимуть координати $X = x + U(x, y, t)$, $Y = y + V(x, y, t)$. Тоді з формул (1) випливатиме, що частинки рідини рухатимуться по еліпсам

$$\frac{(X-x)^2}{a^2\left(e^{ky}+e^{-2kd}e^{-ky}\right)^2}+\frac{(Y-y)^2}{a^2\left(e^{ky}-e^{-2kd}e^{-ky}\right)^2}=1,$$

які зі збільшенням глибини поступово «сплющуються», а на дні вироджуються в горизонтальні відрізки. Цей факт можна продемонструвати графічно, якщо середовище зобразити у вигляді множини точок і накреслити траєкторії деяких з них.

Приклад. Створіть порожню модель.

File->New->Blank Model

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `w=1, L=12, a=1, d=6, k=0.4`.

Створіть функцію `U(x,y,t)`.

`Home`-> Definitions – Finctions -> Global - Analitic

Function name = `U`

Definition–Expression =

`a*sin(w*t-k*x)*(exp(k*y)+exp(-2*k*d)*exp(-k*y))`

Definition – **Arguments** = `x,y,t`

Аналогічно створіть другу функцію (1) `V`:

`V(x,y,t)=a*cos(w*t-k*x)*(exp(k*y)-exp(-2*k*d)*exp(-k*y))`

Додайте в модель двовимірний компонент.

Untitled.mph : RMB > Add Component - 2D

Його геометрія складатиметься з одного прямокутника.

`Geometry` -> **Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

Size – **Width, Height** = `2*L,d` (*два поля*)

Position – **Base, x, y** = `Corner,-L,-d` (*три поля*)

LMB > Build All Objects

Згенеруйте в геометрії прямокутну сітку.

`Mesh` -> **Generators** – Mapped

Mesh 1 – Mapped 1:

Domain Selection – **Geometric entity level** = `Entire geometry`

Задайте розміри клітинок сітки.

Mesh 1 – Size:

Element Size – **Custom**= `True`

Element Size Parameters – **Maximum element size**= `1`

LMB > Build All

Додайте в модель вузол динамічного дослідження.

Untitled.mph (root): RMB > Add Study = `Add Study` ->

Preset Studies - Time Dependent ->X

Задайте діапазон і інтервал зміни параметра часу `t`.

Study 1 - Step 1: Time Dependent:

Study Settings – **Times** = `range(0,0.2,2*pi)`

Виконайте обчислення.

Study 1 : LMB > Compute

Для зображення середовища створіть двовимірну графічну групу з деформованим `Surface` вузлом.

`Home` **-> Results -** Add Plot Group **->**2D Plot Group

2D Plot Group 1:

Data – **Data set** = `Sudy 1/Solution 1`

Data – **Time(s)** = `6.2`

Plot Settings – **Plot data set edges** = `False`

`2D Plot Group 1` **-> Add Plot -** Surface

2D Plot Group 1 – Surface 1:

Data – **Data set** = `From parent`

Expression – **Expression** = `1`

Coloring and Style – **Coloring, Color** = `Uniform, Black` (*два поля*)

– **Wireframe** = `True`

У вузлі **Surface 1** створіть підвузол деформацій.

2D Plot Group 1 – Surface 1 : RMB > Deformation

2D Plot Group 1 – Surface 1 – Deformation 1:

Expression – **x component** = `U(x,y,t)`

– **y component** = `V(x,y,t)`

Введені тут вирази відповідають вектору зсувів, що визначається формулами (1).

Scale – **Scale factor** = `1`

LMB > Plot

В результаті графічна група **2D Plot Group 1** буде містити деформоване каркасне зображення прямокутної області. Для цієї графічної групи створіть і налаштуйте «вид».

Results-Views : RMB > View 2D

Results – Views - View 2D 2:

View – **Lock axis** = `True`

Results – Views - View 2D 2 - Axis :

Axis – **x minimum, x maximum** = `-10,10` (*два поля*)

– **y minimum, y maximum** = `-d-0.1,2` (*два поля*)

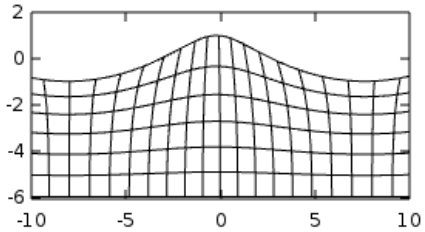
– **View scale** = `Automatic`

Після створення і налаштування вузла **View 2D 2** активуйте вузол **2D Plot Group 1**. В його ПВ внесіть зміну.

2D Plot Group 1: Plot Settings – **View** = `View 2D 2`

Цим встановлюється «вид» за замовчуванням для поточної графічної групи. В графічному вікні виконайте команду Zoom Extents . Потім над графічним

вікном розкрийте список видів кнопкою і виберіть команду Go to View 2D 2. В результаті ви отримаєте наступне зображення.



Точки середовища накреслимо, використовуючи графік `Scatter Surface`.

`2D Plot Group 1`-> Add Plot - More Plots -> Scatter Surface

2D Plot Group 1 - Scatter Surface 1 :

Data– **Data set** = `From parent`

Expression – **x component** = `x+U(x,y,t)`

– **y component** = `y+V(x,y,t)`

Evaluation Points–**x grid points** – **Entry method**=`Coordinates`

–**x grid points** – **Coordinates** =`range(-L,1,L)`

–**y grid points** – **Entry method**= `Coordinates`

–**y grid points** – **Coordinates** = `range(-d,1,0)`

Тут ми визначили двовимірний масив точок з початковими *x* - координатами `range(-L,1,L)` і *y* - координатами `range(-d,1,0)`. Вирази `x+U(x,y,t)` і `y+V(x,y,t)` використовуються для визначення нових положень цих точок.

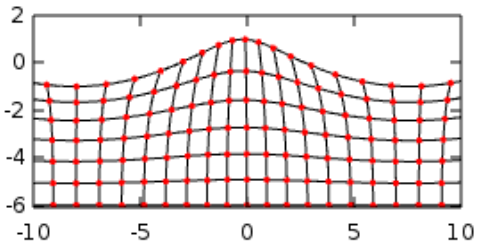
Radius – **Expression** =`0.15`

– **Radius scale factor** =`1`

Color – **Expression** =`1`

Coloring and Style – **Coloring, Color** = `Uniform, Red` (*два поля*)

LMB > Plot



Для побудови траєкторій руху точок середовища, потрібно джерело даних, яке представлятиме ці точки. Створіть його.

`Results`-> Data Set - Cut Point 2D

Results – Data Sets - Cut Point 2D 1:

Data– **Data set** = `Sudy 1/Solution 1`

Point Data – **Entry method** = `Grid`

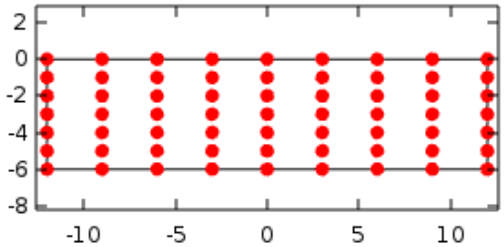
Point Data –**x** = `range(-L,3,L)`

В кресленні еліптичних траєкторій братиме участь кожний третій вузол (в горизонтальному напрямку) точкової сітки, побудованої вище.

Point Data –**y** = `range(-d,1,0)`

LMB > Plot

Якщо помітні не всі точки, то в графічному вікні виконайте команду Zoom Extents .



Щоб побудувати траєкторії руху цих точок, в графічну групу **2D Plot Group 1** додайте вузол `Point Trajectories`.

`2D Plot Group 1` -> Add Plot - More Plots -> Point Trajectories

2D Plot Group 1 - Point Trajectories 1 :

`Data`– Data set = `Cut Point 2D 1`

`Data`– Time (s) = `6.2`

`Trajectory Data` – Plot data = `Points`

– **x-expression** = `x+U(x,y,t)`

– **y-expression** = `y+V(x,y,t)`

Налаштуйте стиль криволінійних траєкторій.

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

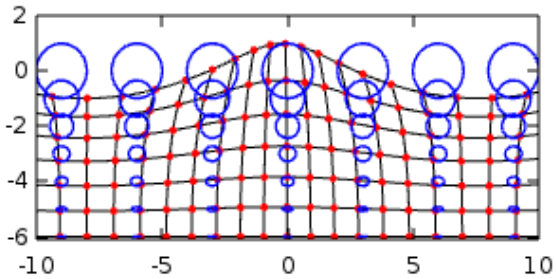
– **`Line style` – Tube radius expression** =`0.05`

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

– **`Line style` – Color** =`Blue`

LMB > Plot

Щоб отримати наступний графік, вам можливо потрібно буде виконати команду Zoom Extents , та зі списку видів обрати вид **View 2D 2**.



Криві, описані рухомими точками середовища, починаються з їх положення в момент часу `t=0` і закінчуються в момент, зазначений в полі **Time (s)**=`6.2` ПВ вузла **Point Trajectories 1**.

Побудуйте анімацію.

`Results` –> Animation –> Player

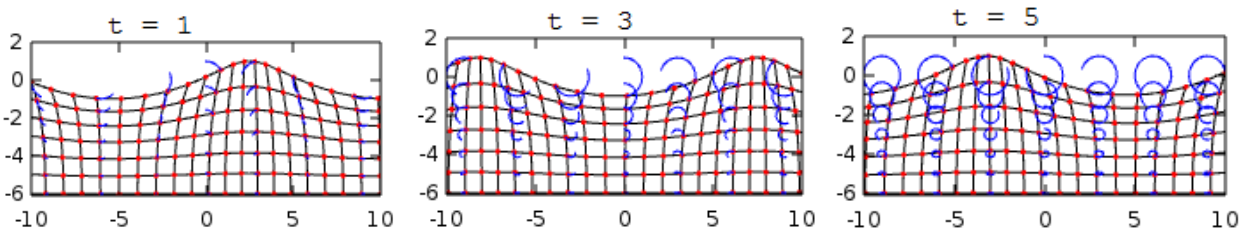
Animation 1:

`Scene` – Subject = `2D Plot Group 1`

`Frames` – Frame selection = `All`

`Playing` – Repeat = `True`

Згенеруйте кадри анімації командою Generate Frame і запустіть процес кнопкою Play . На наступному рисунку зображені кадри в моменти часу `t=1,3,5`.



2.3.16. Подовжні і крутні коливання.

Багато фізичних задач зводиться до розв'язання одновимірного хвильового рівняння

$$\frac{\partial^2 u(x,t)}{\partial t^2} = a^2 \frac{\partial^2 u(x,t)}{\partial x^2}, \qquad (1)$$

де $u(x,t)$ являє зміщення з положення рівноваги точки x одновимірного середовища в момент часу t. Цим рівнянням описуються поперечні коливання струни, подовжні коливання пружного стрижня і пружини, крутні коливаннях круглого стрижня і деякі інші. В параграфі 2.3.11 ми розглянули кілька розв'язків рівняння (1) з початковими умовами

$$u(x,0) = \varphi(x) \ (0 \le x \le L), \ u_t'(x,0) = 0 \qquad (2)$$

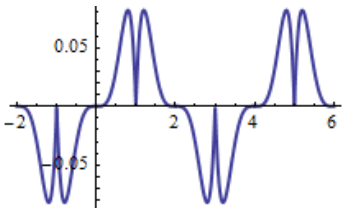
і нульовими граничними умовами $u(0,t) = 0$, $u(L,t) = 0$. Ці граничні умови імітують жорстке закріплення кінців середовища, наприклад, закріплення кінців струни. Однак для частини перелічених здач природнішими є граничні умови, коли один кінець закріплений, а інший вільний. У такій ситуації граничні умови мають вигляд:

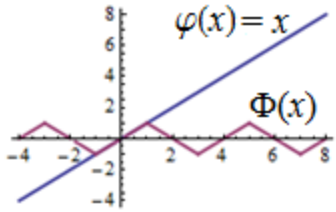
$$u(0,t) = 0, u'_x(L,t) = 0 \qquad (3)$$

Викладемо коротко методику розв'язання цих задач [4].

Розглянемо функцію $\varphi(x)$, задану на відрізку $[0, L]$ таку, що $\varphi(0) = 0$. Продовжимо її парно відносно точки $x=L$ на відрізок $[L,2L]$, тобто побудуємо функцію $\varphi_1(x), x \in [0, 2L]$ таку, що для $x \in [0, L]$, $\varphi_1(x) = \varphi(x)$, а для $x \in [L, 2L]$ $\varphi_1(x) = \varphi(2L - x)$. Функцію $\varphi_1(x)$ з відрізка $[0,2L]$ продовжимо непарно на відрізок $[-2L, 0]$, тобто побудуємо функцію $\varphi_2(x), x \in [-2L, 2L]$ таку, що для $x \in [0, 2L]$, $\varphi_2(x) = \varphi_1(x)$, а для $x \in [-2L, 0]$, $\varphi_2(x) = -\varphi_1(-x)$.. Отриману функцію $\varphi_2(x)$ з відрізка $[-2L,2L]$ продовжимо періодично на всю вісь з періодом $4L$. В результаті отримаємо періодичну функцію $\Phi(x)$, яка на відрізку $x \in [0, L]$ збігається з $\varphi(x)$, є парною відносно точки $x=L$ і непарною відносно точки $x=0$. Назвемо описане продовження функції $\varphi(x)$ парно-непарним періодичним продовженням.

На наступному рисунку ліворуч наведено графік парно-непарного продовження функції $\varphi(x) = x^4(1-x)$ з відрізка $[0,1]$. Парно-непарне продовження функції $\varphi(x) = x$ з відрізка $[0,1]$ на дійсну вісь показано на рисунку праворуч.





Побудувавши функцію (аналогічно до формули (5) параграфа 2.3.11)

$$u(x,t)=\frac{\Phi(x+at)+\Phi(x-at)}{2}, \qquad (4)$$

отримуємо розв'язок задачі (1) – (3). Дійсно, права частина (4) при будь-якій функції $\Phi(x)$ є розв'язком хвильового рівняння (1). Крім того, $u(x,0)=\Phi(x)$, а при $0\le x\le L$ $\Phi(x)=\varphi(x)$, тобто $u(x,0)=\varphi(x)$. Також, продиференціювавши (4) по t, і підставивши t=0, отримуємо $u_t'(x,0)=0$.

Відомо [2], що коли початкові дані задачі про поширення коливань на необмеженій прямий є непарними функціями відносно деякої точки x_0, то розв'язок (4) в цій точці дорівнює нулю в будь-який момент часу. Якщо початкові дані в задачі про поширення коливань на необмеженої прямий є парними функціями відносно деякої точки x_1, то похідна по x розв'язку (4) в цій точці дорівнює нулю. Оскільки $\Phi(x)$ непарна функція відносно точки x=0 і парна - відносно x=L, то $u(x,t)$ буде задовольняти умові $u(0,t)=0$ і умові $u'_x(L,t)=0$ $\forall t\ge 0$. Розглядаючи $u(x,t)$ тільки на відрізку $0\le x\le L$ ми отримуємо розв'язок задачі (1) – (3).

В роботі [4] показано, що для функції $\varphi(x)$,яка задана на відрізку $[0, L]$ і $\varphi(0)=0$, функцію $\Phi(x)$ парно-непарного періодичного продовження можна побудувати за формулою

$$\Phi(x)=\mathrm{psign}\,(x,4L)\cdot\varphi(stc(x,2L)), \qquad (5)$$

де $\mathrm{psign}\,(x,w)$ введена нами раніше періодична знакова функція, яка на півінтервалі $[-w/2,0)$ дорівнює −1, на ділянці $[0,w/2)$ дорівнює 1, і далі повторює свої значення періодично з періодом w. Функція $\mathrm{stc}(x,w)$ визначена в параграфі 2.3.11. Вона є парною «пилкоподібною» періодичною по змінній x функцією з періодом w, яка на відрізку $-w/2\le x\le w/2$ збігається з функцією $|x|$, а поза відрізком повторюється періодично.

У цьому параграфі ми змоделюємо три види механічних коливань, які зводяться до розв'язання задачі (1) – (3). Почнемо з моделювання подовжніх коливань пружини.

Нехай спіральна пружина довжиною L, закріплена на лівому краю і вільна на правому, піддана лінійному розтягненню. Початок координат розмістимо на лівому кінці осі пружини, і вісь X направимо вправо уздовж осі. Якщо до вільного кінця пружини прикласти розтяжне зусилля, то деформація буде лінійною. Припустимо, що під дією зусилля правий кінець зміститься на одиницю. Тоді початкове зміщення матиме вигляд $u(x,0)=\varphi(x)=x/L$. В нульовий момент часу розтягнутий кінець звільняється і пружина починає стискатися і розтискатися - починаються подовжні коливання. Вони

описуються задачею (1) – (3), де $u(x,t)$ є подовжнім зміщенням в момент часу t точок недеформованої пружини, розташованих в поперечному перерізі з координатою x. Розв'язок задачі репрезентується формулою (4), де $\Phi(x)$ визначається в (5).

Нехай параметричні рівняння гвинтової кривої, що репрезентує недеформовану пружину, мають вигляд $x = x(\tau), y = y(\tau), z = z(\tau)$. Точка пружини в момент часу t зміщується в подовжньому напрямку на величину $u(x,t)$. Тому в момент t вона матиме нову координату $X = x(\tau) + u(x(\tau),t)$. Вважатимемо, що поперечні розміри пружини не змінюються, тоді не змінюються і координати y і z точок. В результаті точка пружини, яка в недеформованому стані мала координати $(x(\tau), y(\tau), z(\tau))$, в момент часу t буде мати координати

$$X = x(\tau) + u(x(\tau),t),\ Y = y(\tau),\ Z = z(\tau). \qquad (6)$$

Змоделюємо подовжні коливання пружини радіуса r. Пружину зобразимо у вигляді просторової кругової спіральної кривої довжиною L з n обкрутами на одиниці довжини.

Приклад 1. *Подовжні коливання пружини*. Створіть порожню модель.

File->New->Blank Model

Додайте в модель параметри.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Global Definitions - Parameters** в таблиці задайте параметри `L=4,a=1,n=4,r=1`.

Створіть допоміжні функції $\mathrm{psign}(x,4L)$ і $\mathrm{stc}(x,2L)$. Спочатку створіть періодичну знакову функцію $\mathrm{psign}(x)$.

Home-> Definitions – Finctions -> Global - Analitic

Function name = `psign`

Definition – **Expression** = `(x<2*L)-(x>=2*L)`

Definition – **Arguments** = `x`

Periodic Extension – **Make periodic** = `True`

– **Lower limit, Upper limit** = `0,4*L` (*два поля*)

Потім створіть функцію $\mathrm{stc}(x)$.

Home-> Definitions – Finctions -> Global - Analitic

Function name = `stc`

Definition –**Expression, Arguments** = `abs(x),x` (*два поля*)

Periodic Extension – **Make periodic** = `True`

– **Lower limit, Upper limit** = `-L,L` (*два поля*)

Використовуючи формулу (5), створіть функцію $F(x)$, яка представлятиме парно-непарне періодичне продовження функції $\varphi(x) = x/L$.

```
F(x)=psign(x)*stc(x)/L
```

Зауваження. Формула для функції $F(x)$ схожа на відповідну формулу параграфа 2.3.11, але періоди використаних тут функцій `psign(x)` і `stc(x)` відрізняються, а там співпадали.

За формулою (4), створіть функцію

`u(x,t)=(F(x+a*t)+F(x-a*t))/2`

Для стислості ми не наводимо вміст панелей властивостей цих функцій.

Створіть в моделі тривимірний компонент.

Untitled.mph : RMB > Add Component - 3D

Геометрія компонента складатиметься з параметричної кривої, яка репрезентуватиме пружину.

Component 1 (comp1):

`Geometry` **-> Primitives – More Primitives** – Parametric Curve

Component 1 (comp1) – Geometry 1 - Parametric Curve 1:

Parameter – **Name** = `tau`

– **Minimum, Maximum** = `0,L` (*два поля*)

Expression – **x** = `tau`

– **y** = `r*sin(2*pi*tau*n)`

– **z** = `r*cos(2*pi*tau*n)`

Додайте в модель вузол динамічного дослідження.

Untitled.mph (root): RMB > Add Study = `Add Study ->`

Preset Studies - Time Dependent ->X

Задайте діапазон і інтервал зміни параметра часу `t`.

Study 1 - Step 1: Time Dependent:

Study Settings –**Times** = `range(0,0.25,4*L)`

Виконайте обчислення.

Study 1 : LMB > Compute

Перш ніж моделювати рух пружини, побудуємо графік розв'язку `u(x,t)`. Для цього нам знадобиться одновимірне ДД `Grid 1D`.

`Results` -> Data Set - More Data Sets **->** Grid 1D

Data Sets - Grid 1D 1 :

Data – **Source** = `Data Set`

Data – **Data Set** = `Study 1/Solution 1`

Використання в якості джерела даних **Study 1/Solution 1** дозволяє мати в «робочому просторі» **Grid 1D 1** змінну часу *t*.

Paramemter Bounds – **Name**= `x`

Paramemter Bounds – **Minimum, Maximum** = `0,L` (*два поля*)

Створіть групу одновимірних графіків і в ній вузол `Line Graph`.

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

1D Plot Group 1:

Data Set – **Data** = `Grid 1D 1`

Data – **Time selection** = `From list`

Data – **Time(s)** = `1`

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

– **x minimum, x maximum** =`0,L` (*два поля*)

– **y minimum, y maximum** =`-1,1` (*два поля*)

Legend – **Show legends** = `True`

– **Positions** = `Upper left`

Включення «легенди» (**Show legends**=`True`) в графічній групі (а також в дочірньому графічному вузлі **Line Graph 1**) дозволить відобразити на графіку момент часу.

RMB > Line Graph

1D Plot Group 1 - Line Graph 1:

`Data` – **Data set** = `From Parent`

`y-Axis Data` – **Expression** = `u(x,t)`

`x-Axis Data` – **Parameter, Expression** = `Expression, x` (*два поля*)

`Coloring and Style` – **Color, Width** =`Blue, 3` (*два поля*)

`Legends` – **Show legends** = `True`

LMB > Plot

Виконайте команду Plot і отримайте графік функції `u(x,1)`.

Побудуйте анімацію графіка функції `u(x,t)`.

`Results` –> Animation –> Player

Animation 1:

`Scene` – **Subject** = `1D Plot Group 1`

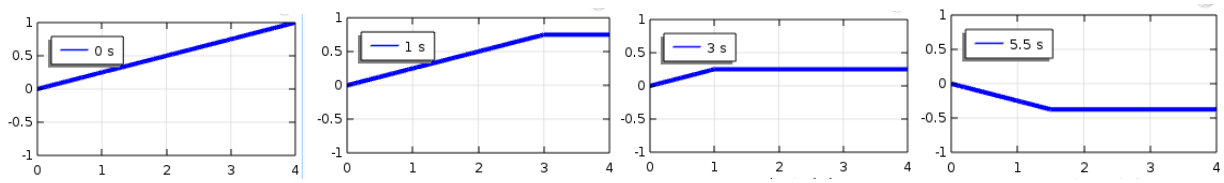
`Frames` – **Frame selection** = `All`

`Playing` – **Repeat** = `True`

LMB > Generate Frame

Graphics (графічне вікно) : LMB > Play

На наступному рисунку зображені графіки розв'язку `u(x,t)` в моменти часу `t=0,1,3,5.5`.



Тепер змоделюємо подовжні коливання пружини, створивши тривимірну графічну групу з деформованим `Line` вузлом.

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

3D Plot Group 2 :

`Data` – **Data set** = `Study 1/Solution 1`

`Data` – **Time(s)** = `13.75`

`Plot Settings` – **Plot data set edges** = `False`

`3D Plot Group 2` –> Add Plot - Line

3D Plot Group 2 - Line 1 :

`Data` – **Data set** = `From Parent`

`Expression` – **Expression** = `1`

`Coloring and Style` – **Line type** =`Tube`

– **Tube radius expression** =`0.03`

– **Radius scale factor** =`True,1`

– **Color table** =`Wave`

– **Color legend** =`False`.

У вузлі **Line 1** створіть підвузол деформацій,

RMB > Deformation

3D Plot Group 2 – Line 1 – Deformation 1:

Expression – **x component** = u(x,t)

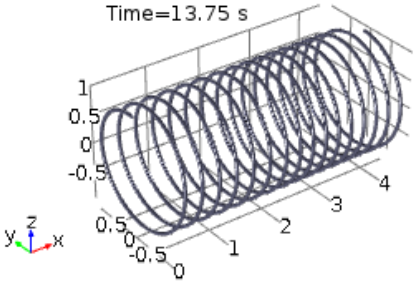
– **y component** = 0

– **z component** = 0

Scale – **Scale factor** = 1

LMB > Plot

В результаті графічна група 3D Plot Group 2 буде містити деформоване зображення пружини.



Побудуйте анімацію подовжніх коливань пружини.

Results –> Animation –> Player

Animation 2:

Scene – **Subject** = 3D Plot Group 2

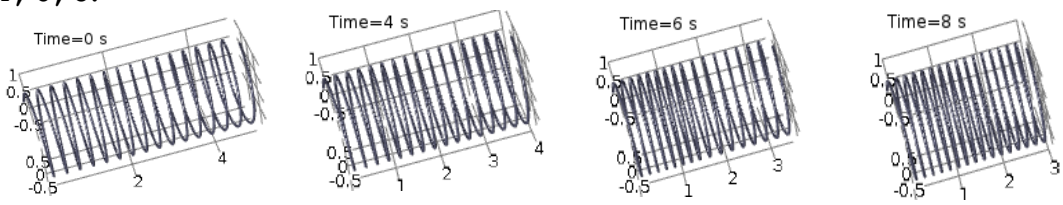
Frames – **Frame selection** = All

Playing – **Repeat** = True

LMB > Generate Frame

Graphics (графічне вікно) : LMB > Play

На наступному рисунку наведені зображення пружини в моменти часу t=0,4,6,8.



Збережіть файл, але не закривайте його.. ■

Подовжні коливання стрижня описуються рівнянням (1) з початковими умовами $u(x,0)=\varphi(x)$, $u'_t(x,t)=\psi(x)$, і граничними умовами, які залежать від способу закріплення кінців [5].

Нехай стрижень квадратного поперечного перерізу довжиною L закріплений на одному кінці і вільний на іншому. Початок координат розташовано в центрі лівого торця, і вісь X спрямовано вправо уздовж осі стрижня. Розтягнемо стрижень за правий кінець. Початкове зміщення матиме вигляд $u(x,0)=\varphi(x)=k\,x$, де k деяка стала. В момент часу $t=0$ розтягнутий кінець звільняється і стрижень починає скорочуватися, а потім довшати - починаються подовжні коливання. Такі коливання будуть описуватися рівнянням (1), де $u(x,t)$ є подовжнім зміщенням в момент часу t перерізу

стрижня, що в недеформованому стані мав координату x. Початкові умови матимуть вигляд

$$u(x,0)=k\,x,\;\; u'_t(x,0)=0\;, \qquad (7)$$

а граничні умови будуть мати вид (3). Розв'язок цієї задачі було побудовано вище (формули (4), (5)). Тут ми змоделюємо рух стрижня, перерізи якого зміщуються відповідно до розв'язку $u(x,t)$ задачі (1), (3), (7).

Позначимо координати довільної точки стрижня через x, y, z. Всі точки одного і того ж поперечного перерізу зміщуються в момент t на одну і ту ж величину $u(x,t)$. Тому в момент t вони матимуть подовжню координату $X=x+u(x,t)$. В площині перерізу зміщення точок немає, тому координати y і z точок перерізу не змінюються. В результаті в момент часу t точки стрижня, що мали в недеформованому стані координати x, y, z, матимуть координати $X=x+u(x,t),\; Y=y,\; Z=z$.

В прикладі покладемо, що перерізи стрижня мають форму квадрата з довжиною сторони 1, довжина стрижня L=4, і в (7) покладемо $k=1/L$.

Приклад 2. *Подовжні коливання стрижня*.

Продовжить роботу з поточним файлом, в який додайте другий тривимірний компонент.

Untitled.mph : RMB > Add Component - 3D

В ДМ з'явиться гілка **Component 2** з вузлом **Geometry 2**. Геометрія складатиметься з поверхні блока.

Component 2 (comp2):

Geometry **-> Primitives** – Block

Component 2 - Geometry 2 – Block 1:

Object Type – **Type** = Surface

Останнім визначається, що будуватиметься поверхня блока, а не тривимірне тіло.

Size – **Width, Depth, Height** = L,1,1 (*три поля*)

Position – **Base**= Corner

Position – **x, y, z** = 0,0,0 (*три поля*)

LMB > Build All Objects

На створеній поверхні згенеруйте прямокутну сітку.

Mesh **-> Generators - Boundary** – Mapped

Mesh 2 – Mapped 1:

Boundary Selection – **Geometric entity level** = Boundary

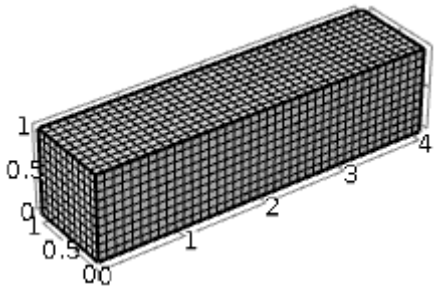
– **Selection** = All boundaries

Mesh 2 – Size:

Element Size – **Custom**= True

Element Size Parameters – **Maximum element size**= 0.1

LMB > Build All



Переобчисліть модель.

Study 1 : LMB > Compute

В гілці **Results – Data Sets** вузол джерела даних **Study 1/Solution 1** не зміниться. Але це ДД можна вже використовувати для побудови розв'язку задачі коливань стрижня, якщо створити копію ДД, і для неї обрати «базовим» другий компонент.

Data Sets – Study 1/Solution 1 : RMB > Duplicate

Data Sets – Study 1/Solution 1 (2) :

Solution – Component = Component 2(comp2)

До назви ДД **Study 1/Solution 1** в усіх вузлах, де воно створювалося і використовувалося, буде додано одиницю **Study 1/Solution 1 (1)**. Тому попередні графіки не зміняться.

Перейдемо до моделювання подовжніх коливань стрижня, створивши тривимірну графічну групу, яка спирається на нове ДД.

Results –> Plot Group - 3D Plot Group

3D Plot Group 3 :

Data – Data set = Study 1/Solution 1(2)

Data – Time(s) = 13.75

Plot Settings – Plot data set edges = False

Створіть зображення каркасної деформованої поверхні стрижня. Оскільки каркасне зображення прозоре, то спочатку побудуйте непрозоре однокольорове зображення деформованого стрижня.

3D Plot Group 3 –> Add Plot – Surface

3D Plot Group 3 - Surface 1:

Data– Data set = From parent

Expression – Expression = 1

Coloring and Style – Coloring, Color table =Color table,Wave

– Color legend = False

RMB > Deformation

3D Plot Group 3 – Surface 1 – Deformation 1:

Expression – x component = u(x,t)

– y component = 0

– z component = 0

Scale – Scale factor = 1

Тепер додайте каркасне деформоване зображення того ж самого стрижня.

3D Plot Group 3 - Surface 1: RMB > Duplicate

3D Plot Group 3 - Surface 2:

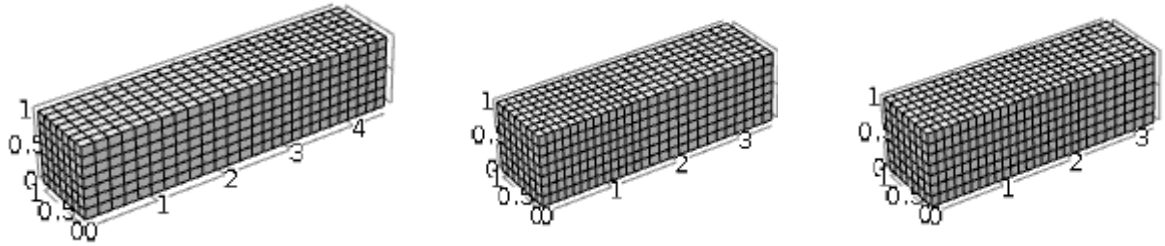
Coloring and Style – **Coloring, Color** = Uniform, Black (*два поля*)
– **Wireframe** = True

Вузол **Deformation 1** дублюється без змін. Це призводить до того, що зміщення точок поверхні «каркасного» зображення **Surface 2** такі ж самі, як зміщення точок суцільного зображення **Surface 1**.

LMB > Plot

В графічній групі **3D Plot Group 3** з'явиться зображення деформованого стрижня.

Побудуйте (самостійно) анімацію на базі цієї графічної групи. На наступному рисунку показані зображення в моменти часу t=2,6,7, які відповідають різним етапам стиснення стрижня.



t=2 t=6 t=7

Зверніть увагу на те, що щільність координатної сітки на поверхні різна в різних частинах стрижня. Це відповідає тому, що різні частини стиснуто по-різному.

■

Розглянемо круглий горизонтальний стрижень довжиною L з радіусом r, закріплений на лівому торці і закручений парою сил на правому. Кругові перерізи стрижня залишаються круговими в процесі закручування, а діаметр перерізів і відстані між ними не змінюється за умови, що кут закручування малий. Початок координат розташовано в центрі лівого торця стрижня і вісь X спрямовано уздовж осі вправо. В початковий момент правий край звільняється і в стрижні виникають крутні коливання. Відомо [5], що для поперечного перерізу з координатою x кут повороту $u(x,t)$ задовольняє хвильовому рівнянню (1).

Початковий кут повороту $u(x,0)$ лінійно змінюється уздовж стрижня $u(x,0)=k\,x$, де стала k залежить від властивостей матеріалу стрижня і пари закручувальних сил.

Нехай x, y, z позначають координати точок недеформованого стрижня. Розглянемо фіксований переріз. Подумки проведемо в ньому координатні осі Oy і Oz. В процесі скручування вони разом з перерізом повертаються на кут $u(x,t)$. В повернутій системі координати точок y, z перерізу не змінюються. Тоді глобальні координати Y, Z цих точок пов'язані з їх координатами y, z в повернутій на кут u системі наступними формулами

$$Y = y\cdot\cos u + z\cdot\sin u$$

$$Z = -y\cdot\sin u + z\cdot\cos u$$

При цьому кут повороту u будь-якого перерізу є функцією координати x і моменту часу t, тобто $u = u(x,t)$, де функцію $u(x,t)$ слід шукати як розв'язок рівняння (1) з початковими умовами $u(x,0) = k\,x,\ u'_t(x,0) = 0$ і граничними умовами $u(0,t) = 0$ (лівий торець закріплений, повороту немає), $u'_x(L,t) = 0$ (правий край вільний) [5]. Розв'язок цієї задачі побудовано в поточному параграфі раніше. Використаємо його для графічного зображення крутних коливань. Вважатимемо, що L=4, $k = 1/L$ і $r = 1$.

Приклад 3. *Крутні коливання циліндричного стрижня*. Продовжить роботу з поточним файлом, в який додайте ще один тривимірний компонент.

Untitled.mph : RMB > Add Component - 3D

В ДМ з'явиться гілка **Component 3** з вузлом **Geometry 3**. Геометрія компонента складатиметься з одного циліндра.

Component 3 (comp3):

Geometry **-> Primitives** – Cylinder

Size and Shape – **Radius, Height** = r, L (*два поля*)

Axis – **Axis type** = x-axis

В створеній геометрії генеруйте чотирикутну сітку.

Mesh **-> Generators - Boundary** – Mapped

Mesh 3 – Mapped 1:

Boundary Selection – **Geometric entity level** = Boundary

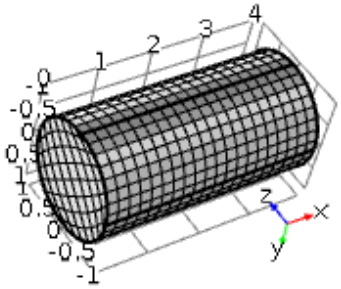
– **Selection** = All boundaries

Mesh 3 – Size:

Element Size – **Custom**= True

Element Size Parameters – **Maximum element size**= 0.2

LMB > Build All



Переобчисліть модель.

Study 1 : LMB > Compute

В гілці **Results – Data Sets** вузол джерела даних **Study 1/Solution 1** не зміниться. Але його вже можна використовувати для побудови розв'язку задачі крутних коливань, якщо створити копію ДД і обрати «базовим» новий компонент.

Data Sets – Study 1/Solution 1 : RMB > Duplicate

Data Sets – Study 1/Solution 1 (3) :

Solution – **Component** = Component 3(comp3)

Тепер змоделюйте крутні коливання стрижня, створивши тривимірну графічну групу, яка спирається на нове ДД.

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

3D Plot Group 4 :

Data – **Data set** = `Study 1/Solution 1(3)`

Data – **Time(s)** = `6.75`

Plot Settings – **Plot data set edges** = `False`

Створіть зображення каркасної деформованої поверхні стрижня. Оскільки каркасне зображення прозоре, то спочатку побудуйте непрозоре зображення стрижня.

`3D Plot Group 4` -> Add Plot – Surface

3D Plot Group 4 - Surface 1:

Data– **Data set** = `From parent`

Expression – **Expression** = `1`

Coloring and Style – **Coloring, Color table** =`Color table,Wave`

– **Color legend** = `False`

RMB > Deformation

3D Plot Group 4 – Surface 1 – Deformation 1:

Expression – **x component** = `0`

– **y component** = `y*cos(u(x,t))+z*sin(u(x,t))-y`

– **z component** = `-y*sin(u(x,t))+z*cos(u(x,t))-z`

Вирази, які введені в полях **y component** і **z component**, представляють зміщення точок циліндра відносно їх недеформованого положення.

Scale – **Scale factor** = `1`

Тепер додайте каркасне деформоване зображення того ж самого стрижня.

3D Plot Group 4 - Surface 1: RMB > Duplicate

3D Plot Group 4 - Surface 2:

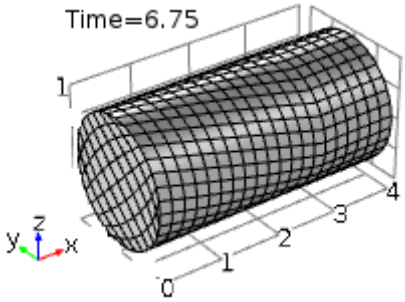
Coloring and Style – **Coloring, Color** = `Uniform, Black` (*два поля*)

– **Wireframe** = `True`

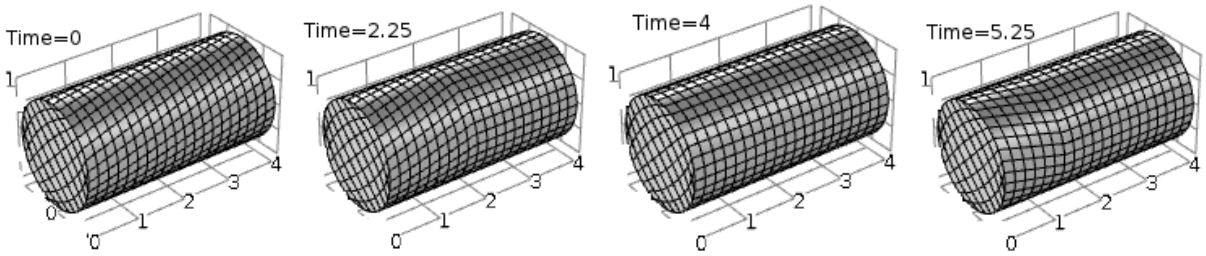
Вузол **Deformation 1** дублюється без змін. Тому каркасне зображення **Surface 2** накладається на суцільне **Surface 1**.

LMB > Plot

В графічній групі **3D Plot Group 4** з'явиться зображення закрученого стрижня.



Побудуйте (самостійно) анімацію на базі цієї графічної групи. Зображення циліндричного стрижня в початковий і подальші `t=2.25,4,5.25` моменти часу наведені на наступному рисунку.



Зверніть увагу, що геометричне тіло - циліндр залишається циліндром, проте змінюється координатна сітка на його поверхні. Вона показує, як закручені ті чи інші частини циліндра.

Література.

1. Крауфорд Ф. Волны. – М.: Наука, 1974. – 528 с.
2. Тихонов А.Н., Самарский А.А. Уравнения математической физики. – М.: Наука, 1977. – 736с.
3. Доля П.Г. Периодическое продолжение функций и решение уравнения колебаний струны в системах символьной математики.// Вестник Харьк. нац. ун-та., - 2006.- № 733. Сер. ”Математическое моделирование.
4. Dolya P.G. Solution to the homogeneous boundary value problems of free vibrations of a finite string // Journal of Mathematical Physics, Analysis, Geometry, - 2008, vol.4, No 2 , pp. 237 – 251.
5. Тимошенко С.П. Колебания в инженерном деле. – М.: Гос. изд. физ.-мат. литературы, 1959.
6. Рашевский П.К. Курс дифференциальной геометрии. – М.: Гос. изд. техн.-теретич. лит., 1956.
7. Тимошенко С.П., Гудьер Дж. Теория упругости. – М.: Наука, 1979.
8. Филоненко Бородич М.М. Теория упругости. – М.: Гос. изд. физ.-мат. лит., 1959.
9. Снеддон И.Н., Берри Д.С. Классическая теория упругости. – М.: Гос. изд. физ.-мат. лит., 1961.
10. Карслоу Г., Егер Д. Теплопроводность твердых тел. – М.: Наука, 1964.– 487 с.

3. Класичні задачі математичної фізики.

Система COMSOL Multiphysics призначена для розв'язання статичних і динамічних систем **д**иференціальних **р**івнянь в **ч**астинних **п**охідних (ДРЧП) другого порядку з однією, двома або трьома просторовими змінними. Для багатьох задач COMSOL містить зумовлені, так звані, прикладні режими або інтерфейси, які використовуються як шаблони, що приховують складні деталі конструювання математичних моделей. Але як прикладні режими, так і класичні задачі математичної фізики зводяться до розв'язання одного або системи кількох ДРЧП.

В залежності від задачі в COMSOL Multiphysics, як правило, використовуються дві форми ДРЧП. Коефіцієнтна форма (`Coefficient Form PDEs`) має вигляд

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \left(-c\,\nabla u - \alpha\, u + \gamma\right) + \beta \cdot \nabla u + \mathrm{a}\, u = f\,, \qquad (1)$$

де у випадку одного рівняння коефіцієнти є відомими функціями (або вектор-функціями) просторових координат, часу і, можливо, самої шуканої величини *u*. У випадку системи *u* представляє вектор-функцію, а коефіцієнти – матриці або вектора. Коефіцієнтна форма використовується для лінійних і слабо нелінійних задач, коли старші похідні по просторовим координатам входять в рівняння лінійним чином. Окремими випадками рівняння (1) є класичні ДРЧП, такі як хвильове рівняння або теплопровідності.

Загальна (генеральна) форма має вигляд

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \Gamma = F\,, \qquad (2)$$

де *Г* і *F* можуть бути величинами (частіше векторами), що залежать від шуканих функцій і їх просторових похідних. Загальний вид коефіцієнтної форми (1) може бути отриманий з (2), якщо покласти $\Gamma = -c\,\nabla u - \alpha\, u + \gamma$ і $F = f - \beta \cdot \nabla u - \mathrm{a}\, u$. Наприклад, щоб (2) набуло вигляду двовимірного рівняння Пуассона $\nabla \cdot \nabla u = 1$ потрібно покласти $e_a = 0, d_a = 0$, $\Gamma = \nabla u = [u'_x, u'_y]$ и $F = 1$. Однак в якості *Г* і *F* можуть виступати досить складні вирази. Форму запису (2) зручно використовувати для нелінійних рівнянь, коли коефіцієнтна форма непридатна.

При розв'язанні одного рівняння (2) коефіцієнт *F* є скаляром, а терм *Г* – вектором, як правило залежним від просторових координат і розв'язку *u*. При застосуванні генеральної форми граничні умови також використовують терм *Г*, який зазвичай є вектором потоку «збережуваної» величини (тепла, заряду, маси, моменту). На межі тіла цей потік може дорівнювати нулю або будь-якій відомій величині. Замість потоку на границі може задаватися значення шуканої функції *u* або який-небудь вираз, залежний від *u* і її похідних.

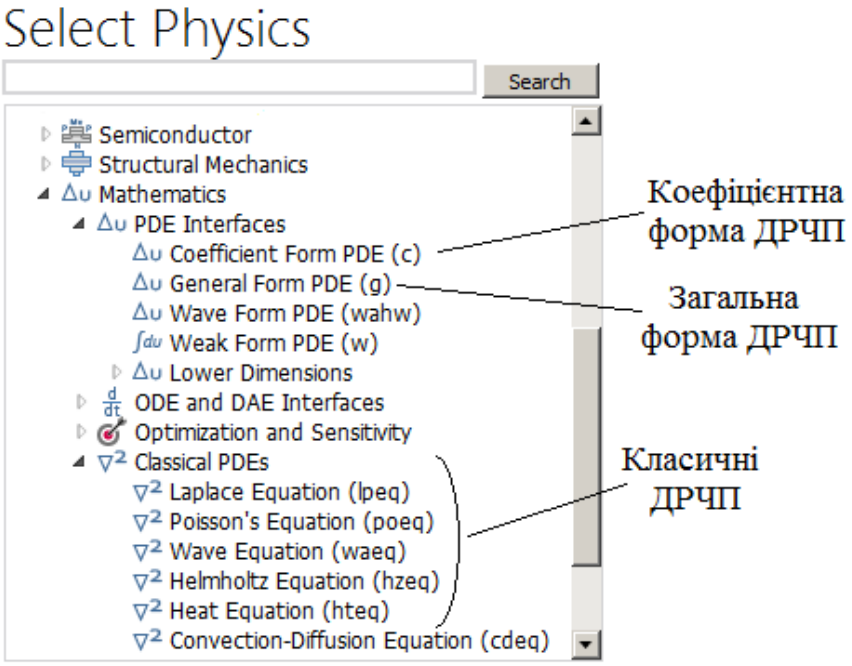
При чисельній реалізації різниці між формами (1) і (2) немає, якщо вони представляють одне і те же ДРЧП. Насправді в обох випадках за лаштунками використовується третя, так звана, слабка форма ДРЧП. Однак COMSOL

Multiphysics має гарний графічний інтерфейс, який дозволяє абстрагуватися від деталей реалізації та методів обчислення.

Нагадаємо позначення, які використовуються під час запису ДРЧП в COMSOL Multiphysics.

$\nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z}\right]$	оператор градієнта
$\nabla u = \text{grad}\, u = \frac{\partial u}{\partial x}\boldsymbol{i} + \frac{\partial u}{\partial y}\boldsymbol{j} + \frac{\partial u}{\partial z}\boldsymbol{k}$	градієнт скалярної функції $u(x, y, z)$
$\nabla \cdot \boldsymbol{u} = \text{div}\, \boldsymbol{u} = \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z}$	дивергенція вектор-функції $\boldsymbol{u} = (u_x, u_y, u_z)$
$\nabla \times \boldsymbol{u} = \text{rot}\, \boldsymbol{u} = \left(\frac{\partial u_z}{\partial y} - \frac{\partial u_y}{\partial z}\right)\boldsymbol{i} + \left(\frac{\partial u_x}{\partial z} - \frac{\partial u_z}{\partial z}\right)\boldsymbol{j} + \left(\frac{\partial u_y}{\partial x} - \frac{\partial u_x}{\partial y}\right)\boldsymbol{k}$	ротор вектор-функції $\boldsymbol{u} = (u_x, u_y, u_z)$

Більшість класичних ДРЧП є окремими випадками коефіцієнтної форми (1). Кожне таке рівняння має власний інтерфейс, який при побудові шаблона моделі можна обрати в розділі **`Select Physics`** з гілки `Mathematics - Classical PDEs`.



Для довідки наведемо основні класичні ДРЧП, використовуючи стислі позначення векторного аналізу і покоординатний запис (для двовимірного випадку).

Таблиця 1. Класичні ДРЧП.

	Рівняння	Стислі позначення COMSOL	Покоординатні позначення (2D)
1	Рівняння Лапласа	$\nabla\cdot(-\nabla u)=0$	$\frac{\partial}{\partial x}\left(-\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(-\frac{\partial u}{\partial y}\right)=0$
2	Рівняння Пуассона	$\nabla\cdot(-c\nabla u)=f$	$\frac{\partial}{\partial x}\left(-c\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(-c\frac{\partial u}{\partial y}\right)=f$
3	Рівняння Гельмгольца	$\nabla\cdot(-c\nabla u)+\mathrm{a}u=f$	$\frac{\partial}{\partial x}\left(-c\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(-c\frac{\partial u}{\partial y}\right)+$ $+\mathrm{a}u=f$
4	Рівняння теплопровід-ності	$d_a\frac{\partial u}{\partial t}+\nabla\cdot(-c\nabla u)=f$	$d_a\frac{\partial u}{\partial t}+\frac{\partial}{\partial x}\left(-c\frac{\partial u}{\partial x}\right)+$ $+\frac{\partial}{\partial y}\left(-c\frac{\partial u}{\partial y}\right)=f$
5	Хвильове рівняння	$e_a\frac{\partial^2 u}{\partial t^2}+\nabla\cdot(-c\nabla u)=f$	$e_a\frac{\partial^2 u}{\partial t^2}+\frac{\partial}{\partial x}\left(-c\frac{\partial u}{\partial x}\right)+$ $+\frac{\partial}{\partial y}\left(-c\frac{\partial u}{\partial y}\right)=f$
6	Рівняння конвекції-дифузії	$d_a\frac{\partial u}{\partial t}+\nabla\cdot(-c\nabla u)+$ $+\beta\cdot\nabla u=f$	$d_a\frac{\partial u}{\partial t}+\frac{\partial}{\partial x}\left(-c\frac{\partial u}{\partial x}\right)+$ $+\frac{\partial}{\partial y}\left(-c\frac{\partial u}{\partial y}\right)+\beta_x\frac{\partial u}{\partial x}+\beta_y\frac{\partial u}{\partial y}=f$

В цій главі наводяться приклади розв'язання задач, які з самого початку сформульовані як ДРЧП, хоча в кожному випадку вони мають фізичне підгрунтя. Деякі з них не мають практичного сенсу і наведені для того, щоб показати можливості «інструментів» системи на зрозумілих прикладах. Частина задач є переробленими прикладами інших авторів.

3.1. Крайова задача для ЗДР.

На відрізку [0,1] розв'яжемо крайову задачу

$$u''_{xx}+u=-x,\ \ 0\le x\le 1,\ u(0)=0,\ u(1)=0.$$

Її точний розв'язок відомий $u(x)=\frac{\sin x}{\sin 1}-x$, і може бути згодом порівняний з наближеним.

За допомогою майстра *створіть* 1D *модель задачі*

File->New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Helmholtz Equation (hzeq) -> Stationary

Вимкніть в моделі підтримку розмірностей фізичних величин.

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

Геометрія моделі складається з відрізка [0,1]. Створіть його командою

Geometry->**Primitives->** Interval

В ПВ створеного вузла **Interval 1** залиште значення за замовчуванням і завершіть конструювання геометрії натисканням на кнопку 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 **Helmholtz Equation (hzeq) - Helmholtz Equation 1** в секції вибору **Selection** повинна стояти одиниця, яка означає, що рівняння буде діяти в області з індикатором 1 (це відрізок [0, 1]). Коефіцієнти рівняння задаються в полях в нижній частині цієї ПВ.

Diffusion Coefficient – **c** = -1

Source Term – **f** = -x

Absorption Coefficient – **a** = 1

Цим задається рівняння $u''_{xx} + u = -x$.

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = All boundaries

Dirichlet Boundary Condition – **r** = 0

Виконайте обчислення.

Home **-> Study** - Compute

Після завершення обчислень в графічній зоні вузла **Results - 1D Plot Group 1 - Line Graph 1** буде створено 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:

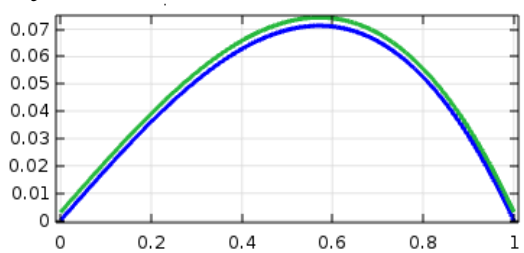
Selection – < > = 1 (індикатор відрізка [0,1])

y-Axis Data – **Expression** = sin(x)/sin(1)-x + 0.003

x-Axis Data – **Parameter** = Expression

x-Axis Data – **Expression** = x

В полі **Expression** додаток 0.003 використовується для того, щоб графіки наближеного і точного розв'язків не зливалися. Виконайте команду Plot і отримайте наступний рисунок.



Розв'яжемо те ж саме ДР з іншими значеннями шуканої величини *u* на кінцях відрізка. Деактивуйте вузол **Dirichlet Boundary Condition 1** командою

Dirichlet Boundary Condition 1: RMB > Disable

і створіть два нових вузла типу `Dirichlet Boundary Condition`, по одному для кожного кінця відрізка [0,1].

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

`Boundary Selection` – **Selection** = `Manual,1` (лівий кінець x=0)

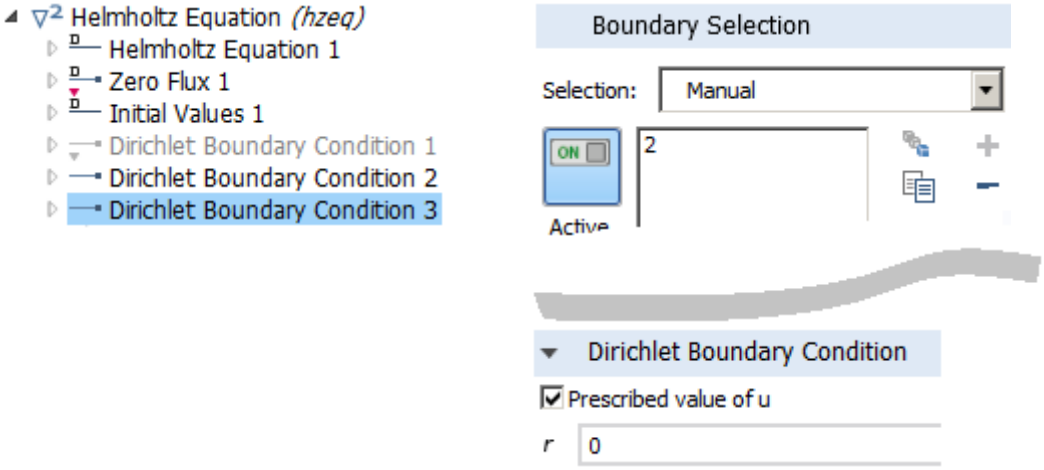
`Dirichlet Boundary Condition` – **r** = `1`

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 3:

`Boundary Selection` – **Selection** = `Manual,2` (правий кінець x=1)

`Dirichlet Boundary Condition` – **r** = `0`



Отже, якщо граничні значення на кінцях відрізка різні, то потрібно мати два вузла типу `Dirichlet Boundary Condition`.

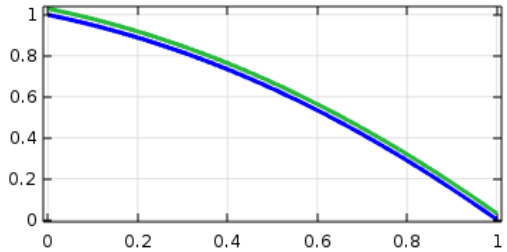
Виконайте команду Compute і в графічній зоні вузла **Results - 1D Plot Group 1 - Line Graph 1** отримайте графік розв'язку. Інший графік поки залишився без змін, і його треба поміняти. Для порівняння з аналітичним розв'язком $u = \dfrac{1-\cos 1}{\sin 1}\sin x + \cos x - x$ внесіть зміни в поле **Expression** вузла **Line Graph 2**.

1D Plot Group 1 - Line Graph 2:

`y-Axis Data` – **Expression** =

`cos(x)-x+(1-cos(1))/sin(1)*sin(x)+0.03`

Виконайте команду Plot і отримайте наступний рисунок.

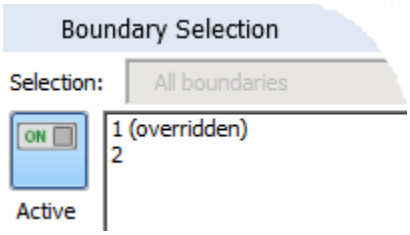


Розглянемо задачу, коли на правому кінці задана умова $u'_x(1)=0$. Точний розв'язок нової крайової задачі $u''_{xx}+u=-x,\ u(0)=1,\ u'_x(1)=0$ має вигляд

$u = \frac{1+\sin 1}{\cos 1}\sin x + \cos x - x$. Побудуємо наближений розв'язок. Для цього достатньо деактивувати останній вузол `Dirichlet Boundary Condition`.

Dirichlet Boundary Condition 3: RMB > Disable

Дійсно, гранична умова `Zero Flux` у вигляді $-\boldsymbol{n}\cdot(-c\nabla u)=0$ створюється на всіх границях автоматично при побудові «фізики». Задавання іншої граничної умови на частині межі відміняє `Zero Flux` на цій ділянці В одновимірному випадку вираз $-\boldsymbol{n}\cdot(-c\nabla u)=0$, де $\boldsymbol{n}$ – зовнішня нормаль до границі, в точці x=1 перетворюється на $-u'_x(1)=0$ (при `c=-1`). Отже, якщо зараз поглянути в ПВ вузла **Zero Flux 1**, то побачимо, що ця гранична умова діє на правому кінці (в точці 2), а умова `Zero Flux` на лівому анульована (бо на ньому задана умова Діріхле).



Зауваження. Якщо на ділянці границі не задано ніякої крайової умови, то за замовчуванням використовується умова `Zero Flux`, яка обнуляє «потік» шуканої величини.

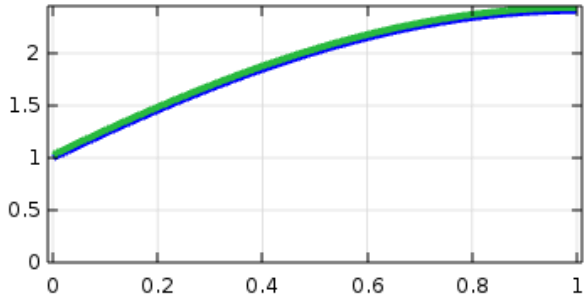
Переобчисліть модель і замініть вираз у вузлі **Line Graph 2**.

1D Plot Group 1 - Line Graph 2:

`y-Axis Data` – Expression =

`(1+sin(1))/cos(1)*sin(x)+cos(x)-x+0.03`

1D Plot Group 1 : LMB > Plot



3.2. Вільні коливання напівнескінченної струни.

Задача про вільні коливання напівнескінченної струни, лівий край якої жорстко закріплено, зводиться до розв'язання одновимірного хвильового рівняння

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2} \quad (0 < x < \infty) \tag{1}$$

з граничною і початковими умовами

$$u(0,t) = 0, \qquad u(x,0) = \varphi(x), \;\; u'_t(x,0) = \psi(x)\,. \tag{2}$$

де a – швидкість хвиль, $u(x,t)$ – відхилення точок струни від положення рівноваги в момент часу t, $\varphi(x)$ і $\psi(x)$ – початкові форма і поперечна швидкість точок струни [1].

Для розв'язання задачі (1) – (2) в COMSOL Multiphysics можна використати `Classical PDEs-Wave Equation` інтерфейс.

Розв'яжемо задачу, коли початкова швидкість $\psi(x) \equiv 0$, а початкова форма струни $\varphi(x)$ має вигляд «шпилястого капелюха» . Покладемо також $a = 1$.

За допомогою майстра *створіть* 1D *модель задачі*

File->New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Wave Equation (waeq) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

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

Створіть функцію $\varphi(x)$ початкової форми струни.

Global Definitions: RMB > Functions – Analytic

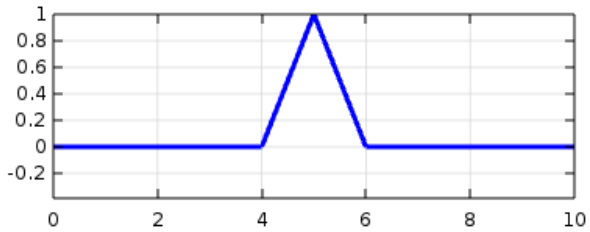
Function name = `f`

`Definition` – Expression = `(abs(x-4)-2*abs(x-5)+abs(x-6))/2`

Для перевірки виконайте побудову графіка цієї функції на відрізку [0,10].

`Plot Parameters` – Low limit, Upper limit = `0, 10`

LMB> Plot



Оскільки струна напівнескінченна, то в моделі нам знадобиться спеціальний «елемент», який буде імітувати нескінченну частину струни. Це відрізок обмеженої довжини, на якому при обчисленні будуть використовуватися масштабні коефіцієнти, які «розтягуватимуть» відрізок до нескінченності. Насправді розтягування буде виконуватися таким чином, щоб за розглядуваний час «кінцеві ефекти» не вплинули на розв'язок в «зоні інтересу», яка в нашому прикладі буде відрізком [0,10].

Отже геометрію моделі побудуємо як об'єднання двох відрізків: $[0,10] \cup [10,20]$.

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i1):

`Interval` – Left endpoint = `0`

`Interval` – Right endpoint = `10`

`Geometry->`**`Primitives->`** Interval

Geometry 1 - Interval 1 (i2):

`Interval` – Left endpoint = `10`

`Interval` – Right endpoint = `20`

Interval 2: LMB > Build All Objects

Другий відрізок [10,20] позначимо як нескінченний, тобто такий, на якому буде виконуватися масштабування. Для цього виконайте інструкції

`Definitions` -> **`Coordanate Systems`** – Infinite Element Domain

Definitions - Infinite Element Domain 1 (ie1):

`Domain Selection` – **Selection** = `Manual,2` (відрізок [10,20])

Щоб ввести індикатор 2 в секцію вибору **Selection** вам потрібно клацнути в графічному вікні по правому відрізку.

Побудуйте скінченно-елементну сітку з достатньо дрібними елементами, задавши їх максимальний розмір.

Mesh 1:

`Mesh Settings` – **Sequence type** = `User-controlled mesh`

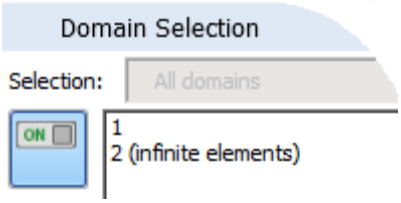
Mesh 1 - Size:

`Element Size` – **Custom** = `True`

`Element Size Parameters` – **Maximum element size** = `0.1`

LMB> Build All

Задайте параметри рівняння. В ПВ вузла **Wave Equation (waeq) - Wave Equation 1** в секції вибору повинні стояти індикатори обох відрізків. Це означає що рівняння буде діяти на множині $[0,10] \cup [10,20]$.



В розділі **`Equation`** наведено загальний вигляд класичного хвильового рівняння

$$e_a \frac{\partial^2 u}{\partial t^2} + \nabla \cdot (-c \nabla u) = f, \quad \nabla = \frac{\partial}{\partial x}.$$

Коефіцієнти рівняння задаються в полях в нижній частині вікна.

Wave Equation (waeq) - Wave Equation 1:

`Diffusion Coefficient` – **c** = `1`

`Source Term` – **f** = `0`

`Absorption Coefficient` – $\mathbf{e_a}$ = `1`

Цим задається хвильове рівняння $u''_{tt} - u''_{xx} = 0$.

Задайте граничні умови. В нашому прикладі граничну умову треба задавати лише на лівому краю тому, що другий відрізок [10,20] «імітує» півпряму $[10,\infty)$, на правому кінці якої граничну умову задавати непотрібно (він розташований на «нескінченності»).

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

`Boundary Selection` – **Selection,** <> = `Manual, 1`

`Dirichlet Boundary Condition` – **r** = `0` (умова закріплення)

Задайте початкові умови. Перевірте, що в ПВ вузла **Initial Values 1** в секції вибору **Selection** обрано індикатори обох відрізків. Потім введіть

Initial Values 1:

Initial Values – Initial Value for u:– u = f(x)

Initial Values – Initial time derivative of u:– $\frac{\partial u}{\partial t}$ = 0

Виконайте налаштування розв'язувача, задавши часовий інтервал і похибки обчислень.

Study 1 - Step 1: Time Dependent:

Study Settings –Times = range(0,0.1,12)

Study Settings –Relative tolerance = True, 0.0001

Study 1: RMB > Show Default Solver

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

Absolute Tolerance – Tolerance = 0.0001

Обчисліть модель.

Home **-> Study** - Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Line Graph 1**, що містить графіки розв'язку $u(x,t)$ в усі задані моменти часу. Кривих багато, тому цей графік малоінформативний. До того ж, графіки побудовані для обох відрізків [0,10] і [10,20], хоча другий відрізок нас не цікавить, бо він допоміжний. Обчислення на ньому виконувалися з масштабуванням і використовувати розв'язок на ньому недоцільно. Отже треба зобразити графіки лише на «нормальній» частині геометрії, тобто на відрізку [0,10].

1D Plot Group 1 - Line Graph 1:

Selection–Selection = Manual, 1

Щоб видалити індикатор «нескінченного» відрізка, виділіть рядок з цифрою 2 та клацніть мишею по кнопці «-» (мінус), що розташована праворуч секції вибору.

1D Plot Group 1 : LMB > Plot

Графічну групу **1D Plot Group 1** з усіма кривими ми використаємо для побудови анімації.

Створимо дублікат графічної групи **1D Plot Group 1**, в якому зобразимо меншу кількість кривих, наприклад, зобразимо форму струни лише для деяких початкових моментів (на графіку моменти часу зображені на легенді).

1D Plot Group 1 : RMB > Duplicate

1D Plot Group 2:

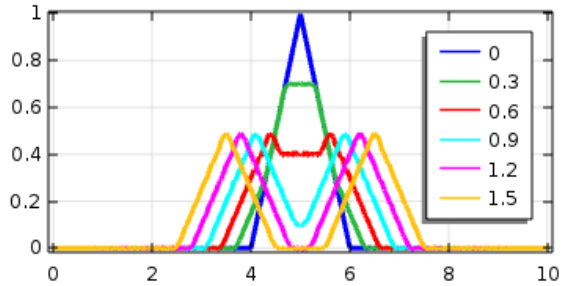
Data –Time selection = Manual

Data –Times (1-121) = range(1,3,16)

1D Plot Group 2 - Line Graph 1:

Legends–Show legends = True

1D Plot Group 2 : LMB > Plot



Щоб краще проаналізувати поведінку струни побудуйте анімацію. Для цього виконайте інструкції

`Results` –> Export –> Animation – Player

Export - Animation 1:

Scene– Subject = `1D Plot Group 1`

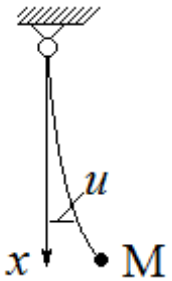
Frames– Frame selection = `All`

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.

3.3. Вільні коливання вагомої нитки з вантажем на кінці.

Розглянемо задачу коливання однорідної вагомої нерозтяжної нитки, яка підвішена за один кінець і несе на іншому кінці вантаж маси M. В стані рівноваги нитка розташована на вертикальній прямій, яку будемо вважати за вісь x.



Малі плоскі коливання вагомої нитки навколо стану рівноваги описуються рівнянням

$$\mu \frac{\partial^2 u}{\partial t^2} = g \frac{\partial}{\partial x}\left\{[M + \mu\,(L - x)]\frac{\partial u}{\partial x}\right\} \quad (0 < x < L) \tag{1}$$

де $u(x,t)$ - горизонтальне зміщення точок струни в момент часу t, L – довжина нитки, M – маса вантажу на кінці, μ – лінійна густина нитки, g – прискорення вільного падіння [2]. Розв'язок рівняння (1) повинен задовольняти двом початковим

$$u(x,0) = \varphi(x), \;\; u'_t(x,0) = \psi(x)\,. \tag{2}$$

і двом граничним умовам [2]

$$u(0,t) = 0, \qquad \left[\frac{\partial^2 u}{\partial t^2} + g\frac{\partial u}{\partial x}\right]_{x=L} = 0 \tag{3}$$

Для розв'язання задачі (1) – (3) в COMSOL Multiphysics можна використати `Classical PDEs - Wave Equation` інтерфейс.

Розв'яжемо задачу, коли початкова швидкість $\psi(x) \equiv 0$, і в початковому стані нитка має прямолінійну форму з рівнянням $\varphi(x) = x/2$. Покладемо також

$L = 1$. Звісно, що при моделюванні в COMSOL Multiphysics вісь x буде розташована горизонтально

За допомогою майстра *створіть* 1D *модель задачі*

File->New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Wave Equation (waeq) -> Time Dependent

Вимкніть в моделі підтримку розмірностей фізичних величин.

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

Задайте параметри моделі.

Global Definitions : RMB > Parameters

В ПВ створеного вузла **Parameters 1** в колонці **Name** вводьте імена параметрів, а в колонці **Expression** – значення. Поля в стовпці **Value** будуть заповнюватися автоматично. Заповніть таблицю параметрів так, як показано нижче.

Parameters

Name	Expression	Value
M	0.5	0.5
mu	10	10
L	1	1

Геометрія моделі складається з відрізка $[0, L]$. Створіть його наступними інструкціями.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1):

`Interval` – Right endpoint = L

Interval 1: LMB > Build All Objects

На створеному відрізку генеруйте 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 повинні стояти одиниця (індикатор відрізка). В розділі **`Equation`** наведено загальний вигляд класичного хвильового рівняння

$$e_a \frac{\partial^2 u}{\partial t^2} + \nabla \cdot (-c \nabla u) = f \,, \; \nabla = \frac{\partial}{\partial x} \,.$$

Коефіцієнти рівняння задаються в полях в нижній частині цього вікна.

`Diffusion Coefficient` – c = g_const*(M+mu*(L-x))

`Source Term` – f = 0

`Absorption Coefficient` – e_a = mu

Цим задається рівняння $\mu \cdot \frac{\partial^2 u}{\partial t^2} - g \frac{\partial}{\partial x} \left\{ [M + \mu (L - x)] \frac{\partial u}{\partial x} \right\} = 0$.

Задайте граничні умови. Умова закріплення $u(0, t) = 0$ - це гранична умова Діріхле.

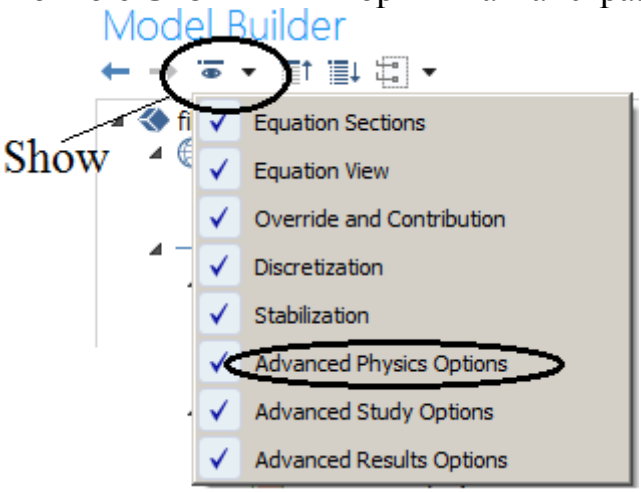
Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,** <> = Manual, 1 (точка x=0)

Dirichlet Boundary Condition – **r** = 0 (умова закріплення)

Однак умова $\left[\frac{\partial^2 u}{\partial t^2} + g\frac{\partial u}{\partial x}\right]_{x=l} = 0$ є нестандартною. В таких випадках в COMSOL Multiphysics можна створювати вузли граничних умов типу Constraint (обмеження). Гранична умова Constraint використовує вираз R, який повинен обертатися на нуль (R=0) і, тим самим, обмежує розв'язок на обраній частині геометрії. Майте на увазі, що використання вузлів типу Constraint повинно бути увімкнено в налаштуваннях майстра моделей. Це виконується встановленням прапорця **Advanced Physics Options** в списку, який відкривається кнопкою Show вгорі вікна майстра.



Перевірте, що опція **Advanced Physics Options** увімкнена і виконайте наступні інструкції

Physics -> **Boundaries** – Constraint

Constraint 1:

Boundary Selection – **Selection,** <> = Manual, 2 (точка x=L)

Constraint – **R** = utt+g_const*ux

Цим задається гранична умова $u''_{tt} + g\,u'_x\big|_{x=L} = 0$.

Задайте початкові умови. Перевірте, що в ПВ вузла **Initial Values 1** в секції вибору обрано індикатор відрізка 1. Потім введіть

Initial Values 1:

Initial Values – **Initial Value for u:** – **u** = x/2

Initial Values – **Initial time derivative of u:** – $\frac{\partial u}{\partial t}$ = 0

Виконайте налаштування розв'язувача, задавши часовий інтервал і похибки обчислень.

Study 1 - Step 1: Time Dependent:

Study Settings – **Times** = range(0,0.025,1.7)

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

Study 1: RMB > Show Default Solver

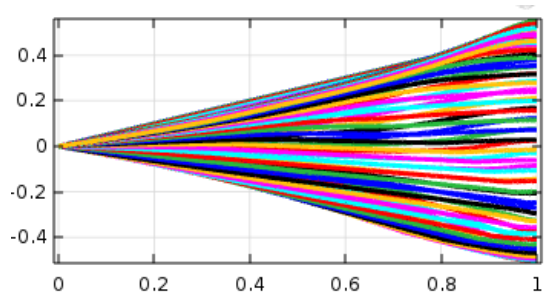
Study 1 – Solver Configurations – Solution 1 – Time-Dependent Solver 1:

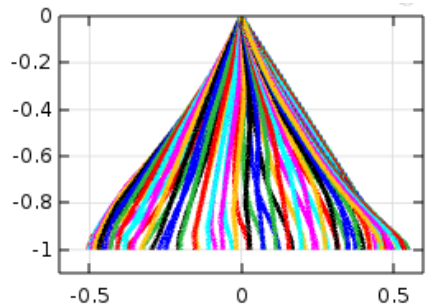
Absolute Tolerance – **Tolerance** = 0.00001

Виконайте обчислення.

`Home -> Study` - Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Line Graph 1**, що містить графіки розв'язку $u(x,t)$ в усі задані моменти часу t_i (наступний рисунок ліворуч).





Але бажано ці графіки побудувати в справжній орієнтації. Для цього можна створити таку ж графічну групу з лінійним графіком, в якому вздовж реальних осей X, Y відкласти переставлені координати, поклавши $Y=-x$, $X=u$.

1D Plot Group 1 : RMB > Duplicate

1D Plot Group 2:

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

`Axis` –**x minimum, x maximum** = `-0.6, 0.6` (два поля)

`Axis` –**y minimum, y maximum** = `-1.1, 0` (два поля)

1D Plot Group 2 - Line Graph 1:

`y-Axis Data` – **Expression** = `-x`

`x-Axis Data` – **Parameter** = `Expression`

`x-Axis Data` – **Expression** = `u`

1D Plot Group 2 : LMB > Plot (попередній рисунок праворуч)

Перед побудовою анімації на графіку бажано відобразити точкову масу у вигляді невеликого диску. Це можна зробити наступними інструкціями.

1D Plot Group 2 - Line Graph 1 : RMB > Duplicate

1D Plot Group 2 - Line Graph 2 :

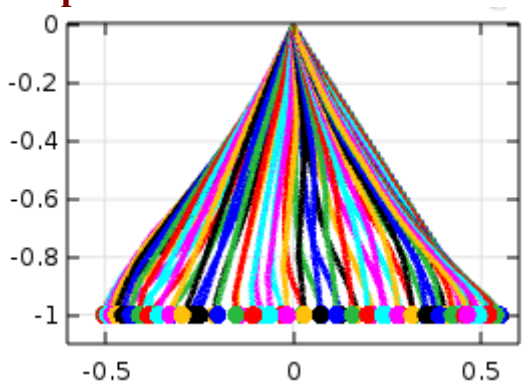
`y-Axis Data` – **Expression** = `-1`

`x-Axis Data` – **Parameter** = `Expression`

`x-Axis Data` – **Expression** = `at1(1,u)`

`Coloring and style` – **Width** = `8` (*розмір точки*)

1D Plot Group 2 - Line Graph 1 : LMB > Plot



Тут ми використали оператор `at1(x0,expr)`. Нагадаємо, що він обчислює вираз `expr` на одновимірному примітиві (в нашому прикладі на відрізку) в точці $x=x_0$. Отже на лінійному графіку **Line Graph 2** для моментів часу $t=t_i$ будуються криві з рівняннями $y=-1,\ x=u(1,t_i)$, тобто точки з цими координатами.

Побудуйте анімацію.

`Results` –> Export –> Animation – Player

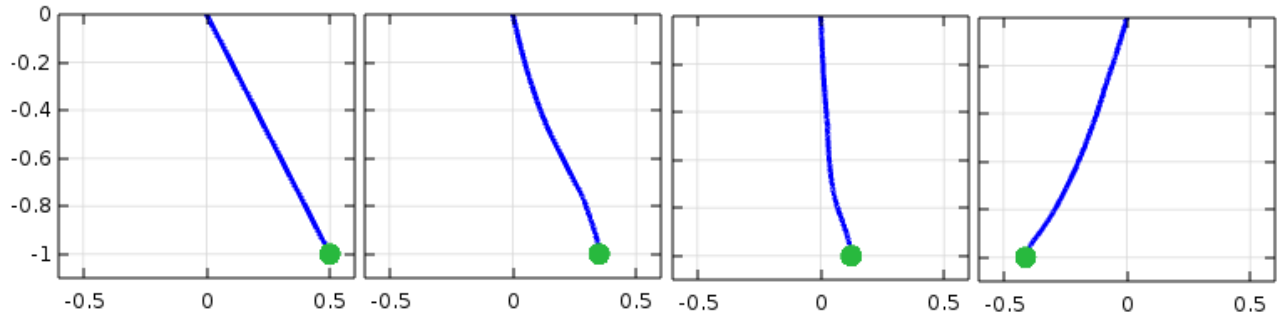
Export - Animation 1:

`Scene`– Subject = `1D Plot Group 2`

`Frames`– Frame selection = `All`

LMB> Generate Frame

Запустіть анімацію, клацнувши мишею по кнопці Play, що розташована над графічним вікном. На наступному рисунку наведено кілька кадрів, на яких зображено нитку у деякі моменти часу.



$t=0$ $t=0.2$ $t=0.35$ $t=0.6$

Визначимо момент часу, в який маса досягає найбільшого відхилу. Для цього використаємо оператор `attimemax(t1,t2,expr1,expr2)`. Він на інтервалі часу `[t1,t2]` знаходить момент 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`.

Derived Values: RMB > Point Evaluation

Derived Values - Point Evaluation 1:

`Data` –**Time selection** = `All`

`Selection` – **Selection, <>** = `Manual, 2` (точка $x=L$)

`Expression` – **Expression** = `attimemax(0,1.7,u,t)`

`Data Series Operation` – **Operation** = `Average`

LMB > Evaluate

В інформаційній зоні створюється таблиця **Table 1**, в якій буде надруковано шукане значення моменту часу `1.5859`.

Average: attimemax(0,1.7,u,t), Point: 2
1.5859

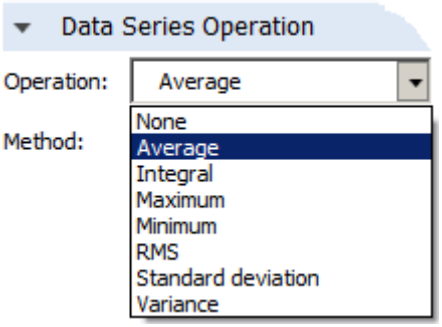
Дійсно, оскільки вираз `attimemax(0,1.7,u,t)` розташований у вузлі **Point Evaluation 1**, то обчислення величини u виконуються в точці з індикатором 2. На інтервалі часу `[0,1.7]` оператор шукає максимум $u(1,t)$, і для знайденого моменту повертає значення виразу `t`, тобто час t_i, в який вираз $u(1,t_i)$ максимальний. Потім це число усереднюється (операція `Average`). Оскільки

воно однакове для всіх моментів часу, то усереднення повертає те ж саме значення, і результат друкується в таблиці.

Зауваження. В `Time Dependent` режимах вузли **Global Evaluation** і **Point Evaluation** обчислюють вираз, заданий в їх полі **Expression** в усі моменти часу (з діапазону поля **Times** вузла **Step 1:Time Dependent**), якщо в полі **`Data Series Operation`** – **Operation** обрано значення `None`. Якщо обрано

`Data Series Operation` – **Operation**=`Average`,

то обчислюється середнє (по всім моментам часу) значення виразу. При цьому в полі **Time selection** повинно стояти `All` або обрано не менше двох моментів часу. Замість `Average`, можна обрати інші операції, які будуть виконуватися над рядом даних, отримуваним зі значень виразу в обрані моменти часу.



Окрім моменту часу корисно знайти величину максимального відхилення вантажу. Це можна зробити тим же операторам `attimemax(0,1.7,u,u)`, який замість значення `t` надрукує значення виразу `u`. Але для такого випадку в системі передбачено використання оператора `timemax(t1,t2,expr)`, який на відрізку часу $t_1 \le t \le t_2$ визначає максимальне значення виразу `expr`.

Derived Values - Point Evaluation 1: RMB > Duplicate

Derived Values - Point Evaluation 2:

`Expression` – **Expression** = `timemax(0,1.7,u)`

LMB > Evaluate –Table 1

Average: attimemax(0,1.7,u,t), Point: 2	Average: timemax(0,1.7,u), Point: 2
1.5859	0.55029

Отже максимальний відхил вантажу u_{max} =`0.55029` досягається в момент t_{max}=`1.5859`.

Зауваження. Для пошуку максимального відхилення вантажу можна в полі **Expression** вузла `Point Evaluation` вказати вираз *u*, а в списку операцій обрати `Maximum`.

Derived Values - Point Evaluation 2: RMB > Duplicate

Derived Values - Point Evaluation 3:

`Expression` – **Expression** = `u`

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

LMB > Evaluate –Table 1

Average: attimemax(0,1.7,u,t), Point: 2	Average: timemax(0,1.7,u), Point: 2	Maximum: Dependent variable u, Point: 2
1.5859	0.55029	0.54933

Отримане значення `0.54933` трохи відрізняється від попереднього тому, що оператор `timemax(0,1.7,u)` знаходить максимум виразу u, використовуючи інтерполяцію його дискретних значень. Використання же операцій `Maximum/Minimum` знаходить екстремум u безпосередньо з дискретного набору значень.

3.4. Розподіл температури в кільці.

Нехай кільце утворено обертанням його нормального перерізу, що є колом малого радіуса, навколо осі, перпендикулярній площині кільця. Поперечний переріз кільця вважається настільки малим, що температуру всіх точок одного перерізу можна вважати однаковою. Початкова температура задана. Задача полягає в визначенні температури кільця в будь який момент часу, коли воно охолоджується завдяки внутрішній теплопровідності і теплообміну через поверхню з навколишнім середовищем.

В якості координати, що визначає положення точки кільця (центру нормального перерізу), оберемо довжину дуги окружності, відлічувану від деякої фіксованої точки. Позначимо цю координату через x. Тоді диференціальне рівняння розповсюдження тепла в кільці матиме вигляд [4]

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} - b(u - u_0), \text{ де } a^2 = \frac{k}{c\rho},\ b = \frac{p\alpha}{c\rho\sigma}, \tag{1}$$

σ - площа поперечного перерізу кільця, k – сталий коефіцієнт внутрішньої теплопровідності, ρ і c – густина і теплоємність речовини кільця, α - коефіцієнт теплообміну, p – периметр перерізу кільця, u_0 – зовнішня температура. При чому в (1) доданок $b(u - u_0)$ відповідає за теплообмін кільця з навколишнім середовищем через його поверхню.

Зауваження. Рівняння (1) придатне до будь якого зігнутого стриженя малого перерізу, вісь якого не має петель.

Розв'яжемо задачу розподілу температури в однорідному ізотропному кільці одиничного радіуса. Нехай теплообмін на поверхні відсутній і задано початковий розподіл температури $u(x,0) = \varphi(x)$ $(\varphi(-\pi) = \varphi(\pi))$. Тоді математична модель задачі буде зводитися до розв'язання рівняння [3]

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} \quad (t > 0,\ -\pi < x < \pi), \tag{2}$$

з початковою умовою

$$u(x,0) = \varphi(x) \tag{3}$$

і додатковими умовами неперервності температури і теплового потоку

$$u(-\pi,t) = u(\pi,t),\ u'_x(-\pi,t) = u'_x(\pi,t) \tag{4}$$

де $x = -\pi$ і $x = \pi$ відповідають спільному перерізу, який прийнято як за початок так і за кінець зігнутого стриженя.

Для розв'язання задачі (2) – (4) в COMSOL Multiphysics можна використати `Classical PDEs - Heat Equation` інтерфейс.

За допомогою майстра *створіть* 1D *модель задачі*

File->New -> Model Wizard -> 1D -> Mathematics - Classical PDEs - Heat Equation (hteq) -> Time Dependent

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Створіть функцію $\varphi(x)$ початкової температури.

Global Definitions: RMB > Functions - Analytic

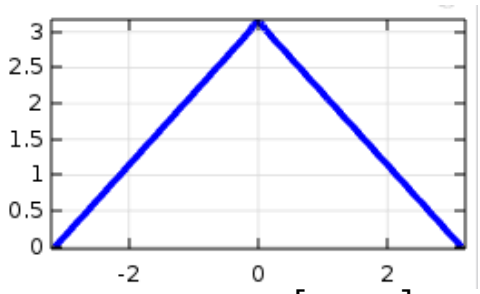
Function name = f0

Definition – **Expression** = pi-abs(x)

Для перевірки виконайте побудову графіка цієї функції на відрізку $[-\pi,\pi]$.

Plot Parameters – **Low limit, Upper limit** = -pi, pi

LMB> Plot



Геометрія моделі складається з відрізка $[-\pi,\pi]$. Створіть його наступними інструкціями.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1):

Interval – **Left endpoint** = -pi

Interval – **Right endpoint** = pi

Interval 1: LMB > Build All Objects

На створеному відрізку *генеруйте* СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** = Finer

LMB> Build All

Задайте параметри рівняння. В ПВ вузла **Heat Equation (hteq) - Heat Equation 1** в секції вибору повинна стояти одиниця (індикатор відрізка). В розділі **Equation** наведено загальний вигляд класичного рівняння теплопровідності

$$d_a \frac{\partial u}{\partial t} + \nabla \cdot (-c \nabla u) = f\,,\ \nabla = \frac{\partial}{\partial x}\,.$$

Коефіцієнти рівняння задаються в полях в нижній частині цієї панелі.

Diffusion Coefficient – **c** = 1

Source Term – **f** = 0

Absorption Coefficient – $\mathbf{d_a}$ = 1

Цим задається рівняння (2).

Задайте початкові умови. Перевірте, що в ПВ вузла **Initial Values 1** в секції вибору обрано 1 – індикатор відрізка. Потім введіть

Initial Values 1:

Initial Values – Initial Value for u: – u = f0(x)

Задайте граничні умови. Граничні умови (4) фактично ототожнюють ліву і праву межу розрахункової області, роблячи її відрізком періодичності шуканої величини *u*. Для задання подібної умови в системі передбачено створення вузлів типу Periodic Condition, які здатні замінити в моделі інші можливі граничні умови.

Physics -> **Boundaries** – Periodic Condition

Periodic Condition 1:

Boundary Selection – Selection, <> = Manual,1,2 (точки $x = \pm\pi$)

Тут в полі **Selection** треба вказати щонайменше дві точки, які будуть ототожнюватися, інакше система не буде сприймати умову. Після створення вузла **Periodic Condition 1** автоматично буде створено вузол **Definitions – Explicit 1**, який визначає межу-джерело і межу призначення. Вам непотрібно турбуватися про цей вузол.

Виконайте налаштування розв'язувача.

Study 1 - Step 1: Time Dependent:

Study Settings –Times =

range(0,0.01,0.25) range(0.275,0.025,1.5)

Спочатку процес остигання йде швидко. Тому часовий інтервал ми розділили нерівномірно.

Виконайте обчислення.

Home **-> Study** - Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Line Graph 1**, що містить графіки розв'язку $u(x,t)$ в усі задані моменти часу t_i. Кривих багато, тому змінимо налаштування так, щоб показувати кожну десяту криву.

1D Plot Group 1:

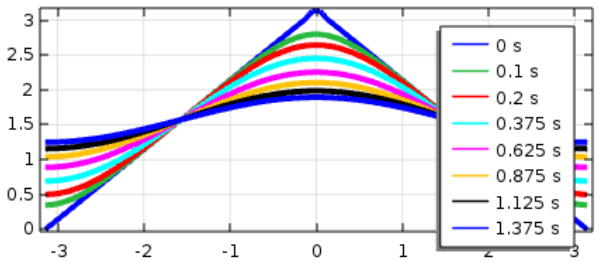
Data –Time selection = Manual

Data –Times (1-76) = range(1,10,76)

1D Plot Group 1 - Line Graph 1:

Legends–Show legends = True

LMB > Plot



Останній графік віднесено до координати *x* вздовж відрізка $[-\pi, \pi]$.

Досить просто побудувати кільце з функціональним забарвленням, відповідним температурі *u*. Для цього в новій одновимірній графічній групі у вузлі Line Graph побудуйте коло по параметричному рівнянню, де в якості параметра буде виступати координата *x* вздовж відрізка $[-\pi, \pi]$ з джерела

даних **Study 1/Solution 1**. Додавши підвузол `Color Expression` ви зможете розфарбувати коло відповідно до бажаного виразу.

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

1D Plot Group 2:

`Data` –Time selection = `First`

RMB > Line Graph

1D Plot Group 2 – Line Graph 1:

`Selection` – Selection, <> = `Manual,1`

`y-Axis Data` – Expression = `sin(x)`

`x-Axis Data` – Parameter = `Expression`

`x-Axis Data` – Expression = `cos(x)`

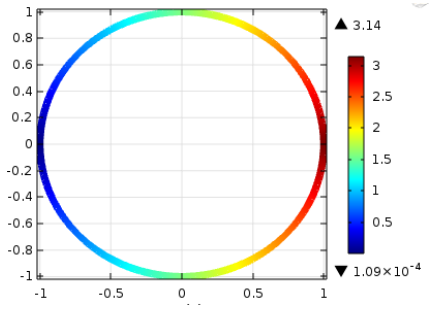
`Coloring and Style` – Line, Width = `Solid, 5` (*два поля*)

1D Plot Group 2 – Line Graph 1: RMB > Color Expression

1D Plot Group 2 - Line Graph 1 - Color Expression 1:

`Expression` – Expression = `u`

LMB > Plot



На рисунку зображено розподіл температури в початковий момент часу, але змінивши значення в полі **1D Plot Group 2: `Data` –Time selection** ви можете обрати будь який момент.

Щоб зобразити графік температури над кільцем знадобиться графічна група `2D Plot Group`. Але для неї потрібно двовимірне джерело даних, наприклад, `Grid 2D`. Створіть його.

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`

`Parameter Bounds – First parameter` – Minimum = `-1`

`Parameter Bounds – Second parameter` – Minimum = `-1`

На площині створеного ДД **Grid 2D 1** ми будемо зображати кільце. Тобто потрібно намалювати криву, і в її точках (при побудові графіка) використати значення розв'язку *u*. Це означає, що потрібно ще одне ДД у вигляді окружності, яке можна створити за допомогою `Parameterized Curve 2D`.

Results – Data Sets: RMB > More Data Sets - Parameterized Curve 2D

Data Sets – Parameterized Curve 2D 1:

`Data` – Data set = `Grid 2D 1`

`Parameter` – Name = `s`

Parameter – **Minimum** = -pi

Parameter – **Maximum** = pi

Expressions – **x:** = cos(s)

Expressions – **y:** = sin(s)

Створення двох допоміжних джерел даних недостатньо. Ще треба вказати, що параметр s довжини дуги створеної параметричної окружності співпадає з параметром x вздовж відрізка $[-\pi,\pi]$ компонента **Component 1**. Щоб ототожнити ці параметри треба побудувати спеціальний оператор, який виконує відображення однієї геометрію на іншу. Такі оператори називають Component Coupling. Нам достатньо створити дуже просте відображення $s = x$.

Component 1 (comp1) – Definitions:

RMB > Component Couplings - General Extrusion

Component 1 (comp1) – Definitions -General Extrusion 1 (genext1):

Source Selection – **Selection, <>** = Manual,1 (*відрізок* $[-\pi,\pi]$)

Destination Map – **x-expression** = s

Тут ім'я s є ідентифікатором параметра (координати) в розрахунковій області адресата (будь-якій геометрії, в якій існує координата з назвою s). Цей параметр s буде ототожнюватися з подовжньою координатою x вздовж відрізка. В полі **Opertor name** залиште ім'я genext1 без змін. Тоді оператор genext1(...) буде задавати відповідність абсцис x точок відрізка $[-\pi,\pi]$ і параметра s в геометрії адресата (в нашому прикладі на колі Parameterized Curve 2D 1). Використання «в геометрії окружності» оператора genext1(u) буде означати обчислення виразу $u(s)$ в точці $s = x$.

Оператор відповідності General Extrusion можна використовувати в будь-яких місцях моделі, і для уточнення перед його іменем слід використовувати префікс comp1 (тобто застосовувати запис comp1.genext1(u)), що представляє ім'я компонента, в якому оператор створений, і з якого вибирається величина u.

Щоб оператор став доступним, переобчисліть модель командою Update Solution, або заново виконайте команду Compute.

Тепер можна будувати бажаний графік.

Results -> 2D Plot Group

2D Plot Group 3:

Data – **Data set** = Parameterized Curve 2D 1

Data – **Time** = 0

RMB > Line

2D Plot Group 3 - Line 1:

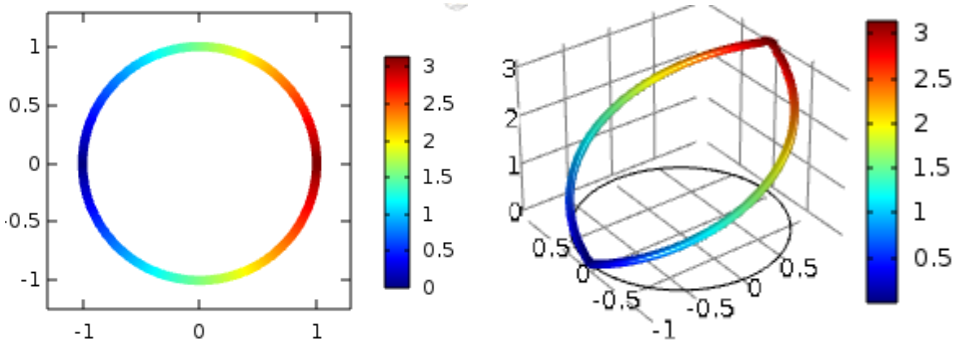
Expression – **Expression** = comp1.genext1(u)

Coloring and style – **Line type** = Tube

Coloring and style – **Tube radius expression** = 0.04

Coloring and style – **Radius scale factor** = True, 1

LMB > Plot (*наступний рисунок ліворуч*)



2D Plot Group 3 – Line 1: RMB > Height Expression

Height Expression 1: **`Axis`** – **Scale factor** = `True ,0.5`

LMB > Plot (*попередній рисунок праворуч*)

2D Plot Group 3 – Line 1: RMB >Duplicate (*малюємо кільце*)

2D Plot Group 3 - Line 2:

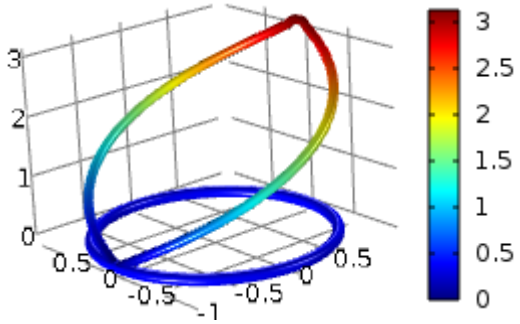
`Expression` – **Expression** = `1`

`Coloring and style` – **Coloring, Color** = `Uniform,Blue`

2D Plot Group 3 – Line 1 - Height Expression 1:

`Expression` – **Height data, Expression** = `Expression, 0` (*два поля*)

LMB > Plot



На рисунку зображено графік температури в початковий момент часу, але змінивши значення в полі **1D Plot Group 3:** **`Data`** –**Time** можна відобразити графік в будь який момент.

3.5. Власні коливання струни.

Нехай дано рівняння $e_a \dfrac{\partial^2 U}{\partial t^2} + \nabla \cdot (-c \nabla U) = f \cdot e^{i\omega t}$, де f залежить лише від просторових координат. Будемо шукати його розв'язок у вигляді $U = u \cdot e^{i\omega t}$, де u – функція тільки просторових координат. Підставивши цей вираз у вихідне рівняння, отримаємо $-e_a \omega^2 u + \nabla \cdot (-c \nabla u) = f$. Якщо покласти $\lambda = -i\omega$, тобто $-\omega^2 = \lambda^2$, то отримаємо неоднорідне рівняння Гельмгольца $\lambda^2 e_a u + \nabla \cdot (-c \nabla u) = f$, яке є початковим в COMSOL при пошуку власних коливань фізичних систем. Для розв'язання таких задач в COMSOL Multiphysics треба обрати `Classical PDEs - Wave Equation (waeq)` інтерфейс, і на наступному кроці побудови шаблону моделі замість `Time Dependent` обрати `Eigenfrequency` тип дослідження (`Study`).

Нехай ν позначає власну частоту коливань системи (eigenfrequency), яка зазвичай вимірюється в герцах. Тоді кутова частота коливань ω обчислюється за формулою $\omega = 2\pi\nu$. Власним значенням (eigenvalue) в термінології розробників називають число $\lambda = -i\omega$. В `Eigenfrequency` дослідженнях розшукуваними змінними є ν (eigenfrequency – власні частоти) та λ (eigenvalue – власні числа), які пов'язані співвідношенням $\lambda = -2\pi i\nu$. Шуканими є також власні форми (eigenmode) коливань. В COMSOL в `Eigenfrequency` дослідженнях для позначення власної частоти ν використовується змінна `freq`, а для власного числа λ – `lambda`. Нагадаємо також про оператор `with(...)`, який можна використовувати для доступу до відповідних власних чисел або частот. Наприклад, щоб дізнатися третю власну частоту можна використати оператор `with(3,freq)`.

В системі існує схожий тип дослідження `Eigenvalue`, який функціонально ідентичний до `Eigenfrequency`. Обидва використовують однаковий солвер, але дані розв'язку зберігаються по-різному: в одному – в змінних `freq`, в іншому – в `lambda`.

Дослідимо власні коливання струни одиничної довжини. Насамперед, при побудові моделі слід вибрати `Eigenfrequency` тип дослідження.

File->New->Model Wizard->1D->
Mathematics- Classical PDEs – Wave Equations (waeq)->
Custom Studies - Eigenfrequency

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Геометрія моделі складається з відрізка [0,1]. Побудуйте його.

`Geometry`->**`Primitives`->** Interval

В ПВ створеного вузла **Interval 1** залиште значення за замовчуванням і завершіть конструювання геометрії натисканням на кнопку Build All Objects .

На створеному відрізку *генеруйте* СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – **Element Size** = `Extra 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

$$\lambda^2 e_a u + \nabla\cdot(-c\nabla u) = f, \quad \nabla = \frac{\partial}{\partial x}.$$

Для розв'язання задачі на власні коливання струни можна використати наступні значення:

`Diffusion Coefficient` – **c** = 1

`Source Term` – **f** = 0

`Absorption Coefficient` – $\mathbf{e_a}$ = 1

Цим задається рівняння $\lambda^2 u - \Delta u = 0$

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = All boundaries

Dirichlet Boundary Condition – **r** = 0

Відкрийте ПВ вузла **Step 1: Eigenfrequency**. В полі **Desired number of eigenfrequencies** вказується бажана для визначення кількість власних частот. Залиште в ньому значення 6. Поле **Search for eigenvalues around** вказує число, в околі якого будуть обчислюватися власні значення. Залиште в ньому значення 0.

Виконайте обчислення.

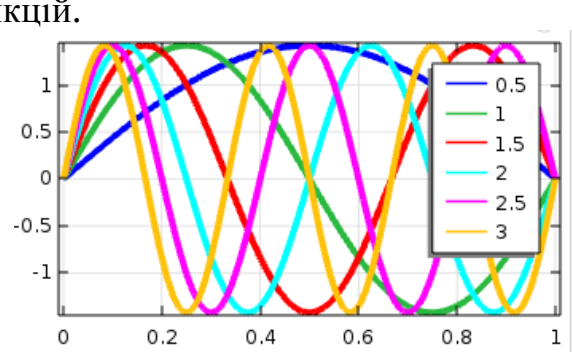
Home -> **Study** - Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Line Graph 1**, що містить форми перших шести власних функцій задачі. Додайте на графік легенду:

1D Plot Group 1 - Line Graph 1:

Legends – **Show legends** = True

На легенді будуть надруковані значення власних частот, що відповідають графікам власних функцій.

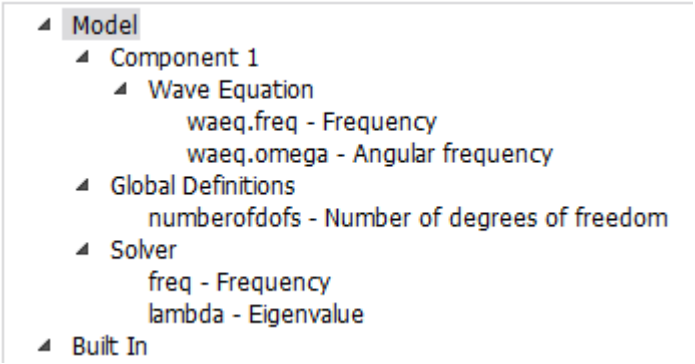


Для нашої задачі власні частоти f_k обчислюються за формулою $f_k = \dfrac{k}{2L}$, де L - довжина струни, а $k = 1,2,\ldots$ номер гармоніки. Як бачите, вони співпадають.

Надрукуємо значення кутової частоти і власних чисел. Для цього в гілці **Derived Values** створіть вузли типу Global Evaluation. В їх полях **Expression** слід ввести бажані вирази, або вибрати їх з вікна готових величин.

Results -> Derived Values : RMB > Global Evaluation

В ПВ вузла **Derived Values - Global Evaluation 1** над полем **Expression** клацніть по кнопці Replace Expression. Відкриється вікно готових величин, з якого подвійним клацанням можна обрати потрібну.



Зверніть увагу, що кутову частоту треба вводити з префіксом (іменем «фізики»), наприклад, `waeq.omega`. В той же час власні числа слід використовувати без префікса, тобто слід використовувати просто ідентифікатор `lambda`. Виконайте подвійне клацання по рядку `waeq.omega` – `Frequency`. Цим ви заповните поле **Expression**.

Results -> Derived Values – Global Evaluation 1:

`Expression` – **Expression** = `waeq.omega`

Надрукуйте значення кутової частоти.

LMB > Evaluate

Щоб додати в створену таблицю **Table 1** значення змінної `lambda`, для неї потрібно побудувати схожий вузол.

Results -> Derived Values : RMB > Global Evaluation

Derived Values – Global Evaluation 2:

`Expression` – **Expression** = `lambda`

LMB > Evaluate – Table 1

Останньою командою ви обираєте назву вже створеної таблиці **Table 1**.

Eigenfrequency	Angular frequency	Eigenvalue
0.50000	3.1416	-3.1416i
1.0000	6.2832	-6.2832i
1.5000	9.4248	-9.4248i
2.0000	12.566	-12.566i
2.5000	15.708	-15.708i
3.0000	18.850	-18.850i

Як бачите кутова частота (angular frequancy) ω пов'язана з власною частотою (eigenfrequency) ν співвідношенням $\omega = 2\pi\nu$, а власне значення (eigenvalue) λ - співвідношенням $\lambda = -2\pi i\nu$. З цих формул зрозуміло, що при дійсних значеннях власних частот ν власні значення λ будуть чисто уявними.

Зауваження 1. Якщо в моделі використати грубішу СкЕ сітку, наприклад, `Finer`, то в значеннях частот з'являться невеликі похибки. Це буде зрозуміло з легенди на графіку власних функцій у вузлі **1D Plot Group 1 - Line Graph 1**.

Зауваження 2. Типи дослідження `Eigenfrequency` та `Eigenvalue` ігнорують будь-які джерела зовнішнього впливу, тобто величину `f` в рівнянні $\lambda^2 e_a u + \nabla\cdot(-c\nabla u) = f$, якщо вона не залежать від шуканої змінної. Наприклад, виконайте обчислення моделі, змінивши значення `f` на сталу.

Wave Equation (waeq) - Wave Equation 1:

`Source Term` – **f** = `1`

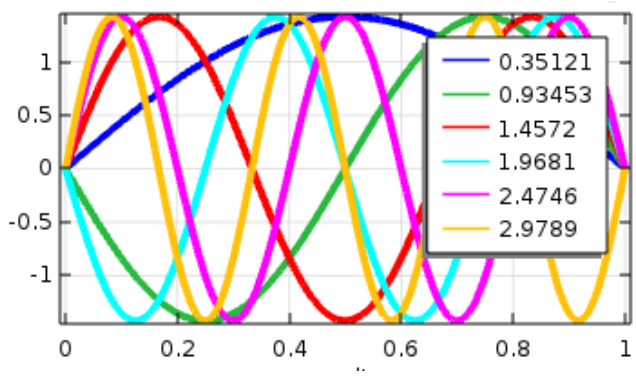
Study 1 : LMB > Compute

Графіки у вузлі **1D Plot Group 1** не зміняться, тобто значення f системою проігноровано. Але спробуйте тепер виконати обчислення для $f = 5\cdot u$.

Wave Equation (waeq) - Wave Equation 1:

`Source Term` – **f** = `5*u`

Study 1 : LMB > Compute



Як бачите, зовнішній вплив `f(u)`, що залежить від розв'язку `u`, змінює власні частоти.

3.6. Телеграфне рівняння.

Диференціальне рівняння розповсюдження хвиль вздовж довгої лінії, яка має самоіндукцію L, ємність C, опір R та електричний витік G, які припадають на одиницю довжини, відносно функції струму $I(x,t)$ має вигляд [4]:

$$\frac{\partial^2}{\partial x^2} I = LC \frac{\partial^2}{\partial t^2} I + (RC + GL) \frac{\partial}{\partial t} I + GRI$$

а відносно потенціалу $V(x,t)$:

$$\frac{\partial^2}{\partial x^2} V = LC \frac{\partial^2}{\partial t^2} V + (RC + GL) \frac{\partial}{\partial t} V + GRV$$

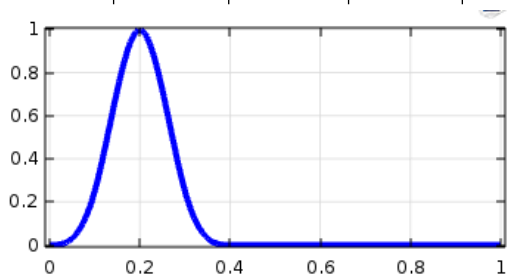
Обидва рівняння мають схожу форму і можуть бути представлені у вигляді

$$u''_{tt} + (A+B)\, u'_t + A \cdot B\, u = C^2 u''_{xx}, \qquad (1)$$

При відсутності опору $R=0$ і витоку $G=0$ отримуємо $A=0, B=0$ і рівняння передавальної лінії (1) перетворюється на хвильове рівняння.

Розв'яжемо рівняння (1) при $C=1, A=0.25, B=0.25$ на відрізку [0,1] і граничних умовах $u'_x(0,t)=0, u'_x(1,t)=0$. Нехай $u'_t(x,0)=0$ і початковий профіль сигналу $u(x,0)=\varphi(x)$ має вигляд

$$\varphi(x) = 125|x|^3 - 500|x-0.1|^3 + 750|x-0.2|^3 - 500|x-0.3|^3 + 125|x-0.4|^3$$



Для розв'язання поставленої задачі в COMSOL Multiphysics нам знадобиться `Coefficient Form PDE (c)` інтерфейс.

Створіть 1D *модель задачі*

File->New -> Model Wizard -> 1D -> Mathematics - PDE Interfaces - Coefficient Form PDE (c) -> Time Dependent

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Створіть функцію $\varphi(x)$ початкового профілю хвилі.

Global Definitions: RMB > Functions - Analytic

Function name = `f0`

`Definition` – **Expression** = `125*abs(x)^3-500*abs(x-0.1)^3+750*abs(x-0.2)^3-500*abs(x-0.3)^3+125*abs(x-0.4)^3`

Ця функція поза відрізком `[0,0.4]` дорівнює нулю. Для перевірки виконайте побудову графіка функції `f0(x)` на відрізку `[0,1]`.

`Plot Parameters` – **Low limit, Upper limit** = `0, 1`

LMB> Plot (*попередній графік*)

Задайте параметри моделі.

Global Definitions : RMB > Parameters

Заповніть таблицю параметрів так, як показано на рисунку.

Parameters

Name	Expression	Value
C	1	1
A	0.25	0.25
B	0.25	0.25

Геометрія моделі складається з відрізка `[0,1]`. Створіть його.

`Geometry->`**`Primitives->`** Interval

Interval 1: LMB > Build All Objects

На створеному відрізку *генеруйте* СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – **Element Size** = `Finer`

LMB> Build All

Задайте параметри рівняння. В ПВ вузла **Coefficient Form PDE (c) - Coefficient Form PDE 1** в секції вибору повинна стояти одиниця – індикатор відрізка [0,1]. В розділі **`Equation`** наведено загальний вигляд коефіцієнтної форми ДРЧП

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot (-c \nabla u - \alpha u + \gamma) + \beta \cdot \nabla u + a u = f, \ \nabla = \frac{\partial}{\partial x}.$$

Задайте коефіцієнти рівняння в полях нижньої частині цієї панелі.

`Diffusion Coefficient` – **c** = `C^2`

`Absorption Coefficient` – **a** = `A*B`

`Source Term` – **f** = `0`

`Mass Coefficient` – $\mathbf{e_a}$ = `1`

`Damping or Mass Coefficient` – $\mathbf{d_a}$ = `A+B`

Решту коефіцієнтів залиште нулями: $\alpha = 0, \beta = 0, \gamma = 0$. Цим задається рівняння (1).

Задайте початкові умови. Перевірте, що в ПВ вузла **Initial Values 1** в секції вибору обрано одиницю – індикатор відрізка. Потім введіть

Initial Values 1:

Initial Values – Initial Value for u:– u = `f0(x)`

Граничні умови $u'_x(0,t)=0, u'_x(1,t)=0$ було задано автоматично у вузлі **Zero Flux 1** при побудові моделі. Тому ніяких змін робити непотрібно.

Виконайте налаштування розв'язувача, задавши часовий інтервал і похибки обчислень.

Study 1 - Step 1: Time Dependent:

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

Study Settings –Relative tolerance = `True, 0.0001`

Study 1: RMB > Show Default Solver

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

Absolute Tolerance – Tolerance = `0.0001`

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Line Graph 1**, що містить графіки розв'язку $u(x,t)$ в усі задані моменти часу t_i. Змініть налаштування так, щоб на графіку було зображені тільки деякі криві.

1D Plot Group 1:

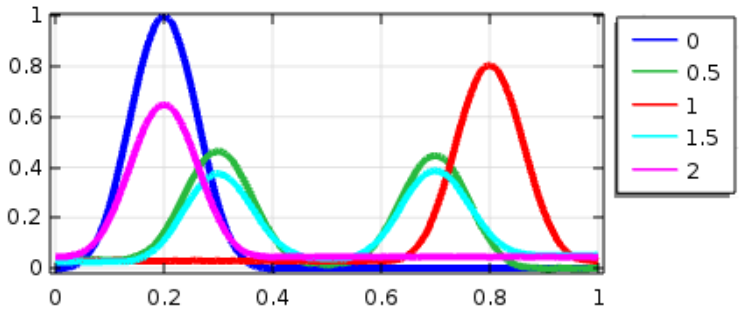
Data –Time selection = `Manual`

Data –Times (1-201) = `1,51,101,151,201`

1D Plot Group 1 - Line Graph 1:

Legends–Show legends = `True`

LMB > Plot

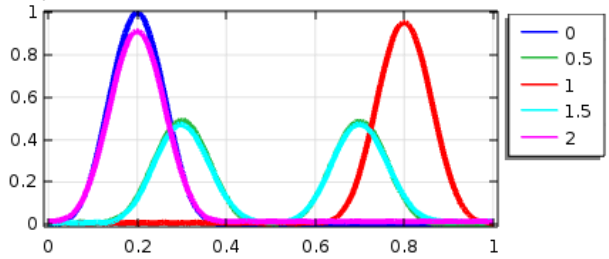


Початковий профіль ділиться навпіл і розповсюджується в різні боки. Рухомий вліво профіль відбивається від лівого кінця і починає просуватися вправо. В момент `t=0.5` направо розповсюджуються обидві хвилі. Червона крива (`t=1`) має форму початкового сигналу з трохи меншою амплітудою і утворена рухомими в різні боки хвилями. Сигнал в момент часу `t=2` повертається на початкове місце з меншою амплітудою. Щоб спостерігати цей процес, побудуйте самостійно анімацію розповсюдження хвиль.

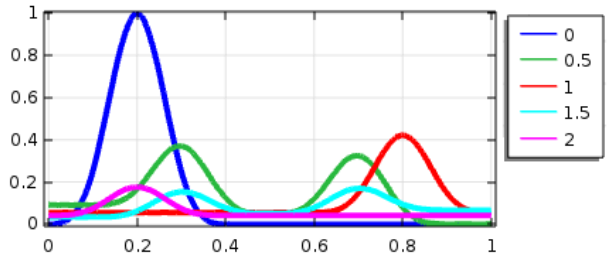
Подивимось, як рухається імпульс, коли передавальна лінія має менші втрати. Покладіть в таблиці параметрів $A=0.05, B=0.05$ і обчисліть модель. Наступний рисунок ліворуч зображає форму імпульсу в ті ж самі моменти часу, що і попередній графік. Як бачимо в момент `t=2` профіль лише трохи зменшив свою амплітуду.

Нехай тепер передавальна лінія має більші втрати. Покладіть в таблиці параметрів $A=1, B=1$ і обчисліть модель. На наступному рисунку праворуч

зображено отримані форми імпульсу. При більших значеннях параметрів A, B імпульс згасає значно сильніше.



$A = 0.05, B = 0.05$



$A = 1, B = 1$

3.7. Вісесиметричні коливання нескінченної тремтівки.

Розв'яжемо методом скінченних елементів ДРЧП

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} \quad (0 < r < \infty) \tag{1}$$

з початковими умовами $u(r,0) = \varphi(r), u'_t(r,0) = 0$, яке описує вісесиметричні коливання нескінченної тремтівки з заданою початковою формою і початковою швидкістю. По суті це еквівалент задачі

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \quad (-\infty < x < \infty, -\infty < y < \infty) \tag{2}$$

з початковими умовами $u(x,y,0) = \varphi\left(\sqrt{x^2+y^2}\right), u'_t(x,y,0) = 0.$

Розв'яжемо задачу при $\varphi(r) = \begin{cases} \sqrt{1-r^2}, & 0 \le r < 1 \\ 0, & r \ge 1 \end{cases}.$

Створіть одновимірну вісесиметричну (`1D Axisymmetric`) *модель задачі*

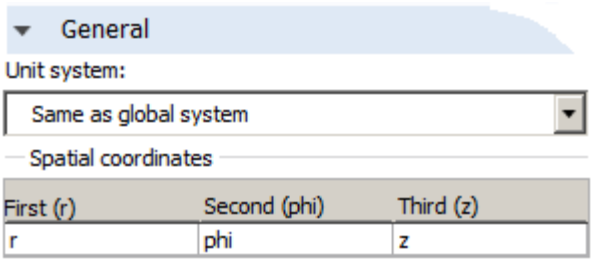
File->New->Model Wizard->1D Axisymmetric->
Mathematics – PDE Interfaces – Coefficient From PDE(c) ->
TimeDependent

Зауважимо, що в режимі `1D Axisymmetric` в дереві досліджень `Study` немає гілки `Classical PDEs`.

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 циліндричні координати (r, φ, z).

Нескінечна тремтівка утворюється обертанням нескінченого променя навколо вісі z, який ми змоделюємо складеним з трьох частин На першій, відрізку [0,1], буде задано ненульовий початковий профіль, на другому відрізку [1,4] – нульовий. Третій відрізок [4,5] нам знадобиться, щоб призначити його «нескінченним». Отже геометрію моделі побудуємо як об'єднання трьох відрізків: $[0,1] \cup [1,4] \cup [4,5]$.

Geometry->**Primitives->** Interval

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i2):

Interval – **Left endpoint** = 1

Interval – **Right endpoint** = 4

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i3):

Interval – **Left endpoint** = 4

Interval – **Right endpoint** = 5

Interval 3: LMB > Build All Objects

Останній відрізок [4,5] позначимо як нескінченний, тобто такий, на якому буде виконуватися масштабування. Для цього виконайте інструкції

Definitions -> **Coordanate Systems** – Infinite Element Domain

Definitions - Infinite Element Domain 1 (ie1):

Domain Selection – **Selection** = Manual,3 (відрізок [4,5])

На створених відрізках *генеруйте* СкЕ *сітку* з достатньо дрібними елементами, задавши їх максимальний розмір.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Sequence type** = User-controlled mesh

Mesh 1 - Size:

Element Size – **Custom** = True

Element Size Parameters – **Maximum element size** = 0.01

LMB> Build All

Задайте параметри рівняння. В ПВ вузла **Coefficient Form PDE (c) - CoefficientFromPDE 1** в секції вибору повинні стояти індикатори всіх трьох відрізків. В розділі **Equation** наведено загальний вигляд коефіцієнтної форми ДРЧП

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot (-c \nabla u - \alpha u + \gamma) + \beta \cdot \nabla u + \mathrm{a}\, u = f\,,\ \nabla = \frac{\partial}{\partial r}.$$

Задайте коефіцієнти рівняння в полях нижньої частині цієї панелі.

Diffusion Coefficient – **c** = 1

Absorption Coefficient – **a** = 0

Source Term – **f** = 0

Mass Coefficient – $\mathbf{e_a}$ = 1

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Convection Coefficient – **β** =-1/r

Коефіцієнти α і γ залиште незмінними, тобто $\alpha = 0, \gamma = 0$. Цим задається рівняння (1).

Задайте початкові умови. Початковий профіль $u(r,0) = \sqrt{1-r^2}$ ми будемо задавати на першому відрізку [0,1]. На двох інших відрізках зміщення точок в нульовий момент часу дорівнює нулю. Щоб задати відмінні вирази на різних ділянках геометрії потрібно мати два вузли типу `Initial Values`. Створимо в моделі другий такий вузол і налаштуємо його.

Coefficient Form PDE (c) : RMB > Initial Values

Coefficient Form PDE (c) - Initial Values 2 :

`Domain Selection` – **Selection** = `Manual,2,3` (відрізки [1,4] і [4,5])

`Initial Values` – **`Initial Value for u:`** – **u** = `0`

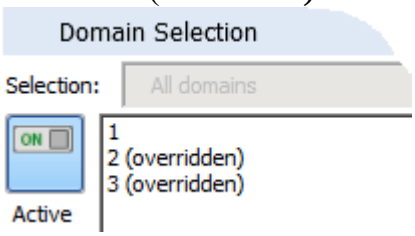
Тепер налаштуємо перший вузол.

Coefficient Form PDE (c) - Initial Values 1 :

`Domain Selection` – **Selection** = `Manual,1` (відрізок [0,1])

`Initial Values` – **`Initial Value for u:`** – **u** = `sqrt(1-r^2)`

Зверніть увагу, що в секції вибору вузла **Initial Values 1** вказані всі три вузли, але умови на двох останніх замінені (overriden).



Зауваження. На всіх трьох відрізках початкову умову $\varphi(r)$ можна було б задати одночасно єдиним виразом `sqrt((r<1)*(1-r^2))`.

Граничні умови задавати непотрібно, бо в початку координат (в точці `r=0`) воно непотрібно через вісесиметричність, а в точці `r=5` – через те, що вона імітує нескінченність.

Виконайте налаштування розв'язувача, задавши часовий інтервал і похибки обчислень.

Study 1 - Step 1: Time Dependent:

`Study Settings` – **Times** = `range(0,0.01,4)`

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

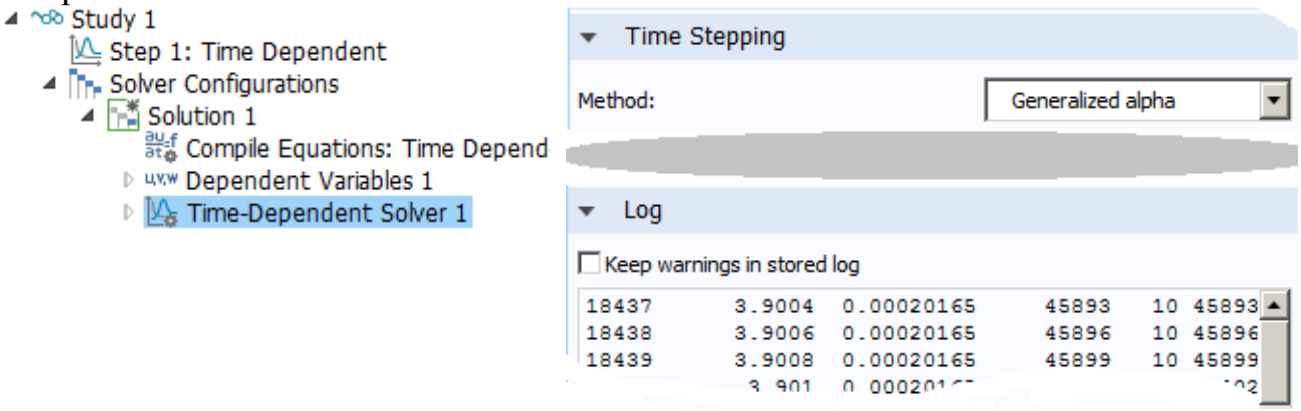
Study 1: RMB > Show Default Solver

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

`Absolute Tolerance` – **Tolerance** = `0.000001`

Зауваження. При розв'язанні хвильових рівнянь чисельний метод `Generalized alpha` зазвичай краще, ніж встановлений за замовчуванням `BDF` метод. Для зміни розв'язувача у ПВ вузла **Time-Dependant Solver 1** в розділі **`Time Stepping`** в списку **Method** можна обрати цей метод. Він не потребує ручного задання кроку по часу, і у вікні в розділі **`Log`** після виконання обчислень друкує таблицю відповідності номерів кроків і моментів часу, які їм

відповідають. Це корисно, бо часовий крок розв'язувач `Generalized alpha` обирає самостійно.



В нашому прикладі ми залишимо `BDF` метод.

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Line Graph 1**, що містить графіки розв'язку $u(r,t)$ в задані моменти часу t_i. Кривих багато, до того ж, графіки побудовані для всіх трьох відрізків [0,1], [1,4] і [4,5], хоча останній відрізок нас не цікавить, бо він допоміжний. Обчислення на ньому виконувалися з масштабуванням і використовувати розв'язок на ньому недоцільно. Отже треба зобразити графіки лише в «зоні інтересу», тобто на відрізках [0,1] і [1,4].

1D Plot Group 1 - Line Graph 1:

`Selection`–Selection = `Manual, 1, 2`

Щоб видалити індикатор «нескінченного» відрізка, виділіть рядок з цифрою 3 та клацніть мишею по кнопці «-» (мінус), що розташована праворуч секції вибору.

Графічну групу **1D Plot Group 1** з усіма кривими ми використаємо для побудови анімації. Щоб на кадрах анімації відображався час треба в «базисній» графічній групі показати легенду.

`Legends`–Show legends = `True`

1D Plot Group 1 : LMB > Plot

Тепер для спостерігання за розв'язком можна побудувати анімацію.

`Results` –> Export –> Animation – Player

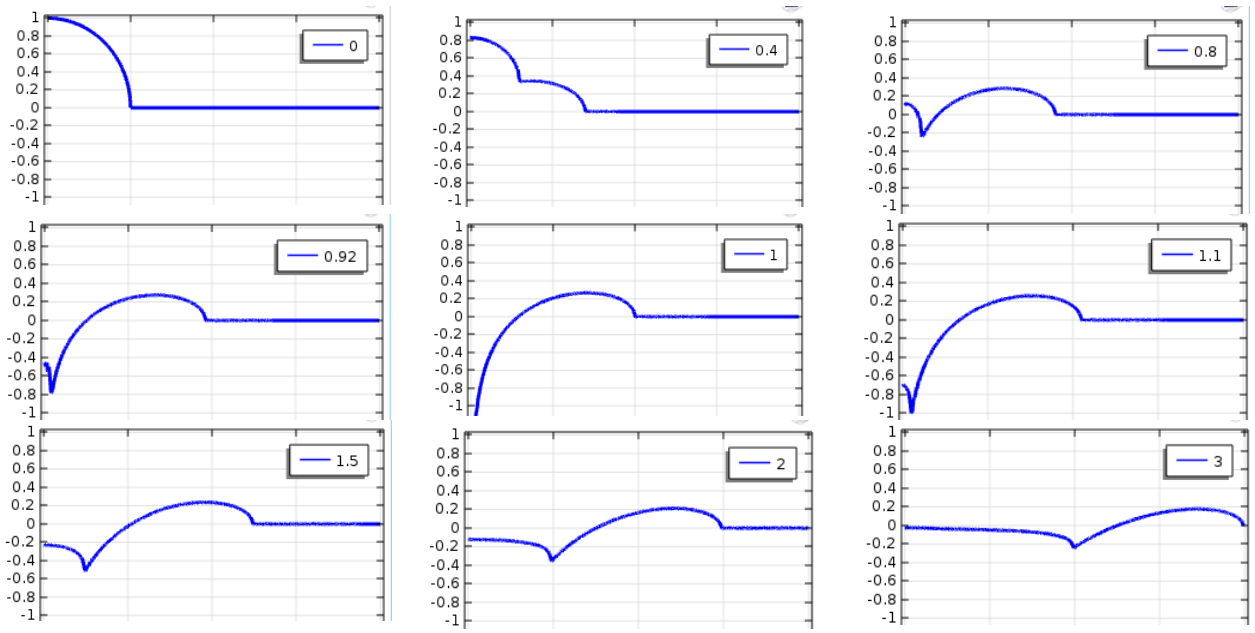
Export - Animation 1:

`Scene`– Subject = `1D Plot Group 1`

`Frames`– Frame selection = `All`

LMB> Generate Frame

Запустіть анімацію, клацнувши мишею по кнопці Play . На наступному рисунку наведено кілька кадрів, на яких зображено радіальний переріз тремтівки у деякі моменти часу. Вони утворюються з використанням кнопки «фотоапарат» графічного вікна, яка розміщує копію графіка в буфер обміну або зберігає зображення в графічний файл.



Зауваження. Існуючий точний розв'язок розглядуваної задачі показує, що $u(0,1)=-\infty$ (мінус нескінченність лише в точці $x=0$ в єдиний момент часу $t=1$).

Після виконання обчислень в ДМ автоматично було створено ще графічну гілку **2D Plot Group 2**, яка відображає графік розв'язку $u(x,y,t)$ для моменту часу, обраного в списку **Data – Time(s)**. Оберемо для відображення нульовий момент часу.

2D Plot Group 2 : Data – Time (s) = 0

Встановимо однакові для всіх моментів часу мінімальне і максимальне значення діапазону відображуваних кольорів

2D Plot Group 2 - Surface 1 :

Range – Manual color range = True

Range – Manual color range – Minimum, Maximum = -1, 1

Скорегуємо джерело даних **Revolution 1D 1**, щоб воно представляло повне коло.

Data Sets - Revolution 1D 1:

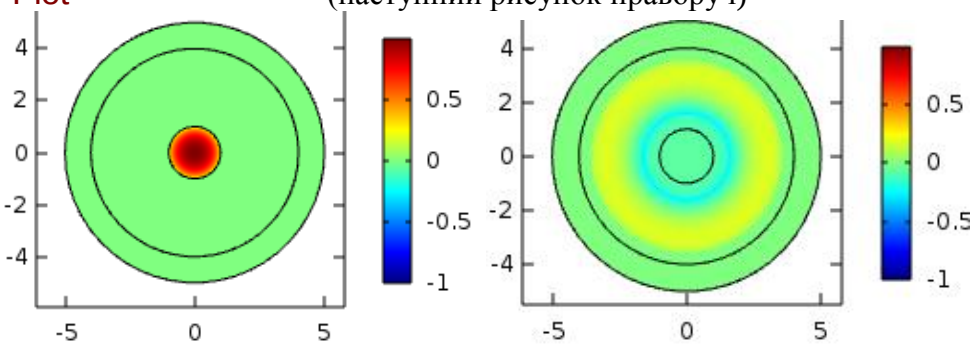
Revolution Layers – Start angle, Revolution angle = 0, 360

2D Plot Group 2 : LMB> Plot (наступний рисунок ліворуч)

Оберемо інший момент часу, наприклад,

2D Plot Group 2 : Data – Time (s) = 2.6

LMB> Plot (наступний рисунок праворуч)



Побудуємо тривимірне зображення тремтівки.

2D Plot Group 2 : RMB> Duplicate

2D Plot Group 3 : `Data` – **Time (s)** = `0.95`

2D Plot Group 3 - Surface 1 :

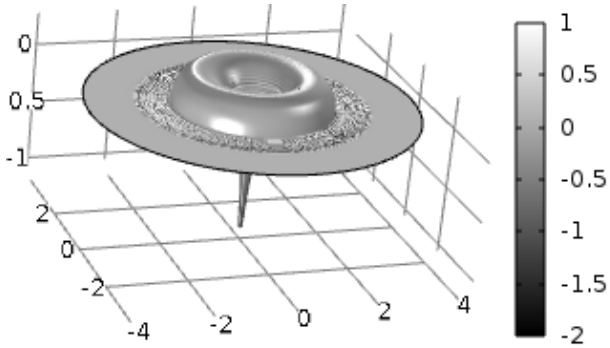
`Coloring and style` – **Color Table** = `GrayScale`

2D Plot Group 3 - Surface 1 : RMB> Height Expression

Height Expression 1:

`Axis` – **Scale factor** = `True ,3`

LMB > Plot



Побудуйте самостійно анімацію розповсюдження хвиль по площині тремтівки.

3.8. Мінімальна поверхня.

Розв'яжемо в одиничному колі $\Omega = \{(x, y):\ x^2 + y^2 \le 1\}$ ДРЧП наступного виду

$$-\nabla \cdot \left(\frac{1}{\sqrt{1+|\nabla u|^2}} \nabla u \right) = 0,$$

зі значеннями $u = |x|$ на межі $\partial\Omega$ області Ω. Воно визначає форму мінімальної поверхні, край якої знаходиться на заданій висоті над границею кола.

За допомогою майстра створіть 2D модель задачі

File->New -> Model Wizard -> 2D -> Mathematics - Classical PDEs - Poisson's Equation (poeq) -> Stationary

Геометрія задачі складається з кола одиничного радіуса.

`Geometry->`**`Primitives->`** Circle

В ПВ створеного вузла **Circle 1** залиште значення параметрів за замовчуванням і завершіть побудову геометрії натисканням на кнопку

.

На створеному колі *генеруйте* СкЕ *сітку* з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Задайте рівняння. Перевірте, що в ПВ гілки **Poisson's Equation (poeq)** в розділі **`Domain Selection`** в секції вибору стоїть 1, тобто обрано коло. В дереві моделі виділіть вузол **Poisson's Equation 1**. Справа відкриється його панель. В розділі **`Equation`** цієї панелі наведено загальний вигляд рівняння Пуассона.

$$\nabla\cdot(-c\nabla u)=f\,,\quad \nabla=\left[\frac{\partial}{\partial x},\frac{\partial}{\partial y}\right].$$

Задайте коефіцієнти рівняння

Diffusion Coefficient – **c** = `1/sqrt(1+ux^2+uy^2)[m]`

Source Term – **f** = `0`

Задайте граничні умови. Для цього виконайте інструкції:

`Physics` -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = `All boundaries`

Dirichlet Boundary Condition – **r** = `abs(x[1/m])`

Цим задаються значення шуканої функції на контурі області.

Виконайте обчислення.

Study 1 : LMB > Compute

Після завершення обчислень в графічній області вузла **Results - 2D Plot Group 1 – Surface 1** буде побудована кругова область з функціональним забарвленням. Додайте для неї вертикальне зміщення.

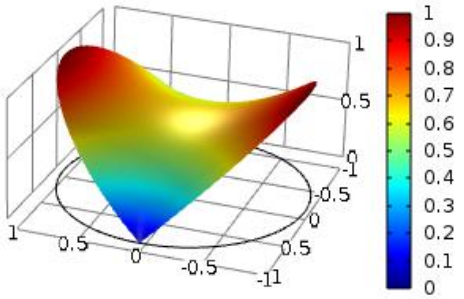
2D Plot Group 1 – Surface 1: RMB > Height Expression

Height Expression 1:

Axis – **Scale factor** = `True ,1`

2D Plot Group 1: LMB > Plot

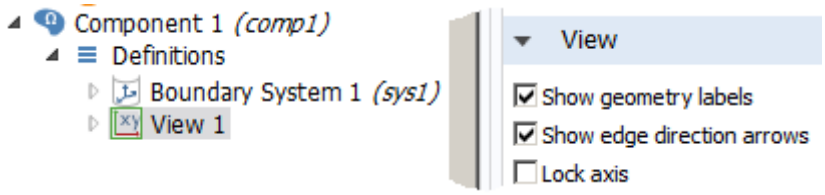
Зображення мінімальної поверхні набуде наступного вигляду

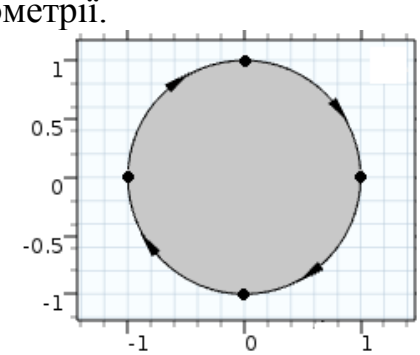


Збережіть файл з ім’ям, наприклад, `minsurf01.mph`, але не закривайте його.

Задання граничних умов, використовуючи просторові координати x, y, зручно, але не завжди можливо (в нашому прикладі $u|_{\partial\Omega}=|x|$). При постановці двовимірних задач часто задаються параметричні рівняння границі області $x=x(\xi)$, $y=y(\xi)$ і, відповідно до цього, граничне значення відноситься до параметра ξ вздовж границі, наприклад, $u|_{\partial\Omega}=f(\xi)$. Для реалізації подібної можливості в COMSOL Multiphysics використовуються допоміжні геометричні змінні, які доступні в граничних точках областей. Наприклад, коли ви використовуєте вирази, які задаються в граничних точках двовимірної області, то крім просторових координат x, y можна використовувати ім'я s. Зазвичай межа двовимірної області складається з ділянок кривих, які в робочому просторі системи зображуються кривими Без’є, і s представляє параметр в рівняннях таких ділянок. Здебільшого, уздовж кожного криволінійного відрізка/ділянки двовимірного контура s змінюється від 0 до 1.

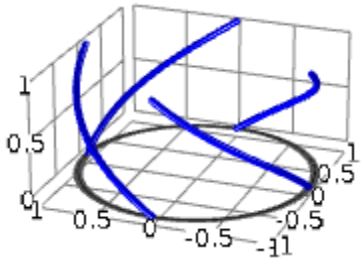
Продемонструємо використання змінної *s*, що пов'язана з геометрією межі моделі. В нашому прикладі границя кола складається з чотирьох дуг, вздовж кожної з яких параметр *s* змінюється від 0 до 1. Щоб правильно використовувати цей параметр треба знати де знаходиться початок відліку. Система може відобразити напрям зростання параметра *s*, якщо у ПВ вузла `Component 1(comp1)-Definitions-View 1` встановити прапорець **Show edge direction arrows** і при активному вузлі **Geometry** над графічним вікном натиснути кнопку Select Boundaries. Корисно буде тут же встановити прапорець **Show geometry labels**. Тоді в графічному вікні у відповідних місцях будуть показуватися індикатори елементів геометрії.





На попередньому рисунку праворуч чітко видно точки початку і кінця кожної дуги, а також напрям зростання параметра *s* вздовж цих дуг.

Щоб наочніше продемонструвати зв'язок параметра *s* з дугами побудуємо графік його значення над контуром кола.



Цей графік побудовано наступними інструкціями.

Results – Data Sets: RMB > Edge 2D

Data Sets – Edge 2D 1:

`Selection` – **Selection** =`All boundaries`

`Results` -> 2D Plot Group

2D Plot Group 2: RMB > Line

2D Plot Group 2 – Line 1:

`Data` – **Data set** = `Edge 2D 1`

`Expression` – **Expression** = `s`

`Coloring and style` – **Line type** = `Tube`

`Coloring and style` – **Tube radius expression** = `0.2`

`Coloring and style` – **Coloring, Color** = `Uniform,Blue`

2D Plot Group 2 – Line 1: RMB > Height Expression

Height Expression 1:

`Axis` – **Scale factor** = `True ,1`

2D Plot Group 2: RMB > Line (зобразимо безпосередньо граничне коло)

2D Plot Group 2 – Line 2:

Data – **Data set** = Study 1/Solution 1

Expression – **Expression** = 1

Coloring and style – **Line type** = Tube

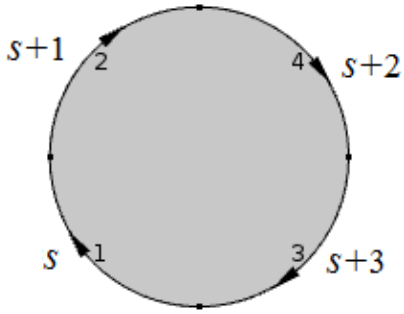
Coloring and style – **Tube radius expression** = 0.3

Coloring and style – **Coloring, Color** = Uniform,Black

2D Plot Group 2: LMB > Plot (попередній рисунок)

Інколи бажано мати єдиний параметр, який буде відноситися до всієї границі розрахункової області. Для цього можна створити змінну моделі, наприклад, з ім'ям *S*, яку на різних ділянках границі пов'язати з параметром *s* різними виразами. Щоб це зробити, в моделі потрібно створити різні вузли Variables, в кожному з яких задати змінну зі спільним ім'ям S, але визначити у вузлах різні ділянки дії. Якщо для S задати в цих вузлах власний вираз, то він буде діяти лише на відповідній ділянці. Коли в моделі ви будете звертатися до змінної S, система буде перевіряти ділянку і обирати потрібний вираз для її обчислення.

На наступному рисунку зображені дуги окружності і відповідні вирази, які ми використаємо в нашому прикладі.



Наведемо інструкції, які створюють змінну S.

Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 1:

Geometric Entity Selection – **Geometric entity level** = Boundary

Geometric Entity Selection – **Selection** = Manual, 1

Variables – **Name, Expression** = S, s

Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 2:

Geometric Entity Selection – **Geometric entity level** = Boundary

Geometric Entity Selection – **Selection** = Manual, 2

Variables – **Name, Expression** = S, s+1

Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 3:

Geometric Entity Selection – **Geometric entity level** = Boundary

Geometric Entity Selection – **Selection** = Manual, 4

Variables – **Name, Expression** = S, s+2

Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 4:

Geometric Entity Selection – **Geometric entity level** = Boundary

Geometric Entity Selection – **Selection** = Manual, 3

Variables – **Name, Expression** = S, s+3

Щоб змінна моделі стала доступною, потрібно виконати переобчислення моделі, наприклад, командою

Study 1: RMB > Update Solution

Щоб перевірити працездатність змінної *S*, використаємо її для задання граничних умов. Спочатку деактивуємо попередній вузол граничних умов.

Dirichlet Boundary Condition 1: RMB > Disable

Потім задамо нові граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

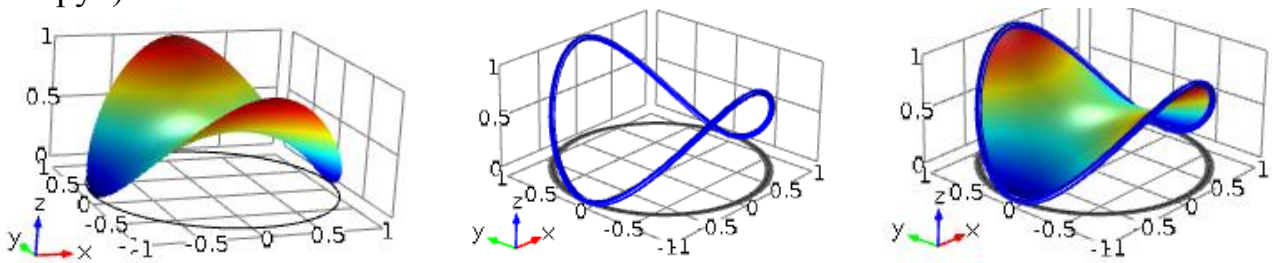
Boundary Selection – **Selection** = All boundaries

Dirichlet Boundary Condition – **r** = (1+cos(S*pi))/2

Виконайте обчислення.

Study 1 : LMB > Compute

У вузлі **2D Plot Group 1** графік розв'язку зміниться (наступний рисунок ліворуч).



Для наочності побудуємо графік функції граничного значення над контуром області. Для цього виконайте інструкції.

2D Plot Group 2: RMB > Duplicate

2D Plot Group 3 – Line 1:

Expression – **Expression** = (1+cos(S*pi))/2

2D Plot Group 3: LMB > Plot (*попередній рисунок посередині*)

Накладемо обидва графіки один на одного. Для цього скопіюємо вузол **Surface 1** з гілки **2D Plot Group 1** в гілку **2D Plot Group 3**.

2D Plot Group 1 – Surface 1: RMB > Copy

2D Plot Group 3: RMB > Past Surface (*попередній рисунок праворуч*)

Останній рисунок моделює дротяний каркас (гранична крива) і поверхню мильної плівки (мінімальна поверхня), яка на нього натягнута.

Створювати змінну моделі, яка задана різними виразами на різних ділянках межі двовимірної області, клопітно. Мабуть простіше (не набагато) створити кілька вузлів типу Dirichlet Boundary Condition для кожної ділянки границі, на яких використати свій параметр *s* для задання граничних умов. Продемонструємо це на прикладі інших граничних умов, які важко задати, використовуючи просторові координати x, y.

Деактивуємо попередній вузол граничних умов.

Dirichlet Boundary Condition 2 : RMB > Disable

Потім задамо нові граничні умови, створивши вузли типу `Dirichlet Boundary Condition` для кожної дуги граничного кола.

Poisson's Equation (poeq): RMB > Dirichlet Boundary Condition

Dirichlet Boundary Condition 3:

`Boundary Selection` – **Selection** = `Manual,1`

`Dirichlet Boundary Condition` – **r** = `2*s`

Poisson's Equation (poeq): RMB > Dirichlet Boundary Condition

Dirichlet Boundary Condition 4:

`Boundary Selection` – **Selection** = `Manual,2`

`Dirichlet Boundary Condition` – **r** = `2-s`

Poisson's Equation (poeq): RMB > Dirichlet Boundary Condition

Dirichlet Boundary Condition 5:

`Boundary Selection` – **Selection** = `Manual,4`

`Dirichlet Boundary Condition` – **r** = `1`

Poisson's Equation (poeq): RMB > Dirichlet Boundary Condition

Dirichlet Boundary Condition 6:

`Boundary Selection` – **Selection** = `Manual,3`

`Dirichlet Boundary Condition` – **r** = `1-s`

Study 1 : LMB > Compute

У вузлі **2D Plot Group 1** графік розв'язку зміниться, що приведе також до розузгодження графіків в гілці **2D Plot Group 3**.

Побудуємо графік «дротяного каркасу і натягнутої на нього мильної плівки» при нових граничних умовах. Щоб намалювати шматки «дротяного каркасу» нам будуть потрібні джерела даних типу `Edge 2D`, що «прив'язані» до дуг кола.

Results – Data Sets: RMB > Edge 2D

Data Sets – Edge 2D 2:

`Selection` – **Selection** =`Manual,1`

Data Sets: RMB > Edge 2D

Data Sets – Edge 2D 3:

`Selection` – **Selection** =`Manual,2`

Data Sets: RMB > Edge 2D

Data Sets – Edge 2D 4:

`Selection` – **Selection** =`Manual,4`

Data Sets: RMB > Edge 2D

Data Sets – Edge 2D 5:

`Selection` – **Selection** =`Manual,3`

Тепер дублюємо гілку **2D Plot Group 3** і в новостворену гілку **2D Plot Group 4** внесемо зміни.

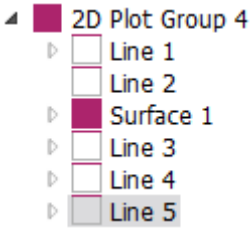
2D Plot Group 3: RMB > Duplicate

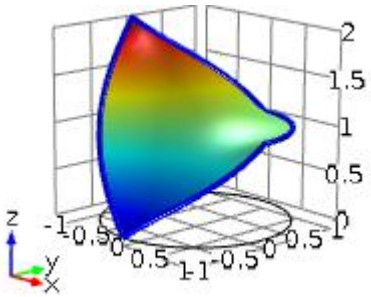
2D Plot Group 4 – Line 1:

`Data` – **Data set** = `Edge 2D 2`

`Expression` – **Expression** = `2*s`

2D Plot Group 4 – Line 1: RMB > Duplicate
2D Plot Group 4 – Line 3:
Data – **Data set** = Edge 2D 3
Expression – **Expression** = 2-s
2D Plot Group 4 – Line 3: RMB > Duplicate
2D Plot Group 4 – Line 4:
Data – **Data set** = Edge 2D 4
Expression – **Expression** = 1
2D Plot Group 4 – Line 4: RMB > Duplicate
2D Plot Group 4 – Line 5:
Data – **Data set** = Edge 2D 5
Expression – **Expression** = 1-s
2D Plot Group 4: LMB > Plot





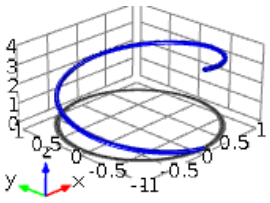
Збережіть файл.

Таким чином, в двовимірних задачах задавати граничні умови можна:

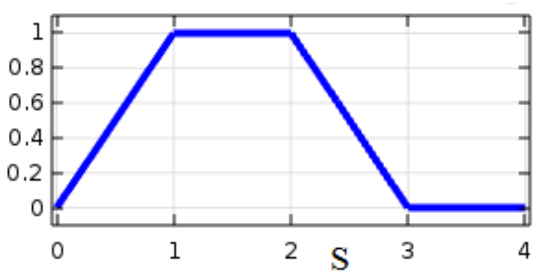
- використовуючи просторові координати x, y;
- параметр s ділянок границі;
- спеціально створену єдину змінну – параметр вздовж межі області.

Зауваження. Окрім змінної s існують інші величини, що пов'язані з геометрією межі двовимірної розрахункової області. Так на граничних кривих завжди визначені одиничні дотичний $\boldsymbol{t} = (t_x, t_y)$ і нормальний $\boldsymbol{n} = (n_x, n_y)$ вектори. В математичних виразах, що «працюють» на межі області, можна використовувати символи `tx`, `ty`, `nx`, `ny`, які представляють координати цих векторів. Існують і інші величини, що пов'язані з геометрією межі моделі, і які можна використовувати в виразах на їх границях.

Вправа 1. Побудуйте графік функції $f(S) = S$ над контуром кола, де S змінна, створена в прикладі.



Вправа 2. Використовуючи створену в прикладі змінну `S`, задайте граничну умову Діріхле $u|_{\partial\Omega} = f(S)$ для функції $f(S)$, яка має наступний вигляд.



Розв’яжіть задачу, побудуйте графік «дротяного каркасу і натягнутої на нього мильної плівки» при цій граничній умові.

Вказівка. Запишіть рівняння функції $f(S)$ у вигляді

$$f(S)=\frac{1}{2}\left(S-|S-1|-|S-2|+|S-3|\right).$$

3.9. Безвихрова течія ідеальної рідини.

Ідеальною рідиною в гідродинаміці називають уявну рідину, у якої відсутня в'язкість і стисливість [8]. Моделлю ідеальної рідини користуються при теоретичному розгляді задач, в яких в'язкість не є визначальним фактором і нею можна нехтувати.

Математичним записом умови нестисливості є рівняння $\operatorname{div}\boldsymbol{V}=0$, де $\boldsymbol{V}(\boldsymbol{x},t)$ – векторне поле швидкостей. У разі відсутності обертання воно задовольняє умові $\operatorname{rot}\boldsymbol{V}=0$, що є необхідною і достатньою умовою потенціальності поля $\boldsymbol{V}$ (в однозв’язних областях). Це означає, що існує скалярна функція $\varphi(\boldsymbol{x},t)$ – потенціал швидкостей, така, що $\boldsymbol{V}=\operatorname{grad}\varphi$. Використовуючи останнє співвідношення в умові нестисливості, бачимо, що φ по просторовим координатам задовольняє рівнянню Лапласа

$$\Delta\varphi=\operatorname{div}\operatorname{grad}\varphi=\frac{\partial^2\varphi}{\partial x^2}+\frac{\partial^2\varphi}{\partial y^2}+\frac{\partial^2\varphi}{\partial z^2}=0.$$

При розгляданні двовимірного усталеного руху $\boldsymbol{V}=(V_x,V_y)$ умови нестисливості і відсутності обертання перетворюються на систему

$$\frac{\partial V_x}{\partial x}+\frac{\partial V_y}{\partial y}=0,\ \frac{\partial V_x}{\partial y}-\frac{\partial V_y}{\partial x}=0. \qquad (1)$$

Потенціал швидкостей φ буде гармонічною функцією двох змінних, тобто буде задовольняти рівнянню Лапласа

$$\Delta\varphi=\frac{\partial^2\varphi}{\partial x^2}+\frac{\partial^2\varphi}{\partial y^2}=0, \qquad (2)$$

а компоненти вектора швидкості $\boldsymbol{V}=(V_x,V_y)$ будуть обчислюватися за формулами

$$V_x=\frac{\partial\varphi}{\partial x},\ V_y=\frac{\partial\varphi}{\partial y}. \qquad (3)$$

З іншого боку, перше рівняння (1) показує, що вираз $-V_y dx + V_x dy$ є диференціалом деякої функції $\psi(x, y)$, такої що

$$-V_y = \frac{\partial \psi}{\partial x}, V_x = \frac{\partial \psi}{\partial y}. \quad (4)$$

Дотична до лінії $\psi = const$ спрямована вздовж вектора $(\psi'_y, -\psi'_x)$ і співпадає з вектором швидкості $\boldsymbol{V} = (V_x, V_y)$. Тому лінії рівня $\psi = const$ є векторними лініями поля швидкостей, які при усталеному русі співпадають з траєкторіями рухомих частинок, тобто з лініями течії, і тому ψ називають функцією течії або функцією струменя. Із другого рівняння (1) випливає, що ψ теж є гармонічною функцією, а порівняння (3) і (4) дає

$$\frac{\partial \varphi}{\partial x} = \frac{\partial \psi}{\partial y}, \frac{\partial \varphi}{\partial y} = -\frac{\partial \psi}{\partial x}.$$

Таким чином, при моделюванні плоского усталеного руху нев'язкої нестисливої рідини можна шукати гармонічну функцію ψ (тобто розв'язувати рівняння Лапласа $\Delta\psi = 0$), а потім визначати поле швидкостей за формулами (4). Інший спосіб полягає в пошуку гармонічної функції потенціалу φ, а поле швидкостей визначати за формулами (3).

При обтіканні тіла ідеальною рідиною вважається, що рідина не проникає в тіло і не відходить від нього, тобто на границі Г нормальна до межі тіла складова швидкості V_n дорівнює нулю $V_n\big|_\Gamma = 0$. Звідси випливає, що границі є лініями струменя, оскільки на них швидкість течії спрямована вздовж їх дотичної. Числові значення ψ, які відповідають граничним лініям течії, повинні бути визначені до початку розв'язання задачі. Лінія симетрії, яка паралельна напрямку течії, теж буде лінією течії.

Якщо задача розв'язується за допомогою потенціалу швидкостей φ, то нормальна до границі компонента швидкості V_n має вигляд

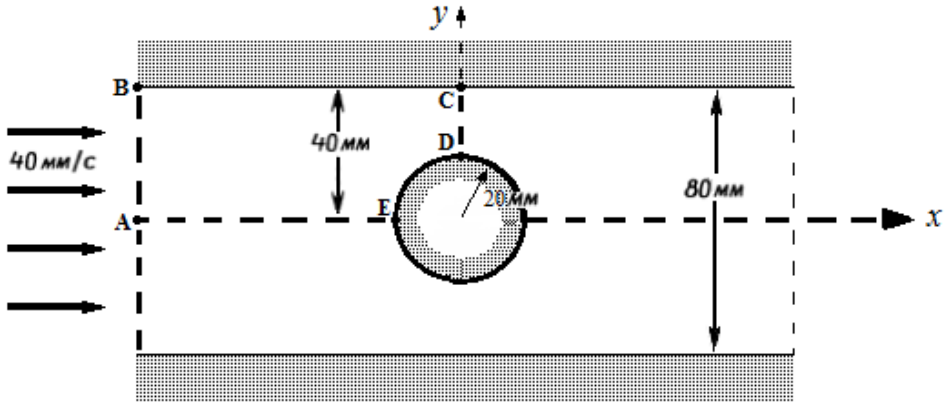
$$V_n = V_x n_x + V_y n_y = \varphi_x n_x + \varphi_y n_y = \frac{\partial \varphi}{\partial n},$$

і гранична умова $V_n = 0$ набуває вигляду $\partial\varphi/\partial n = 0$, де $\boldsymbol{n} = (n_x, n_y)$ одиничний вектор нормалі до границі.

При розв'язанні рівняння $\Delta\varphi = 0$ з граничними умовами на похідні (типу $\partial\varphi/\partial n = 0$) виникає додаткова складність, пов'язана з тим, що розв'язок рівняння Лапласа не зміниться, наприклад, при додаванні константи до функції φ. Отже розв'язок може бути не єдиним. Ця складність долається вибором одного вузла і задаванням в ньому довільного значення φ. Оскільки поле швидкостей $\boldsymbol{V} = (V_x, V_y)$ визначається диференціюванням функції φ, то значення компонент V_x, V_y від цього не міняються.

Приклад. Розглянемо двовимірну задачу усталеного обтікання ідеальною рідиною циліндра в каналі скінченної товщини, використовуючи функцію течії [5]. Діаметр циліндра 40 мм. Він розташований посередині між двома

паралельними стінками, відстань між якими дорівнює 80 мм. На достатній відстані від циліндра однорідний потік рідини має швидкість 40 мм/с.



Обтікання циліндричного тіла

Для розв'язання задачі використаємо `Classical PDEs – Laplace Equation (lpeq)` інтерфейс.

1. Розв'язок з використанням функції течії.

За допомогою майстра *створіть* 2D *модель задачі*

File->New -> Model Wizard -> 2D -> Mathematics - Classical PDEs - Laplace Equation (lpeq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Нехай геометрія задачі складається з прямокутника і кругового отвору в ньому, при чому початок координат розташований в центрі кола. Задача має дві вісі симетрії, які співпадають з осями координат. Тому для заощадження комп'ютерних ресурсів розрахункову область моделі побудуємо як чверть геометрії, а на боках, співпадаючих з осями симетрії, задамо відповідні граничні умови.

`Geometry` -> **`Primitives`** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

`Size` – Width = 60

`Size` – Height =40

`Position` – x, y = -60,0 (*два поля*)

`Geometry` -> **`Primitives`** – Circle

Component 1 - Geometry 1 - Circle 1:

`Size and Shape` – Radius: = 20

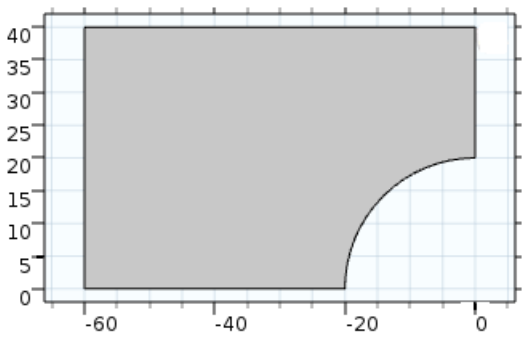
`Geometry` -> **`Booleans and Partitions`** – Difference

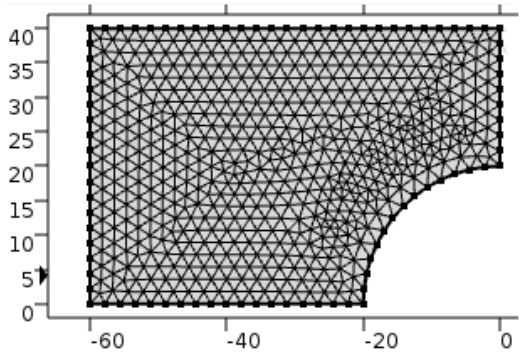
Component 1 - Geometry 1 - Difference 1:

`Difference` – Object to add = r1 (*прямокутник*)

`Union` – Object to subtract = c1 (*коло*)

LMB > Build All Objects (*наступний рисунок ліворуч*)





На створеній геометрії генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** = Finer

LMB> Build All (*попередній рисунок праворуч*)

Перевірте, що в ПВ гілки **Laplace Equation 1 (lpeq)** в розділі **Domain Selection** в секції вибору стоїть 1, тобто обрана побудована область. У вузлі **Laplace Equation 1** наведено загальний вигляд рівняння Лапласа

$$\nabla \cdot (-\nabla u) = 0, \quad \nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right],$$

де ідентифікатор u буде позначати шукану в задачі функцію течії. Ніякі коефіцієнти рівняння задавати непотрібно.

Задайте граничні умови. На лівому боці області однорідний потік рідини має швидкість 40 мм/с. Якщо на ньому (на схематичному рисунку попереду позначено як відрізок AB) задати умову Дірихле $u\big|_{AB} = 40\,y$, то з (4) випливатиме, що на цій лінії буде $V_x\big|_{AB} = u'_y\big|_{AB} = 40$.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** , <> = Manual,1 (відрізок AB)

Dirichlet Boundary Condition – **r** = 40*y

Верхня межа BC та вісь симетрії AE будуть лініями течії. На них буде обертатися на нуль нормальна компонента швидкості $V_y = 0$, тобто $V_y\big|_{BC} = -u'_x\big|_{BC} = 0$ і $V_y\big|_{AE} = -u'_x\big|_{AE} = 0$. Для задавання такої граничної умови (це не умова Діріхле і не умова Неймана) можна використати вузол типу Constraint. В ньому задається вираз, який повинен обертатися на нуль (R=0). Нагадаємо, що для роботи з такими вузлами в налаштуваннях майстра моделей повинна бути увімкнена опція **Advanced Physics Options** в списку, який відкривається кнопкою Show вгорі панелі майстра.

Перевірте, що опція **Advanced Physics Options** увімкнена і виконайте наступні інструкції

Physics -> **Boundaries** – Constraint

Constraint 1:

Boundary Selection – **Selection,**<>=Manual,2,3 (відрізки BC і AE)

Constraint – **R** = ux

Цим задається гранична умова $u'_x = 0$ на обох відрізках.

На дузі ED кола нормальна складова швидкості

$$V_n = \text{Проекція } \boldsymbol{V} \text{ на } \boldsymbol{n} = V_x\, n_x + V_y\, n_y$$

теж повинна обертатися на нуль, де $\boldsymbol{n} = (n_x, n_y)$ – вектор нормалі до граничної кривої. Раніше ми казали, що в COMSOL існують геометричні змінні, які визначені на граничних кривих двовимірних областей, наприклад, одиничні дотичний $\boldsymbol{t} = (t_x, t_y)$ і нормальний $\boldsymbol{n} = (n_x, n_y)$ вектори, для позначення яких застосовуються символи `tx`, `ty`, `nx`, `ny`. Використовуючи їх у вузлі типу `Constraint`, можна задати потрібну граничну умову.

`Physics` -> **`Boundaries`** – Constraint

Constraint 2:

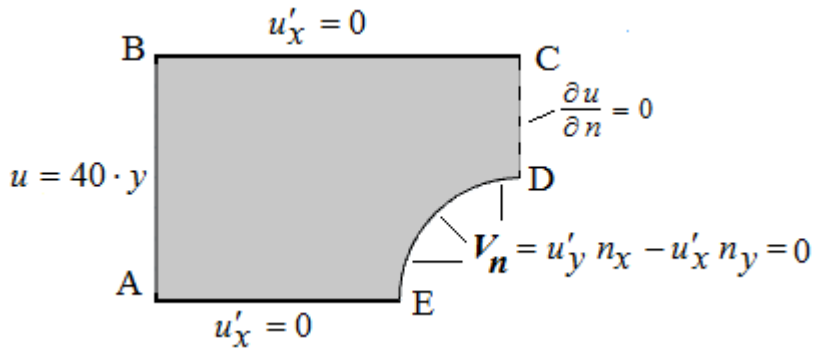
`Boundary Selection` – **Selection,** <> = `Manual,5` (дуга ED)

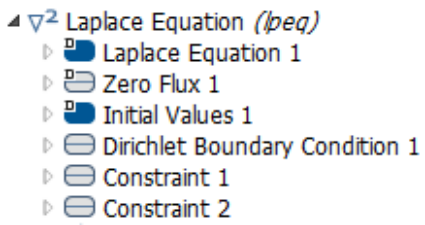
`Constraint` – **R** = `uy*nx-ux*ny`

Цим на дузі ED задається гранична умова $u'_y\, n_x - u'_x\, n_y = V_x\, n_x + V_y\, n_y = V_n = 0$.

Умова симетрії на прямолінійних відрізках, наприклад на відрізку CD, зазвичай полягає в рівності нулю нормальної складової розв'язку, тобто $\dfrac{\partial u}{\partial n} = 0$. В нашому прикладі воно означатиме, що $u'_x = 0$, тобто вертикальна складова вектора швидкості $V_y = -u'_x$ дорівнюватиме нулю. Умову $u'_n\big|_{CD} = 0$ задавати непотрібно, бо вона співпадає з умовою, яка за замовчуванням задана у вузлі **Zero Flux 1**.

На наступному рисунку наведено використану в прикладі схему обмежень на боках розрахункової області, та вигляд дерева моделей з вузлами граничних умов.





Виконайте обчислення.

Study 1 : LMB > Compute

Після завершення обчислень в графічній області вузла **Results - 2D Plot Group 1 – Surface 1** буде побудована розрахункова область з функціональним забарвленням (наступний рисунок ліворуч). Шкала кольорів поруч показує діапазон зміни значень шуканої функції течії *u*. Але розв'язок нашої задачі краще проілюструють лінії течії. Видаліть вузол **Surface 1**, замість нього створіть в поточній графічній групі вузол `Streamline` і налаштуйте його.

2D Plot Group 1 – Surface 1: RMB > Delete

2D Plot Group 1 : RMB > Streamline

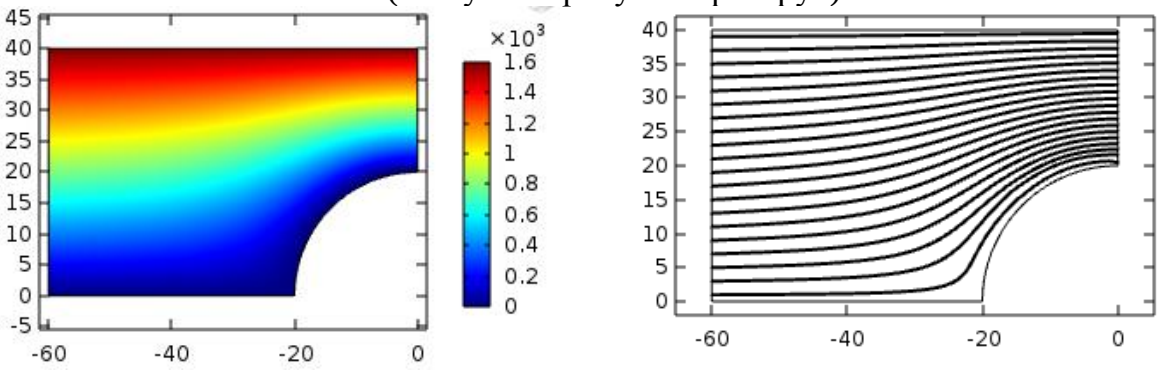
2D Plot Group 1 – Streamline 1:

Expression – x component, y component = uy,-ux (два поля)

Selection – Selection, <> = Manual, 1 (відрізок AB)

При побудові ліній течії (streamline) потрібно вказати через яку ділянку границі вони проходитимуть (ми вказали відрізок AB), а також задати їх кількість (за замовчуванням обрано 20).

LMB > Plot (наступний рисунок праворуч)



Тепер побудуємо лінії струменя на всій зоні течії, а не на її чверті. Для цього нам знадобиться значення функції течії у повній геометрії (в прямокутнику з круговим отвором). Це робиться за допомогою джерела даних Mirror 2D. Воно призначено для продовження розв'язку або виразу з деякої зони на область, яка отримується об'єднанням розрахункової області та її шуканих функцій з віддзеркаленими відносно вісі симетрії розрахунковою областю і функціями.

Results – Data Sets: RMB > More Data Sets - Mirror 2D

Data Sets – Mirror 2D 1:

Data – Data set = Study 1/Solution 1

Axis Data – Point 1 – x, y = 0,0

Axis Data – Point 2 – x, y = 0,40

Цим задається вісь симетрії, що проходить крізь точки (0,0) і (0, 40), тобто відрізок CD.

LMB > Plot (наступний рисунок ліворуч)

Продовжимо розв'язок відносно осі *x*, яка є горизонтальною віссю симетрії задачі. Для цього створимо ще одне ДД Mirror 2D, а в якості його базового джерела даних оберемо тільки що створене.

Results – Data Sets: RMB > More Data Sets - Mirror 2D

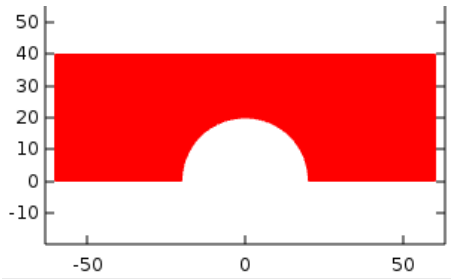
Data Sets – Mirror 2D 2:

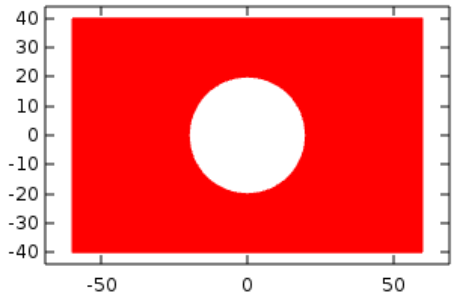
Data – Data set = Mirror 2D 1

Axis Data – Point 1 – x, y = -60,0 (точка на осі симетрії)

Axis Data – Point 2 – x, y = 60,0 (точка на осі симетрії)

LMB > Plot (наступний рисунок праворуч)





Зараз джерело даних **Mirror 2D 2** можна використовувати для побудови двовимірних графіків розв'язку. Майте на увазі, що джерела даних типу `Mirror` призначені для продовження шуканих функцій, а векторні поля продовжуються погано. Тому при подальшому використанні ДД **Mirror 2D 2** ми будемо будувати контурний графік функції *u*, тобто лінії сталого значення функції течії, а не її векторні лінії (хоча в нашій задачі це одне саме).

2D Plot Group 1 : RMB >Duplicate

2D Plot Group 2 :

`Data` – **Data set** = `Mirror 2D 2`

2D Plot Group 2 - Streamline 1: RMB > Disable

2D Plot Group 2 : RMB > Contour

2D Plot Group 2 - Contour 1 :

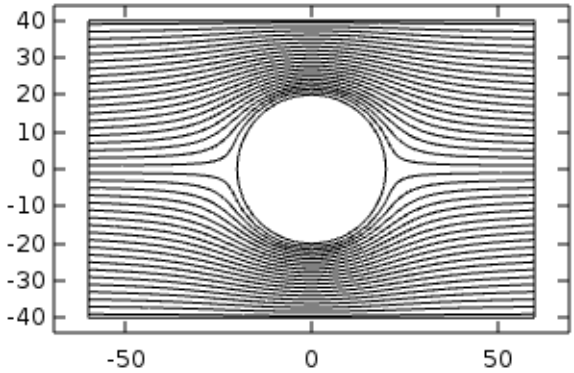
`Expression` – **Expression** = `u`

`Coloring and style` – **Coloring, Color** = `Uniform,Black`

`Coloring and style` – **Color legend** = `False`

LMB > Plot

Зображення в графічній зоні набуде наступного вигляду.



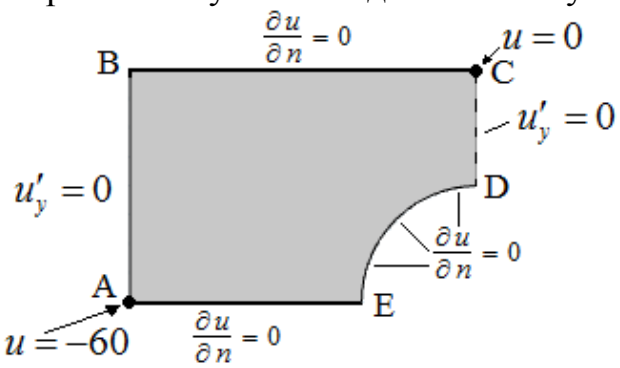
Збережіть файл з ім'ям, наприклад, `ideal_liquid_stream_1.mph`, але не закривайте його.

Зауваження. Той же самий розв'язок ми отримаємо, якщо задамо граничні умови трохи по-іншому. На відрізках AB і CD граничні умови не змінюємо. На відрізках BC, AE і дузі DE замість визначення нульової нормальної складової швидкості визначимо: для межі BC значення функції течії `u=1600`, для межі, що складається з відрізка AE і дуги ED (вона є лінією течії), призначимо нульове значення `u=0`. Це потребує створення вузлів `Dirichlet Boundary Condition`.

Вправа. Збережіть файл з іншим ім'ям. В нову копію файла самостійно внесіть зазначені зміни і виконайте обчислення. Ви повинні отримати той же розв'язок.

2. Розв'язок з використанням потенціалу швидкостей.

Збережіть файл з новим ім'ям, наприклад, `ideal_liquid_potential _1.mph`. В цьому файлі шукану величину `u` будемо розглядати як потенціал векторного поля швидкостей. Компоненти вектора швидкостей і потенціал пов'язані співвідношеннями (3). У зв'язку з цим потрібно повністю переглянути граничні умови. Схему задання граничних умов наведено на наступному рисунку.



Граничні умови $V_n = V_x n_x + V_y n_y = u'_x n_x + u'_y n_y = \dfrac{\partial u}{\partial n} = 0$ на ділянках BC і AED гарантують, що вони є лініями течії. Для їх реалізації достатньо існуючого вузла **Zero Flux 1**. Обмеження $u'_y = 0$ (`Constraint`) на ділянках AB і CD – це умова того, що вектор швидкості $\boldsymbol{V}_n = (u'_x, u'_y) = (u'_x, 0)$ на вході і виході зони буде мати лише горизонтальну складову. Але задання лише похідних на межі області робить задачу неоднозначною. Додавання значення шуканої величини $u = u_0$ в одній точці гарантує однозначність розв'язку, але це буде стала, і її лінії течії побудувати неможливо. Для коректного розв'язання задачі достатньо задати два різних значення потенціалу u в двох різних точках області, наприклад, в вершинах A і C, які зображені на попередній схемі. Це реалізується за допомогою вузлів `Pointwise Constraint`.

Отже внесемо в поточний файл наступні зміни. Видалимо вузли **Dirichlet Boundary Condition 1** і **Constraint 2**.

Dirichlet Boundary Condition 1 : RMB > Delete

Constraint 2 : RMB > Delete

У вузлі **Constraint 1** в секції вибору видалимо попередні індикатори, вставимо нові, і змінимо вираз, який повинен обертатися на нуль.

Constraint 1:

`Boundary Selection` – **Selection,** <> = `Manual,1,4` (відрізки AB і CD)

`Constraint` – **R** = `uy`

Цим задається гранична умова $u'_y = 0$ на лівому і правому боці розрахункової області. На межах BC, AE і DE умова $\dfrac{\partial u}{\partial n} = 0$ задана автоматично у вузлі **Zero Flux 1**.

Створимо і налаштуємо два точкових обмеження.

`Physics` -> **`Points`** – Pointwise Constraint

Pointwise Constraint 1:

Point Selection – **Selection,** <> = Manual,1 (точка A)

Constraint – **R** = 60+u (це означає, що u=−60)

Physics -> **Points** – Pointwise Constraint

Pointwise Constraint 2:

Point Selection – **Selection,** <> = Manual,5 (точка C)

Constraint – **R** = u (це означає, що u=0)

Виконайте обчислення.

Study 1 : LMB > Compute

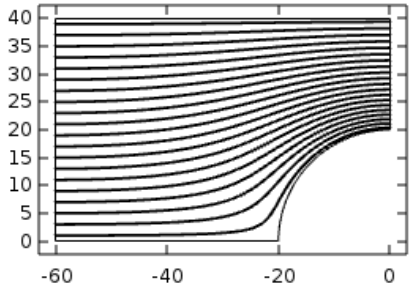
Після завершення обчислень в графічному вікні лінії течії зникнуть, бо в графічних групах залишилися попередні налаштування і їх треба змінити. Так вузол **2D Plot Group 1 – Streamline 1** тепер повинен зображувати лінії течії векторного поля $\boldsymbol{V}_n = \left(u'_x, u'_y\right)$.

2D Plot Group 1 – Streamline 1:

Expression – **x component, y component** = ux,uy (два поля)

LMB > Plot

Ми знову отримуємо таку ж саму картину ліній течії.



Оскільки функція u тепер представляє потенціал швидкості, а не функцію струменя, то у вузлі **2D Plot Group 2 - Contour 1** лінії $u = const$ вже не зображають лінії течії (вони зображають ортогональну до них сім'ю ліній). Видаліть вузол **Contour 1** і замість нього створіть вузол Streamline, в якому зобразіть лінії векторного поля $\boldsymbol{V}_n = \left(u'_x, u'_y\right)$.

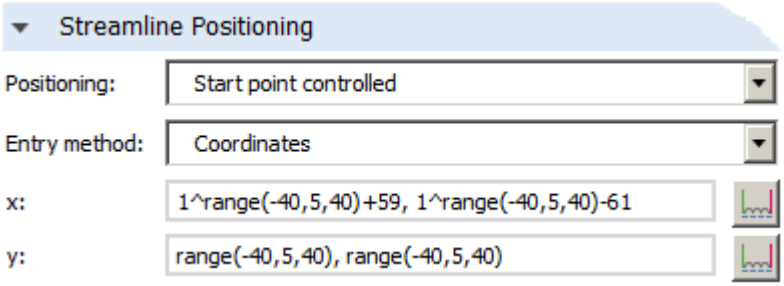
2D Plot Group 2 – Contour 1: RMB > Delete

2D Plot Group 2 - Streamline 1: RMB > Enable

2D Plot Group 2 – Streamline 1:

Expression – **x component, y component** = ux,uy (два поля)

Спочатку рисунок у вас вийде незграбними, бо COMSOL в кожній чверті геометрії побудує окремі системи векторних ліній, які непоєднані одна до одної. Але положенням ліній можна керувати налаштуваннями в розділі **Streamline Positioning**. Підбираючи координати точок, через які повинні проходити векторні лінії, можна побудувати узгоджену систему ліній векторного поля. Налаштуйте розділ **Streamline Positioning** так, як зображено на рисунку.

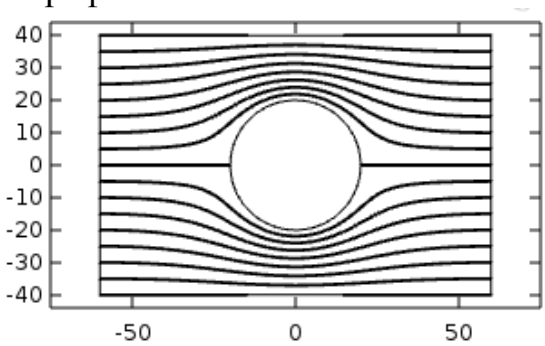


Streamline Positioning – Entry method: = `Coordinates`

Streamline Positioning – x: =
`1^range(-40,5,40)+59, 1^range(-40,5,40)-61`

Streamline Positioning – y: =
`range(-40,5,40), range(-40,5,40)`

Ви отримаєте наступний графік ліній течії.



В кожній чверті рисунка для кожної лінії течії ми задали точку, крізь яку вона повинна проходити, підбираючи координати так, щоб лінії з різних квадрантів з'єднувалися на вертикальній осі. Звичайно, що в полях **x:** та **y:** замість виразів, що використовують функцію `range`, ви можете ввести список координат. Наприклад, замість виразу `1^range(-40,5,40)+59` ви можете ввести послідовність дев'яти чисел `60,60,60,60,60,60,60,60,60`.

3.10. Моделювання точкового джерела.

Розв'яжемо рівняння Пуассона $\nabla\cdot(-\nabla u)=\delta^2$ в одиничному колі Ω з нульовою граничною умовою $u\big|_{\partial\Omega}=0$. Права частина рівняння δ^2 реалізує точкове джерело, яке розташовано в центрі кола. Тут $\delta^2(x,y)=\delta(x)\delta(y)=\dfrac{\delta(r)}{2\pi|r|}$ – двовимірна дельта функція Дірака, яка моделює точковий зовнішній вплив одиничної інтенсивності $\left(\iint_{-\infty}^{\infty}\delta^2(x,y)dxdy=1\right)$.

Точний розв'язок задачі $u=-\dfrac{1}{2\pi}\ln r$, який має особливість в початку координат, може бути порівняний з наближеним, створеним системою.

Можна моделювати точкове джерело у вигляді кола малого діаметра, на якому розподілений великий зовнішній вплив із заданою сумарною інтенсивністю. Тоді в області цього малого кола і деякій його околиці чисельний розв'язок буде дуже відрізнятися від точного. Є інший спосіб, який

враховує, що «за лаштунками» в системі замість розв'язання диференціальних рівнянь мінімізується деякий функціонал. Не вдаючись в тонкощі слабкою постановки задачі, скажемо, що особливість використання точкового джерела в COMSOL Multiphysics полягає в тому, що воно не реалізується як функція правої частини рівняння Пуассона $\nabla\cdot(-\nabla u)=f$, а як окремий елемент `Point Source` – вузол дерева моделі. При цьому функція *f*, що стоїть в правій частині рівняння, ніяк від точкового джерела не залежить. Вузол `Point Source` дає самостійний адитивний внесок в праву частину рівняння і впливає на розв'язок, навіть якщо в правій частині рівняння стоїть нуль. Тому в нашому прикладі можна розв'язувати рівняння Лапласа з точковим джерелом.

Зауважимо, що розв'язок цього прикладу можна знайти в бібліотеці додатків.

Створіть модель послідовністю команд:

File->New->Model Wizard->2D->

Mathematics–Classical PDEs–Laplace Equation (lpeq)->Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Щоб згодом порівняти точний і чисельний розв'язки, створіть функцію, що представляє аналітичний розв'язок.

$$v(x,y)=-\frac{1}{2\pi}\ln r=-\frac{1}{2\pi}\ln\sqrt{x^2+y^2}\ .$$

Для цього виконайте команду

`Definitions->`**`Functions`**`->`Analitic

В ПВ створеного вузла введіть:

Function name = `v`

`Definition` – **Expression** = `-log(sqrt(x^2+y^2))/(2*pi)`

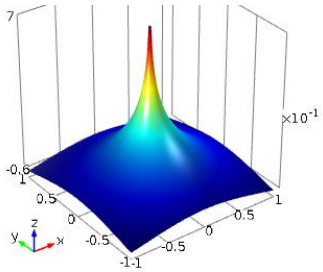
`Definition` – **Arguments** = `x,y`

Для перевірки побудуйте графік цієї функції. В таблиці розділу **`Plot Parameters`** введіть діапазони незалежних змінних

Plot Parameters

Argument	Lower limit	Upper limit
x	-1	1
y	-1	1

і натисніть кнопку Plot.



Геометрія задачі складається з кола одиничного радіуса і точки в його центрі.

`Geometry` -> **`Primitives`** – Circle

Geometry 1 - Circle 1 : LMB > Build Selected (*параметри кола за замовчуванням*)

Geometry-> **Primitives** – Point (*координати точки за замовчуванням*)

Geometry 1 - Point 1 (pt1) : LMB > Build All Objects

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = All boundaries

Dirichlet Boundary Condition – **r** = 0

Створіть і налаштуйте точкове джерело.

Physics->**Points**->Point Source

Point Source 1:

Point Selection – **Selection,** <> = Manual,3 (*точка* (0,0))

Source Term – **f** = 1

На створеному колі *генеруйте* СкЕ *сітку* з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

В околиці точкового джерела варто було б згустити СкЕ сітку. Якби ми моделювали точкове джерело у вигляді кола маленького радіуса, то ми б так і вчинили. Однак в системі можна використовувати алгоритм адаптивного солвера, який робить це автоматично. Adaptive solver враховує конфігурацію області і при необхідності включає здрібнювання СкЕ сітки. Щоб використати його (адаптивний солвер) в стаціонарній задачі в ПВ вузла **Step 1: Stationary** в розділі **Study extensions** встановіть перемикач **Adaptive mesh refinement**.

Study 1 - Step 1: Stationary:

Study extensions – **Adaptive mesh refinement** = True

Виконайте обчислення командою:

Study 1 : LMB > Compute

В гілці **Results - Data Sets** з'являться два вузли, що містять розв'язки: **Study 1/Solution 1** і **Study 1/Adaptive Mesh Refinement 1**. Другий розв'язок отримано з використанням алгоритму адаптивного солвера. Саме його ми будемо аналізувати, скорегувавши налаштування у вузлі **2D Plot Group 1**.

2D Plot Group 1 :

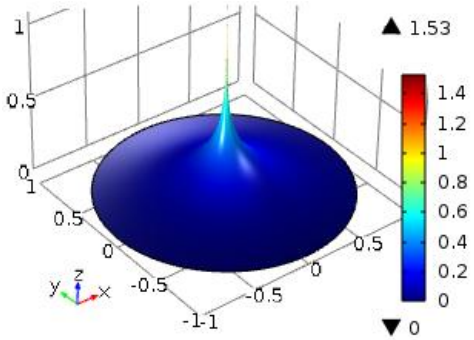
Data – **Data set** = Study 1/Adaptive Mesh Refinement 1

Color Legend - **Show maximum and minimum values** = True

2D Plot Group 1 – Surface 1: RMB > Height Expression

LMB > Plot

В графічному вікні з'явиться зображення поверхні розв'язку зі шкалою кольорів, над якою зазначено максимальне значення (яке, звісно, хибне, оскільки розв'язок в точці (0,0) прямує до нескінченності).



Виконайте порівняння точного розв'язку $v(x,y)$ і розв'язку u, отриманого системою. Для цього побудуйте графік різниці $u-v(x,y)$ розв'язків уздовж відрізка $y=0, 0\le x\le 1$. При цьому слід відсікти невелику околицю початку координат, оскільки аналітичний розв'язок в центрі кола прямує до нескінченності. Тому побудуйте графік на відрізку $0.02\le x\le 1$; $y=0$.

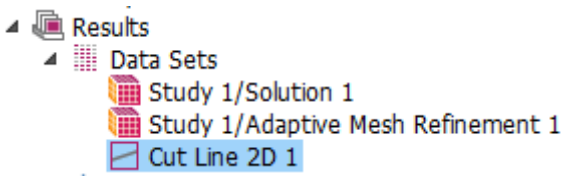
Почніть зі створення джерела даних, що містить точки цього відрізка.

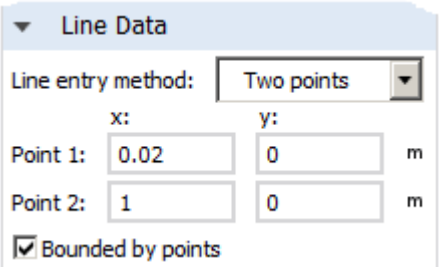
`Results` -> Cut Line 2D

Results – Data Sets – Cut Line 2D 1:

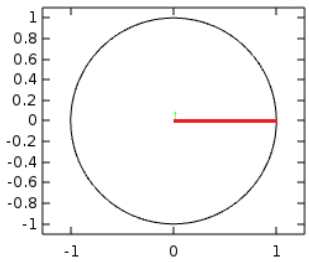
`Data` – **Data set** = `Study 1/Adaptive Mesh Refinement 1`

`Line Data` – **Point 1 – x** = `0.02`





LMB > Plot



Тепер створіть групу одновимірних графіків.

`Results` -> 1D Plot Group

1D Plot Group 2 :

`Data` – **Data set** = `Cut Line 2D 1`

Зверніть увагу, що на головній інструментальній панелі з'явилася закладка `1D Plot Group 2`.

`1D Plot Group 2`-> Line Graph

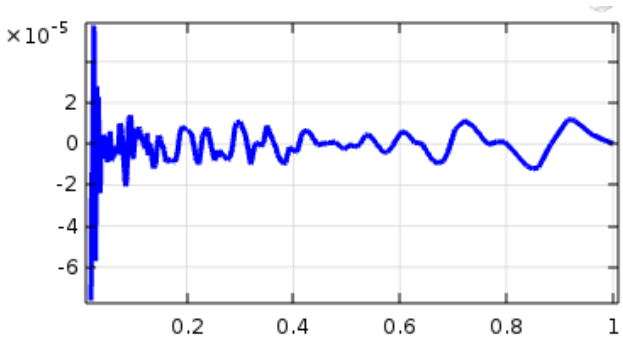
1D Plot Group 2 – Line Graph 1:

`y-Axis Data` – **Expression** = `u-v(x,0)`

`x-Axis Data` – **Parameter** = `Expression`

`x-Axis Data` – **Expression** = `x`

LMB > Plot



Як бачите, різниця точного і наближеного розв'язків на відрізку $0.02 \le x \le 1$ не перевищує $5 \cdot 10^{-5}$, однак зростає при наближенні до лівого кінця. Для точок $0 < x \le 0.02$ розбіжність аналітичного і чисельного розв'язків значно більша. Зауважимо, що якщо при створенні вузла **Cut Line 2D 1** ви оберете розв'язок `Study 1/Solution 1`, який не використовує алгоритм адаптивного солвера, то відмінність точного і наближеного розв'язків буде значно більшою (на відрізку $0.02 \le x \le 1$ воно буде наближатися до значення `0.025`).

Порівняємо СкЕ сітки, які використовувалися в розв'язках `Study 1/Solution 1` і `Study 1/Adaptive Mesh Refinement 1`. Ви можете побачити їх в гілці **Component 1 – Meshes**, де було створено два вузли для СкЕ сіток кожного розв'язку.

Але сітки можна побудувати і в двовимірних графічних групах, створивши в них вузли типу `Mesh`.

`Results` –> 2D Plot Group

2D Plot Group 3 :

`Data` – **Data set** = `Study 1/Solution 1`

2D Plot Group 3 : RMB > Mesh

2D Plot Group 3 - Mesh 1 :

`Color` – **Element color:** = `None`

`Color` – **Wireframe color:** = `Black`

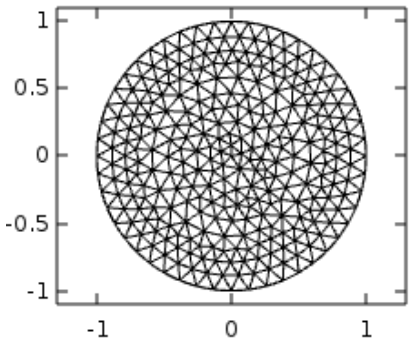
LMB > Plot (*наступний рисунок ліворуч*)

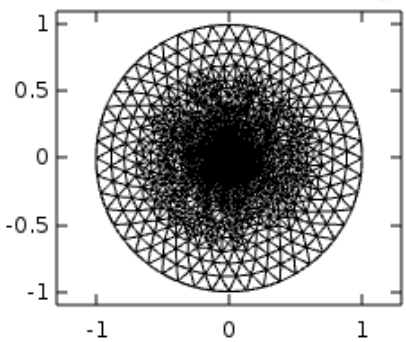
2D Plot Group 3 : RMB > Duplicate

2D Plot Group 4 :

`Data` – **Data set** = `Study 1/Adaptive Mesh Refinement 1`

LMB > Plot (*наступний рисунок праворуч*)





Зауваження. В одновимірних задачах `1D` і `1D Axisymmetric` інструменти `Point Source` відсутні, але їх можна моделювати за допомогою граничної умови типу `Flux/Source`. Дійсно, границі в моделях COMSOL можуть бути внутрішніми, а в одновимірних задачах вони вироджуються в точки. Задання у внутрішній точці граничної умови `Flux/Source` з відповідними параметрами можна інтерпретувати як зосереджені зовнішні зусилля.

3.11. Скін-ефект.

Змінний струм використовується скрізь і задачі, пов'язані з його дослідженням, виникають повсюдно. Розглянемо однорідне середовище з відносною діелектричною проникністю ε і магнітною проникністю μ, позбавлене зарядів. В ізотропних і однорідних середовищах без дисперсії рівняння Максвелла мають наступний вигляд (в системи СІ):

$$\nabla\cdot\boldsymbol{E}=\frac{\rho}{\varepsilon\,\varepsilon_0}\,,\ \nabla\times\boldsymbol{E}=-\mu\,\mu_0\frac{\partial\boldsymbol{H}}{\partial t}\,,\ \nabla\cdot\boldsymbol{B}=0\,,\ \nabla\times\boldsymbol{H}=\boldsymbol{j}+\varepsilon\,\varepsilon_0\frac{\partial\boldsymbol{E}}{\partial t}\,, \qquad (1)$$

де ρ - об'ємна густина електричних зарядів, $\boldsymbol{j}$ – густина електричного струму (жирним шрифтом позначаються векторні величини, курсивом - скалярні). Тут ε_0 позначає діелектричну проникністю вакууму (електрична стала), а μ_0- магнітну проникність (магнітна стала). До того ж існує зв'язок між густиною струму $\boldsymbol{j}$ і напруженістю електричного поля $\boldsymbol{E}$ у вигляді закону Ома $\boldsymbol{j}=\sigma\boldsymbol{E}$, де σ - питома провідність середовища [1].

У випадку відсутності електричних зарядів закон Гауса (перше рівняння (1)) набуває вигляду $\nabla\cdot\boldsymbol{E}=0$. Тоді система (1) може бути спрощена. Застосуємо до другого рівняння (1) оператор ротора, тобто

$$\nabla\times\nabla\times\boldsymbol{E}=-\mu\,\mu_0\frac{\partial\nabla\times\boldsymbol{H}}{\partial t} \qquad (2)$$

і замінимо $\nabla\times\boldsymbol{H}$ правою частиною останнього рівняння (1). Після використання закон Ома, права частина (2) набуде вигляду

$$-\mu\,\mu_0\frac{\partial\nabla\times\boldsymbol{H}}{\partial t}=-\mu\,\mu_0\frac{\partial}{\partial t}\left(\boldsymbol{j}+\varepsilon\,\varepsilon_0\frac{\partial\boldsymbol{E}}{\partial t}\right)=-\mu\,\mu_0\sigma\frac{\partial\boldsymbol{E}}{\partial t}-\mu\,\mu_0\varepsilon\,\varepsilon_0\frac{\partial^2\boldsymbol{E}}{\partial t^2}\,.$$

Використовуючи співвідношення

$$\mathrm{rot}\,\mathrm{rot}\,\boldsymbol{E}=\nabla\times(\nabla\times\boldsymbol{E})=\nabla(\nabla\cdot\boldsymbol{E})-\Delta\boldsymbol{E}$$

і умову відсутності зарядів $\nabla\cdot\boldsymbol{E}=0$, для лівої частини (2) отримуємо $\nabla\times\nabla\times\boldsymbol{E}=-\Delta\boldsymbol{E}$. В результаті (2) набуває вигляду

$$\Delta\boldsymbol{E}-\mu\,\mu_0\sigma\frac{\partial\boldsymbol{E}}{\partial t}-\mu\,\mu_0\varepsilon\,\varepsilon_0\frac{\partial^2\boldsymbol{E}}{\partial t^2}=0\,. \qquad (3)$$

Для гармонічного поля $\boldsymbol{E}=\boldsymbol{E}_c e^{i\omega t}$ і гармонічного змінного струму $\boldsymbol{j}=\boldsymbol{j}_c\,e^{i\omega t}$ рівняння (3) набуває вигляду

$$\Delta\boldsymbol{E}_c+\left(\mu\,\mu_0\varepsilon\,\varepsilon_0\omega^2-i\,\mu\,\mu_0\,\sigma\,\omega\right)\boldsymbol{E}_c=0\,. \qquad (4)$$

У двовимірному випадку, коли струм $\boldsymbol{j}$ і напруженість електричного поля $\boldsymbol{E}$ спрямовані перпендикулярно до площини XY і не залежать від координати Z, маємо $\boldsymbol{j}_c = (0, 0, j_c)$, $\boldsymbol{E}_c = (0, 0, E_c)$. Рівняння (4) набуває вигляду

$$\nabla \cdot \left(\frac{1}{\mu \mu_0} \nabla E_c \right) + \left(\varepsilon \varepsilon_0 \omega^2 - i \sigma \omega \right) E_c = 0. \qquad (5)$$

Це комплексне рівняння Гельмгольца, яке описує розповсюдження плоских гармонічних електромагнітних хвиль в провіднику (якщо $\sigma >> \omega \varepsilon \varepsilon_0$) або недосконалому діелектрику.

З другого рівняння (1) $\nabla \times \boldsymbol{E} = -\mu \mu_0 \dfrac{\partial \boldsymbol{H}}{\partial t}$ (закон індукції Фарадея) і співвідношення $\boldsymbol{H} = \boldsymbol{B} / \mu \mu_0$ (у випадку гармонічного поля) випливає

$$\nabla \times \boldsymbol{E} = -\frac{\partial \boldsymbol{B}}{\partial t}, \text{ де } \frac{\partial \boldsymbol{B}}{\partial t} = \frac{\partial}{\partial t} \boldsymbol{B}_c e^{i\omega t} = i \omega \boldsymbol{B}_c e^{i\omega t}.$$

Звідки $\nabla \times \boldsymbol{E}_c e^{i\omega t} = -i \omega \boldsymbol{B}_c e^{i\omega t}$, або $\boldsymbol{B}_c = \dfrac{i}{\omega} \nabla \times \boldsymbol{E}_c$.

Рівняння (5) зазвичай доповнюють граничними умовами Діріхле, які визначають значення електричного поля E_c на межі середовища, або умовами Неймана, які визначають нормальну похідну E_c на межі.

Після розв'язання рівняння (5) відносно E_c можна обчислити густину струму $\boldsymbol{j}_c = \sigma \boldsymbol{E}_c$, магнітну індукцію $\boldsymbol{B}_c = \dfrac{i}{\omega} \nabla \times \boldsymbol{E}_c$ та напруженість магнітного поля $\boldsymbol{H}_c = \boldsymbol{B}_c / \mu_0 \mu$. Додатково можна обчислити потужність джерел тепла, яке виділяється при проходженні струму.

Приклад. Нехай змінний струм розповсюджується вздовж мідного дроту з круглим поперечним перерізом радіуса 0.1. Електропровідність міді $\sigma = 57 \cdot 10^6$ S/m, магнітна проникність $\mu \mu_0 = 4\pi 10^{-7}$, і діелектрична проникність $\varepsilon \varepsilon_0 = 8.8 \cdot 10^{-12}$. При частоті змінного струму $f = 50$ Hz $(\omega = 2\pi f = 100\pi \text{ рад/сек})$ доданком $\varepsilon \varepsilon_0 \omega^2$ в другому члені рівнянні (5) можна було б нехтувати, але ми цього не робитимемо. Нехай також значення густини струму на поверхні провідника дорівнює 1. Тоді значення E_c на межі провідника дорівнюватиме $E_c = 1/\sigma$.

За допомогою майстра *створіть шаблон моделі*

File->New -> Model Wizard -> 2D -> Mathematics - Classical PDEs - Helmholtz Equation (hzeq) -> Stationary

При побудові моделі нам знадобляться значення частоти *f*, кутової частоти ω, магнітної проникності $\mu \mu_0$, електропровідності σ та діелектричної проникності $\varepsilon \varepsilon_0$. Задайте їх в таблиці параметрів.

Global Definitions : RMB > Parameters

Заповніть таблицю параметрів так, як показано на рисунку.

Parameters

Name	Expression	Value
f	50	50
w	2*pi*f	314.16
mu	4*pi*1E-7	1.2566E-6
sgm	57E6	5.7E7
epsi	8.8E-12	8.8E-12

Тут параметри `mu` та `epsi` позначають добутки $\mu_0\,\mu$ та $\varepsilon_0\,\varepsilon$ відповідно.

Побудуйте розрахункову область, яка складається з кола радіуса 0.1.

`Geometry` -> **`Primitives`** – Circle

Component 1 - Geometry 1 - Circle 1 (c1):

`Size and Shape` – **Radius:** = `0.1`

LMB > Build All Objects

На створеній геометрії *генеруйте* СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – **Element Size** = `Finer`

LMB> Build All

Задайте коефіцієнти *рівняння* Гельмгольца (5).

Helmholtz Equation (hzeq) - Helmholtz Equation 1:

`Diffusion Coefficient` – **c** = `1/mu`

`Source Term` – **f** = `0`

`Absorption Coefficient` – **a** = `i*w*sgm-w^2*epsi`

Цим задається рівняння (5), і величина u буде репрезентувати амплітуду електричного поля E_c.

Задайте граничні умови. На зовнішній межі провідника повинна виконуватися умова Діріхле $E_c = 1/\sigma$.

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

`Boundary Selection`– **Selection**=`1,2,3,4` (*контур кола*)

`Dirichlet Boundary Condition` – **r** = `1/sgm`

Виконайте обчислення.

Study 1 : LMB > Compute

Після завершення обчислень в графічній зоні вузла **2D Plot Group 1** буде побудована розрахункова область з функціональним забарвленням, відповідним амплітуді електричного поля E_c. Але нам цікавіше побачити графік густини струму. Змінимо вираз, графік якого зображується у вузлі, і деякі налаштування.

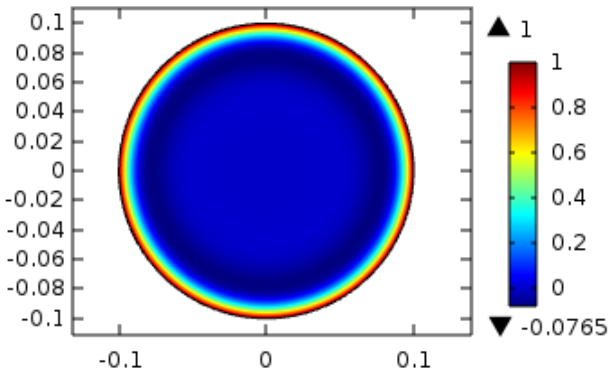
2D Plot Group 1 :

`Color Legend` - **Show maximum and minimum values** = `True`

2D Plot Group 1 – Surface 1:

`Expression` – **Expression** = `real(u)*sgm`

2D Plot Group 1: LMB > Plot



Використовувати функцію `real(...)` було необов'язковим, але ми це зробили, щоб нагадати, що отриманий розв'язок *u* комплекснозначний.

Для наочності побудуємо графік густини струму вздовж горизонтального діаметрального перерізу кола. Почніть зі створення джерела даних, що містить точки цього діаметра.

`Results` –> Data Set - Cut Line 2D

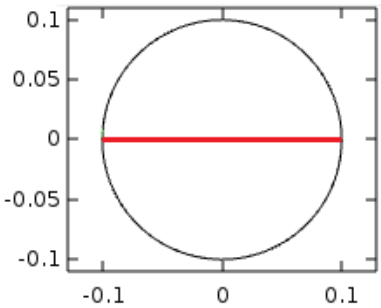
Results – Data Sets – Cut Line 2D 1:

`Data` – **Data set** = `Study 1/Solution 1`

`Line Data` – **Point 1 – x** = `-0.1`

`Line Data` – **Point 2 – x** = `0.1`

LMB > Plot



Тепер створіть групу одновимірних графіків.

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

1D Plot Group 2 :

`Data` – **Data set** = `Cut Line 2D 1`

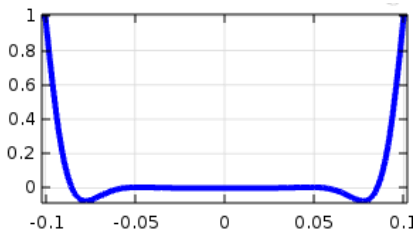
`1D Plot Group 2`–> Line Graph

1D Plot Group 2 – Line Graph 1:

`y-Axis Data` – **Expression** = `real(u)*sgm`

`x-Axis Data` – **Parameter, Expression** = `Expression, x` (*два поля*)

LMB > Plot



На графіках ясно виявляється поверхневий ефект (скін-ефект), притаманний змінному струму. Якщо зменшити частоту струму *f*, то товщина поверхневої

зони, по якій розповсюджується струм, збільшиться. Якщо $f \to 0$, то густина струму стає близькою до рівномірної.

Щоб дослідити це явище, виконаємо параметричний аналіз моделі по частоті *f*. В системі існує кілька способів його зробити. Застосуємо в поточній задачі «інструмент» `Parametric Sweep`. Для цього в гілці **Study** слід створити підвузол **Parametric Sweep**, і в його ПВ вказати параметр та список його значень, які бажано проаналізувати.

Study 1 : RMB > Parametric Sweep

В ПВ створеного вузла **Parametric Sweep** в розділі **`Study Settings`** є таблиця вибору параметрів. Клацніть по ній правою кнопкою миші і з випадного меню виберіть команду Add. Зі списку, який з'явиться в стовпці **Parameter name**, виберіть параметр `f`. У другому стовпці таблиці задайте список значень параметра `range(50,-2,0)`. Ці дії можна записати у вигляді наступних інструкцій:

Study 1 - Parametric Sweep :

`Study Settings` – **Parameter name** = `f`

`Study Settings` – **Parameter value** = `range(50,-2,0)`

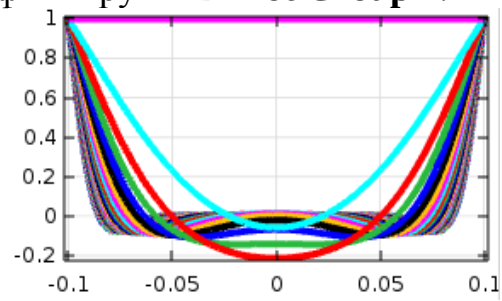
Розділ **`Study Settings`** вузла **Parametric Sweep** набуде наступного вигляду.

Після цього виконайте обчислення моделі.

`Study`-> Compute

В ПВ двовимірної графічної групи **2D Plot Group 1** в розділі **`Data`** з'явиться список **Parameter value (f)**, в якому виберіть значення `50`, щоб отримати графік густини струму для частоти `f=50` Гц. Ви можете обрати будь-яке значення зі списку `50,48,46,...,4,2,0`, який було задано в таблиці.

В одновимірній групі будуть побудовані криві для всіх значень параметра. Ось, як виглядатиме графік в групі **1D Plot Group 2**.



Він репрезентує густину струму вздовж діаметра при всіх значеннях параметра `f` із заданого списку.

В полі **Parameter selection (f)** ви можете обрати не всі, а тільки деякі значення параметра, для яких вам цікаво побудувати графік.

Побудуйте анімацію картини скін-ефекту при зміні частоти `f`.

`Results` –> Export – Animation – Player

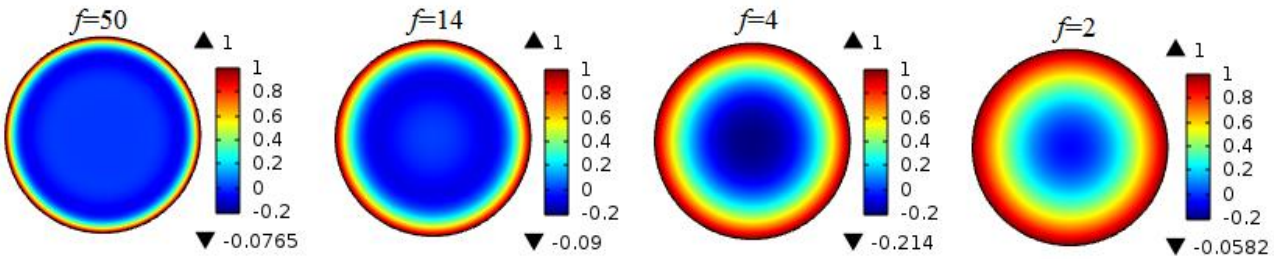
Export - Animation 1:

Scene– Subject = `2D Plot Group 1`

Frames– Frame selection = `All`

LMB> Generate Frame

Запустіть анімацію, клацнувши мишею по кнопці Play , що розташована над графічним вікном. На наступному рисунку наведено кілька кадрів, на яких зображено картину густини струму по перерізу дроту при різних значеннях частоти f $(\omega = 2\pi f)$.



Зауваження. У розглянутої задачі існує аналітичний розв'язок: густина струму j_c в перерізі дроту обчислюється за формулою

$$j_c = j_{bound} \frac{J_0(k\,r)}{J_0(k\,R)}, \; k = \sqrt{i\,\omega\,\sigma\,\mu\,\mu_0}\,, \tag{6}$$

де R – радіус дроту, r – відстань від осі, j_{bound} - граничне значення густини струму, $J_0(x)$ функція Бесселя першого роду нульового порядку.

Перевіримо наш розв'язок $j_c = \sigma E_c$, порівнявши його графік з графіком функції (6). В системі COMSOL Multiphysics функції Бесселя першого роду позначаються, як `besselj(a,x)`, де `a` – порядок функції. Тобто функція $J_0(x)$ позначається як `besselj(0,x)`.

Спочатку в таблицю параметрів додайте комплекснозначну константу *k*, яка буде обчислюватися за формулою (6).

Global Definitions – Parameters :

Parameters – Name, Expression = `k, sqrt(i*w*sgm*mu)`

Щоб новий параметр *k* став доступним, потрібно виконати переобчислення моделі.

Study 1: RMB > Update Solution

Тепер побудуйте графіки.

1D Plot Group 2 : RMB> Duplicate

1D Plot Group 3 :

Data – Parameter selection (f) = `First`

1D Plot Group 3 – Line Graph 1:

Coloring and Style – Line, Width = `Solid, 8` (*два поля*)

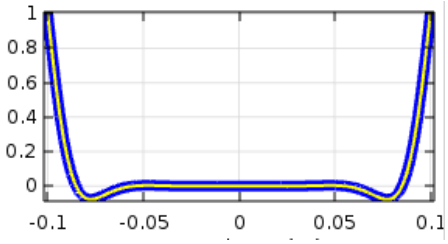
1D Plot Group 3 – Line Graph 1 : RMB> Duplicate

1D Plot Group 3 – Line Graph 2:

y-Axis Data – Expression = `besselj(0,k*x)/besselj(0,k*0.1)`

Coloring and Style – Color, Width = `Yellow, 2` (*два поля*)

LMB > Plot



Тут ми побудували графіки для першого значення `f` зі списку параметрів `range(50,-2,0)`, тобто для значення $f = 50$. Як бачите, графіки співпадають.

3.12. Електростатичний потенціал квадратної рамки.

Окремий розділ електродинаміки являє електростатика. Вона розглядає задачі, в яких заряджені тіла нерухомі [8]. В задачах електростатики розв'язання рівнянь Максвела зводиться до пошуку скалярної функції – потенціалу V, пов'язаного з напруженістю електричного поля $\boldsymbol{E}$ співвідношенням $\boldsymbol{E} = -\nabla V$. Існування потенціалу V випливає з рівняння $\nabla \times \boldsymbol{E} = 0$.

Використання закону Гауса $\nabla \cdot \boldsymbol{E} = \rho / \varepsilon$ призводить до рівняння Пуассона

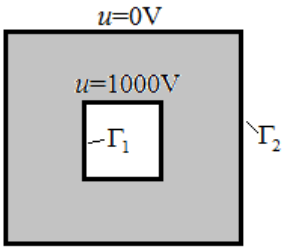
$$-\nabla \cdot (\varepsilon \nabla V) = \rho,$$

де ρ – густина заряду, ε – діелектрична проникність. Зазвичай $\varepsilon = \varepsilon_0 \varepsilon_r$, де ε_0 - діелектрична стала (діелектрична проникність вакууму 8.854 · 10^{-12} Ф·m^{-1}), а ε_r – безрозмірна відносна проникність, яка показує у скільки разів сила взаємодії двох електричних зарядів в певному середовищі менше, ніж у вакуумі.

Граничні умови в задачах електростатики зазвичай бувають двох типів: Діріхле і Неймана. В умовах Дірихле на границі задається значення електростатичного потенціалу V. В умовах Неймана використовується вектор електричної індукції $\boldsymbol{D}$. Він пов'язаний з напруженістю електричного поля $\boldsymbol{E}$ співвідношенням $\boldsymbol{D} = \varepsilon \boldsymbol{E}$. Зокрема на межі Γ двох речовин задається стрибок нормальної компоненти $\boldsymbol{D}_n = \boldsymbol{n} \cdot (-\varepsilon \nabla V)$ вектора $\boldsymbol{D}$, який визначається поверхневою густиною вільних зарядів $\sigma(\boldsymbol{r})$, тобто $\lim\limits_{\varepsilon \to 0} [\boldsymbol{D}_n(\boldsymbol{r} + \varepsilon \boldsymbol{n}) - \boldsymbol{D}_n(\boldsymbol{r} - \varepsilon \boldsymbol{n})] = \sigma(\boldsymbol{r})$, де $\boldsymbol{r} \in \Gamma$ і $\boldsymbol{n}$ - вектор нормалі до Г в точці $\boldsymbol{r}$.

Головною задачею електростатики є визначення поля, створюваного системою зарядів на заданих провідниках.

Приклад. Розглянемо задачу визначення електростатичного потенціалу в квадратній «рамці», заповненій повітрям, з розмірами сторін внутрішнього контура 0.2 м і зовнішнього – 0.5 м. На внутрішній межі електростатичний потенціал дорівнює 1000 V, на зовнішній – 0 V. Електричні заряди відсутні.



Це приводить до рівняння Лапласа $\Delta u = 0$ з граничними умовами Діріхле $u|_{\Gamma_1} = 1000$, $u|_{\Gamma_2} = 0$, де ідентифікатором u позначено шуканий потенціал.

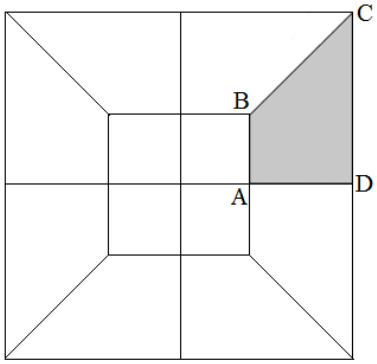
За допомогою майстра *створіть* 2D *модель задачі*

File->New -> Model Wizard -> 2D -> Mathematics - Classical PDEs - Laplace Equation (lpeq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Модель має чотири вісі симетрії, дві з яких співпадають з осями координат, а дві – з діагоналями квадратів. Для заощадження комп'ютерних ресурсів розрахункову область можна побудувати як чверть або навіть як 1/8 частину повної геометрії. Нехай це буде 1/8 частка всієї області.

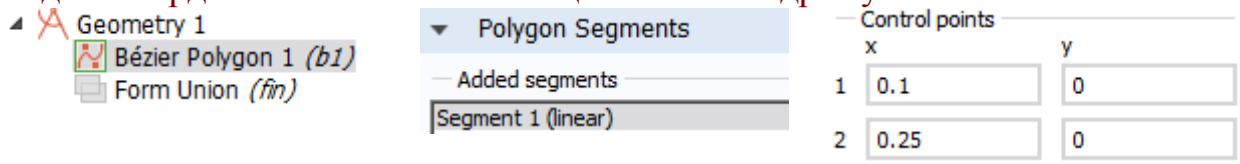


Тоді розрахункова область буде складатися з чотирикутника ABCD, який можна побудувати за допомогою графічного примітива `Bézier Polygon`.

`Geometry` -> **`Pimitives`** – Bézier Polygon

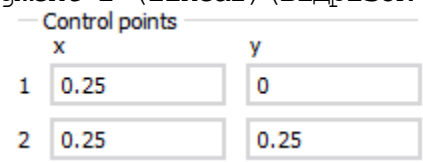
Bézier Polygon 1 : LMB > Add Linear

Введіть координати початкової і кінцевої точок відрізку AD.

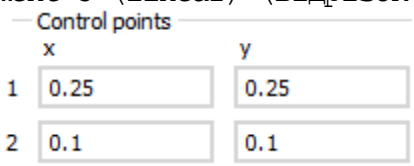


Ще 2 рази клацніть по кнопці Add Linear і кожного разу вводьте координати початкової і кінцевої точок відрізків DC і CB.

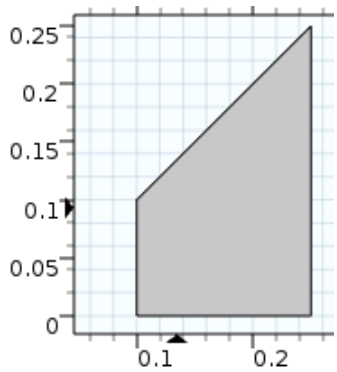
`Segment 2 (linear)(відрізок DC)`



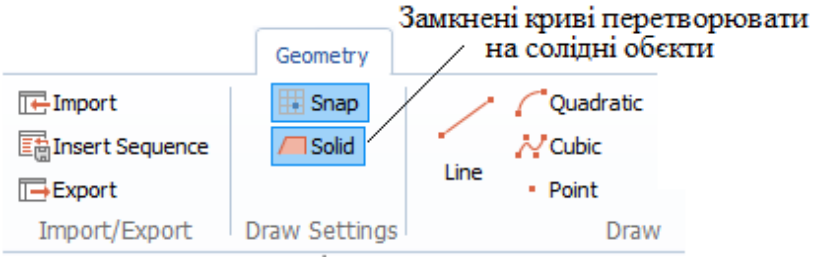
`Segment 3 (linear) (відрізок CB)`



Клацніть по кнопкам Add Linear і Close Curve, замикаючи ламану відрізком BA (її поки що не видно). Завершіть будування чотирикутника клацанням по кнопці Build All Objects.



Зверніть увагу на те, що ми будували ламану, а було отримано двовимірну область. Це тому, що на панелі Geometry в розділі **Draw Settings** увімкнена опція **Solid**.



Генеруйте СкЕ *сітку* зі значеннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Перевірте, що в ПВ гілки **Laplace Equation 1 (lpeq)** в розділі **Domain Selection** в секції вибору стоїть 1, тобто обрана побудована область. У вузлі **Laplace Equation 1** наведено загальний вигляд рівняння Лапласа

$$\nabla\cdot(-\nabla u)=0,\quad \nabla=\left[\frac{\partial}{\partial x},\frac{\partial}{\partial y}\right],$$

де ідентифікатор u позначає електростатичний потенціал. Рівняння не містить коефіцієнтів, тому їх задавати непотрібно.

Задайте граничні умови. На внутрішній межі AB електростатичний потенціал дорівнює 1000V.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** , <> = Manual,1 (*відрізок* AB)

Dirichlet Boundary Condition – **r** = 1000

На зовнішній межі електростатичний потенціал дорівнює 0.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2

Boundary Selection – **Selection** , <> = Manual,4 (*відрізок* CD)

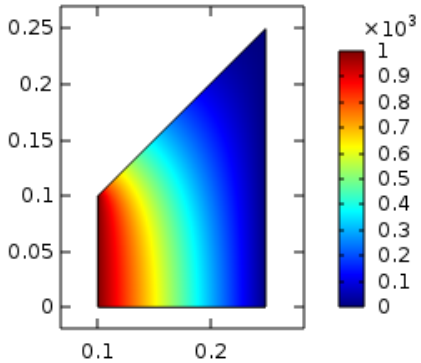
Dirichlet Boundary Condition – **r** = 0

Умова симетрії на прямолінійних відрізках BC і AD полягає в рівності нулю нормальної складової розв'язку, тобто $\frac{\partial u}{\partial n}=0$. Її задавати непотрібно, бо вона співпадає з умовою, яка за замовчуванням задана у вузлі **Zero Flux 1**.

Виконайте обчислення.

Study 1 : LMB > Compute

Після завершення обчислень в графічній зоні вузла **2D Plot Group 1 – Surface 1** буде побудована розрахункова область з функціональним забарвленням. Шкала кольорів поруч показує діапазон зміни значень потенціалу *u*.



Побудуємо графік потенціалу на всій геометрії, продовживши розв'язок *u* через осі симетрії. Спочатку використаємо джерело даних `Mirror 2D`. Воно призначено для продовження розв'язку з деякої зони на область, що отримується об'єднанням цієї зони з власним віддзеркаленням відносно вісі симетрії.

Results – Data Sets: RMB > More Data Sets - Mirror 2D

Data Sets – Mirror 2D 1:

`Data` – **Data set** = `Study 1/Solution 1`

`Axis Data – Point 1` – **x, y** = `0,0`

`Axis Data – Point 2` – **x, y** = `1,1`

Цим задається дзеркальна вісь, що проходить крізь точки (0,0) і (1,1), тобто відносно похилого відрізка BC.

LMB > Plot (наступний рисунок ліворуч)

Послідовним чотириразовим поворотом побудованої зони можна отримати повну геометрію. Для продовження розв'язку, який має подібну симетрію, призначено джерело даних типу `Sector 2D`.

Results – Data Sets: RMB > More Data Sets - Sector 2D

Data Sets – Sector 2D 1:

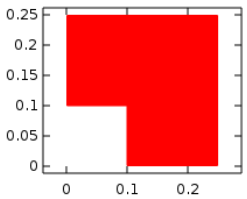
`Data` – **Data set** = `Mirror 2D 1`

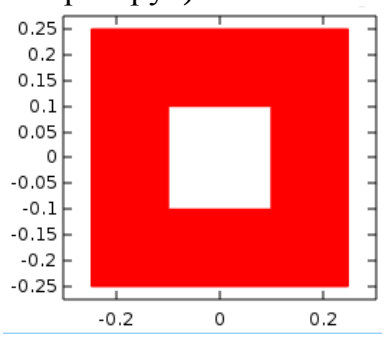
За замовчуванням вісь повороту проходить крізь точку (0,0).

`Symmetry` – **Number of sectors** = `4`

Останнє визначає кількість «секторів симетрії»

LMB > Plot (наступний рисунок праворуч)





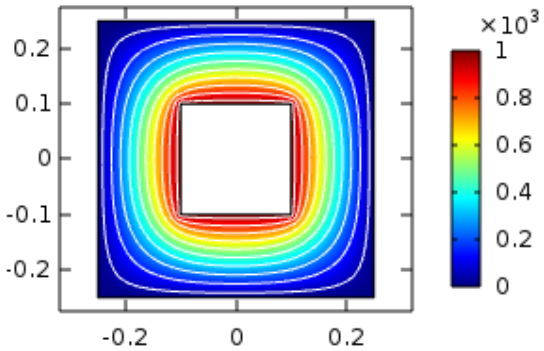
Тепер джерело даних **Sector 2D 1** можна використовувати для побудови двовимірних графіків розв'язку.

2D Plot Group 1 : RMB >Duplicate
2D Plot Group 2 :
Data – **Data set** = Sector 2D 1
Plot Settings – **Plot data set edges** = False
LMB > Plot

Доповнимо рисунок лініями сталого значення потенціалу, додавши в графічну групу вузол Contour.

2D Plot Group 2 : RMB > Contour
2D Plot Group 2 - Contour 1 :
Expression – **Expression** = u (*будувати лінії рівня u=Const*)
Levels – **Entry method** = Number of Levels
Levels – **Levels** = 10
Цим ми визначили кількість ліній рівня.
Coloring and style – **Coloring, Color** = Uniform, White
Coloring and style – **Color legend** = False
LMB > Plot

Зображення в графічній зоні набуде наступного вигляду.



3.13. Магнітне поле електромотора.

Електромотори, трансформатори, електромагніти є тілами, в розрахунках яких використовують закони магнітостатики [8]. Закони Максвела в цьому випадку мають вигляд $\nabla \times \boldsymbol{H} = \boldsymbol{J}$ та $\nabla \cdot \boldsymbol{B} = 0$, де $\boldsymbol{B}$ - магнітна індукція, $\boldsymbol{H}$ – напруженість магнітного поля, $\boldsymbol{J}$ – густина електричного струму. Між векторами $\boldsymbol{B}$ і $\boldsymbol{H}$ є зв'язок $\boldsymbol{B} = \mu \boldsymbol{H}$, де μ – магнітна проникність середовища (Насправді $\boldsymbol{B} = \mu \mu_0 \boldsymbol{H}$, де μ_0 - магнітна проникність вакууму, а μ - відносна магнітна проникність середовища. Але для скорочення запису ми будемо використовувати вказані позначення). Оскільки $\nabla \cdot \boldsymbol{B} = 0$, то існує векторний потенціал $\boldsymbol{A}$ такий, що $\boldsymbol{B} = \nabla \times \boldsymbol{A}$. Тоді

$$\nabla \times \left(\frac{1}{\mu} \nabla \times \boldsymbol{A} \right) = \boldsymbol{J}. \qquad (1)$$

В двовимірному випадку вважають, що струм $\boldsymbol{J}$ та поля $\boldsymbol{B}$, $\boldsymbol{H}$ не залежать від z – координати, і струм $\boldsymbol{J}$ спрямовано вздовж осі Z. З цього випливає, що $\boldsymbol{J} = (0, 0, J(x, y))$, $\boldsymbol{B}$ та $\boldsymbol{H}$ мають складові лише в площині XY, а векторний потенціал $\boldsymbol{A}$ має лише z – компоненту $\boldsymbol{A} = (0, 0, A(x, y))$. Рівняння (1) відносно

«магнітостатичного» потенціалу $A(x,y)$ стає скалярним і набуває вигляду рівняння Пуассона

$$-\nabla\cdot\left(\frac{1}{\mu}\nabla A\right)=J\,. \qquad (2)$$

В цьому випадку магнітна індукція обчислюється за формулою $\boldsymbol{B}=\left(\frac{\partial A}{\partial y},-\frac{\partial A}{\partial x},0\right)$, а напруженість магнітного поля – за формулою $\boldsymbol{H}=\boldsymbol{B}/\mu$.

На межі тіл з різних матеріалів повинна виконуватися умова неперервності величини $\boldsymbol{H}\times\boldsymbol{n}$. В двовимірних задачах маємо

$$\boldsymbol{H}\times\boldsymbol{n}=\frac{1}{\mu}\begin{vmatrix}\boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k}\\ \frac{\partial A}{\partial y} & -\frac{\partial A}{\partial x} & 0\\ n_x & n_y & 0\end{vmatrix}=\boldsymbol{k}\frac{1}{\mu}\left(n_x\frac{\partial A}{\partial x}+n_y\frac{\partial A}{\partial y}\right)=\boldsymbol{k}\frac{1}{\mu}\frac{\partial A}{\partial n}.$$

Отже це призводить до умови неперервності величини $\nabla A/\mu$, яка стоїть в дужках формули (2), і не потребує додаткового розглядання (ця неперервність при слабкому формулюванні ДРЧП виконується автоматично).

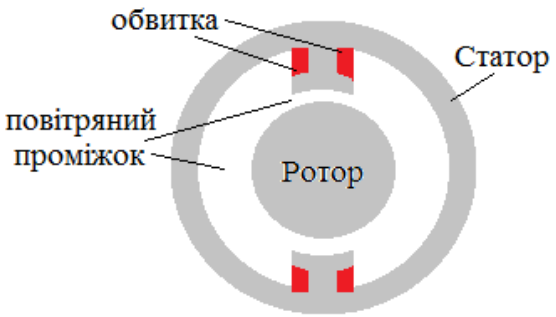
В феромагнітних матеріалах магнітна проникність μ залежить від величини поля $|\boldsymbol{B}|=|\nabla A|$, тобто рівняння (2) є нелінійним.

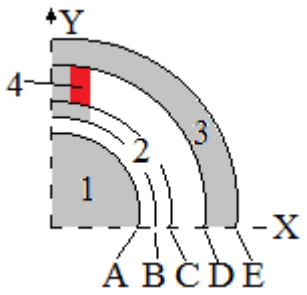
Зазвичай використовуються наступні граничні умови:

- умови Діріхле визначають значення магнітостатичного потенціалу A на межі тіл;
- умови Неймана визначають нормальну компоненту $\boldsymbol{n}\cdot(\nabla A/\mu)$ величини $\nabla A/\mu$ на границі, що еквівалентно завданню тангенційної складової $\boldsymbol{H}_{\boldsymbol{\tau}}=\boldsymbol{\tau}\cdot\boldsymbol{H}=\boldsymbol{n}\cdot(\nabla A/\mu)$ напруженості магнітного поля $\boldsymbol{H}$.

Приклад. Розглянемо задачу визначення статичного магнітного поля при проходженні постійного струму по статорній обвитці електромотора. Двигун розглядається досить довгим в напрямку z, так що кінцевими ефектами можна нехтувати. Тоді можна досліджувати двовимірну модель, схема перерізу якої зображена на наступному рисунку ліворуч.

Магнітна проникність повітря і міді (обвитки) близька до магнітної проникності вакууму $\mu_0=4\pi\cdot10^{-7}$ H/m. Статор і ротор зроблені з феромагнітного матеріалу, магнітна проникність якого визначається за формулою $\mu=\mu_0\left(\mu_{\min}+\frac{\mu_{\max}}{1+c\|\boldsymbol{B}\|^2}\right)$, де $\mu_{\min}=200$, $\mu_{\max}=5000$, $c=0.05$ і $\|\boldsymbol{B}\|^2=\|\nabla A\|^2$. Густина струму дорівнює нулю $J=0$ всюди за виключенням обвитки, де J=10 A/m^2.





Постановка задачі характеризується існуванням у «магнітостатичного» потенціалу A горизонтальної осі симетрії X, та вертикальної осі антисиметрії Y (на лівій частині обвитки струм тече в протилежному напрямку). Отже можна розглядати четвертину геометрії $x \geq 0, y \geq 0$ (попередній рисунок праворуч) з граничною умовою Неймана $\boldsymbol{n} \cdot (\nabla A / \mu) = 0$ на осі X (умова симетрії), та нульовою граничною умовою Діріхле $A|_Y = 0$ на осі Y (умова антисиметрії). Полем ззовні двигуна можна нехтувати, що також призводить до нульових граничних умов Діріхле $A = 0$ на зовнішній межі геометрії (дуга E).

За допомогою майстра створіть 2D модель задачі

File->New -> Model Wizard -> 2D -> Mathematics - Classical PDEs - Poisson's Equation (poeq) -> Stationary

Вимкніть в моделі контроль розмірностей фізичних величин.

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
mumax	5000	5000
mumin	200	200
mu0	4*pi*10^(-7)	1.2566E-6
c	0.05	0.05

Геометрія задачі складається з кількох четвертин концентричних кіл різних радіусів та зони обвитки. Дуга E має радіус 1, дуга D – радіус 0.8, C – 0.6, B – 0.5, A – 0.4. Накресліть їх.

`Geometry` -> **`Primitives`** – Circle

Component 1 - Geometry 1 - Circle 1 (c1):

`Size and Shape` – **Radius:** = `1`

Geometry 1 - Circle 1 (c1) : RMB >Duplicate

Geometry 1 - Circle 2 (c2): `Size and Shape` – **Radius:** = `0.8`

Geometry 1 - Circle 2 (c2) : RMB >Duplicate

Geometry 1 - Circle 3 (c3) : `Size and Shape` – **Radius:** = `0.6`

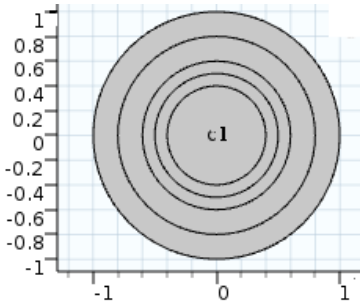
Geometry 1 - Circle 3 (c3) : RMB >Duplicate

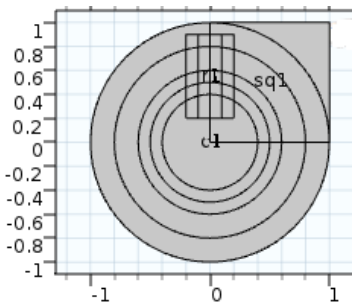
Geometry 1 - Circle 4 (c4) : `Size and Shape` – **Radius:** = `0.5`

Geometry 1 - Circle 4 (c4) : RMB >Duplicate

Geometry 1 - Circle 5 (c5) : `Size and Shape` – **Radius:** = `0.4`

Geometry 1 - Circle 5 (c5) : LMB > Build Selected (*наступний рисунок ліворуч*)





Створіть два прямокутника, які нам знадобляться для зображення обвитки.

`Geometry` **`-> Primitives`** – Rectangle

Component 1 - Geometry 1 - Rectangle1 (r1):

`Size` – **Width, Height** = `0.4, 0.7` (*два поля*)

`Position` – **x, y** = `-0.2,0.2` (*два поля*)

Geometry 1 - Rectangle1 (r1) : RMB >Duplicate

Geometry 1 – Rectangle2 (r2):

`Size` – **Width, Height** = `0.2, 0.7` (*два поля*)

`Position` – **x, y** = `-0.1,0.2` (*два поля*)

Додайте креслення квадрата, який потрібен для виділення четвертини геометрії.

`Geometry` **`-> Primitives`** – Square

Geometry 1 - Square 1 (sq1):

`Size` – **Size length** = `1`

Square 1 (sq1) : LMB > Build Selected (*попередній рисунок праворуч*)

Виконайте операцію логічного компонування створених примітивів.

`Geometry` **`-> Booleans and Partitions`** – Compose

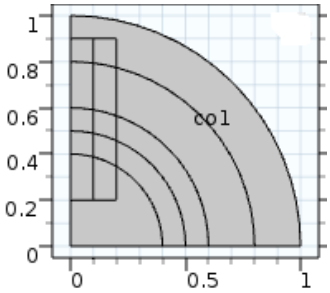
В ПВ створеного вузла **Compose 1 (co1)** в розділі **`Compose`** є текстове поле **Set Formula**, яке призначено для введення формули побудови геометрії за допомогою логічних операцій між областями. Потрібна геометрія відповідає формулі $(c1 \cup c2 \cup c3 \cup c4 \cup c5 \cup r1 \cup r2) \cap sq1$. В COMSOL логічні операції записуються наступним чином: знак '+' (плюс) позначає операцію об'єднання, '*' (зірочка) – перетин, '-' (мінус) – віднімання.

Component 1 - Geometry 1 - Compose 1 (co1):

`Compose` – **Set formula** = `(c1+c2+c3+c4+c5+r1+r2)*sq1`

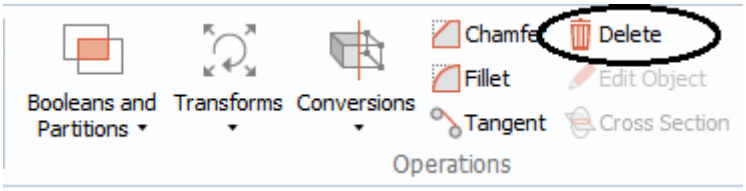
Compose 1 (co1) : LMB > Build Selected

Після виконання команди Build Selected секція **Input objects** заповниться автоматично.



Тепер потрібно видалити внутрішні розділювальні лінії між колами так, щоб всередині утворилися бажані геометричні підобласті. Для цієї операції

призначений «інструмент» Delete, який розташований на панелі Geometry в розділі **Operations**.



Отже виконайте команду

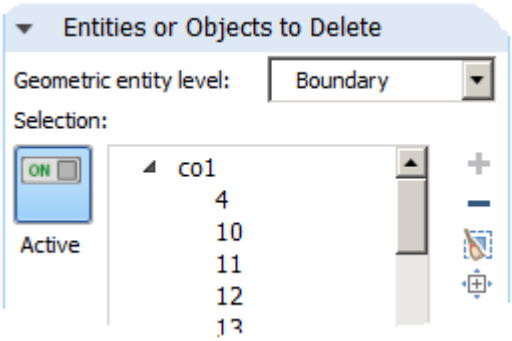
Geometry **->Operations** – Delete

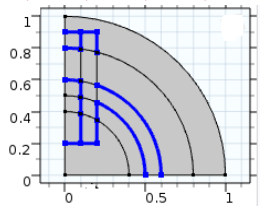
В ДМ з'явиться вузол **Delete Entities 1 (del1)**. В його секції вибору вкажіть ділянки кривих (клацаючи по ним мишею), які будуть видалені. Але спочатку в цьому вузлі ви мусите визначити рівень геометрії об'єктів, які будуть усуватися (точки, лінії або підобласті).

Geometry 1 - Delete Entities 1 (del1):

Entities or Objects to Delete – **Geometric entity level** = Boundary

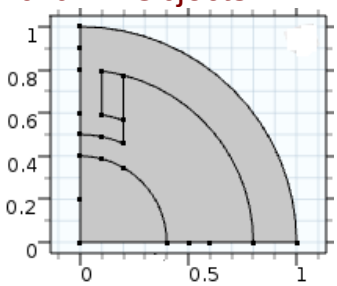
Entities or Objects to Delete – **Selectin :** = 4,10,11,12,13, 14,16,17,18,19,22,29,30,37,38



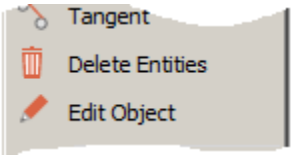


Клацайте по відрізкам, зображеним на рисунку вгорі праворуч, доки не отримаєте геометрію складену з чотирьох підобластей: статора, ротора, обвитки і повітряного проміжку. Якщо ви щось забули видалити, то можна створити ще один вузол Delete Entities і вказати в ньому відрізки, яких бракує.

Delete Entities 1 (del1) : LMB > Build All Objects



Для видалення об'єктів або їх елементів можна також клацнути правою кнопкою миші по гілці **Geometry 1** та з випадного списку вибрати команду Delete Entities .



В рівнянні Пуассона $\nabla\cdot\left(-\nabla A/\mu\right)=J$ нашої моделі використовуються дві змінні μ та J, які мають різні значення в різних підобластях геометрії. Для моделювання подібних задач в Comsol Multiphisics існують два підходи. Можна створити дві різні «фізики» (в нашій задачі використати два рівняння Пуассона), для кожної з яких вказати свою зону дії та власні значення параметрів. Інший підхід, який ми застосуємо в поточному прикладі, полягає в створені змінних, які мають відмінні значення в різних частинах розрахункової області. Для цього в гілці компонента потрібно створити два (чи більше) вузла типу `Variables`, зона дії яких буде поширюватися тільки на деякі частини геометрії. Потім в цих вузлах будуть створені змінні, що мають однакову назву (однакове ім'я) але мають різні вирази для обчислення. Звернення до імені такої змінної, наприклад, при розв'язанні рівняння, буде автоматично використовувати вираз, заданий на відповідній частині геометрії.

Потрібно визначити наступні значення змінних μ та J:

- в обвитці $\mu=\mu_0$, $J=10\ [\mathrm{A/m^2}]$;
- в повітряному проміжку $\mu=\mu_0$, $J=0$ (струму немає);
- в статорі і роторі $\mu=\mu_0\left(\mu_{\min}+\dfrac{\mu_{\max}}{1+c\left\|\nabla u\right\|^2}\right)$, $J=0$, де $\left\|\nabla u\right\|^2=u_x'^2+u_y'^2$.

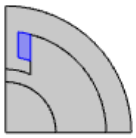
Константи $\mu_0=4\pi\cdot10^{-7}$, $\mu_{\min}=200$, $\mu_{\max}=5000$ і $c=0.05$ ми вже визначили у вузлі **Parameters**.

В нашому прикладі ми створимо три вузли `Variables` (для обвитки, для повітряного проміжку та для статора і ротора), в кожному з яких визначимо свої значення змінних μ та J.

Component 1(comp1) – Definitions: RMB > Variables
Definitions – Variables 1:

Geometric Entity Selection – **Geometric entity level** = `Domain`
Geometric Entity Selection – **Selection** = `Manual,1,3`
Variables – **Name, Expression** = `J, 0`
Variables – **Name, Expression** = `mu, mu0*(mumin+mumax/(1+c*(ux^2+uy^2)))`

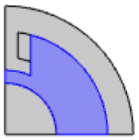
Component 1(comp1) – Definitions: RMB > Variables
Definitions – Variables 2:



Geometric Entity Selection – **Geometric entity level** = `Domain`
Geometric Entity Selection – **Selection** = `Manual,4`
Variables – **Name, Expression** = `J, 10`
Variables – **Name, Expression** = `mu, mu0`

Component 1(comp1) – Definitions: RMB > Variables
Definitions – Variables 3:

Geometric Entity Selection –
Geometric entity level = Domain



Geometric Entity Selection – **Selection** = Manual,2
Variables – **Name, Expression** = J, 0
Variables – **Name, Expression** = mu, mu0

Цим задаються змінні μ та J, які мають відмінні значення в різних частинах розрахункової області.

На створеній геометрії *генеруйте* СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** = Finer

LMB> Build All

Задайте рівняння. Перевірте, що в ПВ гілки **Poisson's Equation (poeq)** в розділі **Domain Selection** в секції вибору стоїть 1, тобто обрано коло. В дереві моделі виділіть вузол **Poisson's Equation 1**. В його панелі в розділі **Equation** наведено загальний вигляд рівняння Пуассона.

$$\nabla\cdot(-c\nabla u)=f\,,\quad \nabla=\left[\frac{\partial}{\partial x},\frac{\partial}{\partial y}\right],$$

і u позначає шуканий «магнітостатичний» потенціал. Задайте параметри рівняння.

Diffusion Coefficient – **c** = 1/mu
Source Term – **f** = J

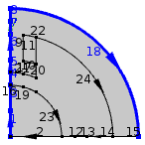
Зауважимо, що в рівнянні Пуассона коефіцієнт **c** = 1/mu та величина J будуть мати різні значення на різних частинах розрахункової області.

Задайте граничні умови. Для цього виконайте інструкції:

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = 1,3,4,5,6,7,8,18
Dirichlet Boundary Condition – **r** = 0

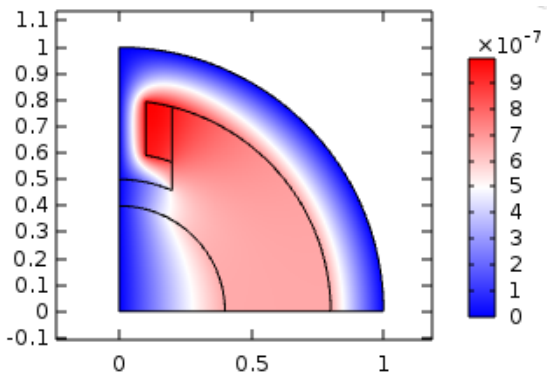


Умова симетрії $\boldsymbol{n}\cdot(\nabla A/\mu)=0$ на осі X задається автоматично у вузлі **Zero Flux 1**.

Виконайте обчислення.

Study 1 : LMB > Compute

Після завершення обчислень в графічній зоні вузла **2D Plot Group 1 – Surface 1** буде побудована розрахункова область з функціональним забарвленням, відповідним магнітному потенціалу u. Шкала кольорів поруч показує діапазон зміни його значень. Ми тільки змінили палітру кольорів на WaveLight.



Побудуємо еквіпотенціальні лінії «магнітостатичного» потенціалу.

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

2D Plot Group 2 : RMB > Contour

2D Plot Group 2 - Contour 1 :

`Expression` – **Expression** = `u` (*будувати лінії рівня u=Const*)

`Coloring and style` – **Color table** = `WaveLight`

Додамо на рисунок стрілкове зображення векторного поля $\boldsymbol{B}$ (магнітної індукції), яке в нашому прикладі має компоненти $\boldsymbol{B} = \left(u'_y, -u'_x\right)$.

2D Plot Group 2 : RMB > Arrow Surface

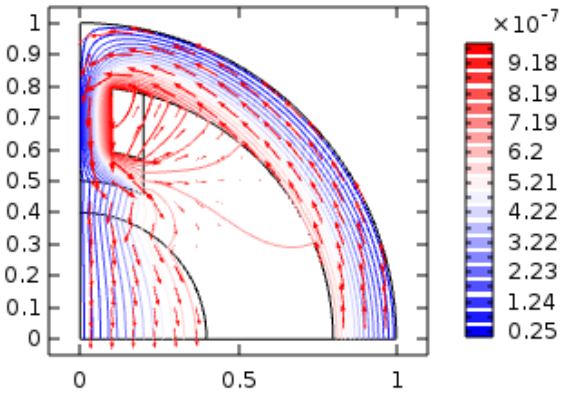
2D Plot Group 2 – Arrow Surface 1:

`Expression` – **x component, y component** = `uy,-ux`

`Coloring and Style` – **Arrow length** = `Logarithmic`

`Coloring and Style` – **Scale factor** = `9000`

LMB > Plot



Графік ясно показує, що поле магнітної індукції $\boldsymbol{B}$ спрямовано вздовж еквіпотенціальних ліній «магнітостатичного» потенціалу.

Вправа. Побудуйте самостійно графік «магнітостатичного» потенціалу на повній геометрії перерізу електромотора, продовживши розв'язок u через вісь симетрії X і антисиметрії Y.

3.14. Постійний струм в проводійному середовищі.

При дослідженні розповсюдження постійного струму в суцільному середовищі основною шуканою величиною є вектор густини струму $\boldsymbol{J}$. Він пов'язаний з напруженістю електричного поля $\boldsymbol{E}$ співвідношенням $\boldsymbol{J} = \sigma \boldsymbol{E}$, де σ - провідність середовища. Електричне поле $\boldsymbol{E}$ є градієнтом електричного

потенціалу V, $\boldsymbol{E} = -\operatorname{grad} V = -\nabla V$. Тоді закон збереження заряду $\nabla \cdot \boldsymbol{J} = \operatorname{div} \boldsymbol{J} = Q$, де Q – потужність джерел струму, призводить до рівняння Пуассона [8]

$$\nabla \cdot (-\sigma \nabla V) = Q.$$

Можливі наступні граничні умови:

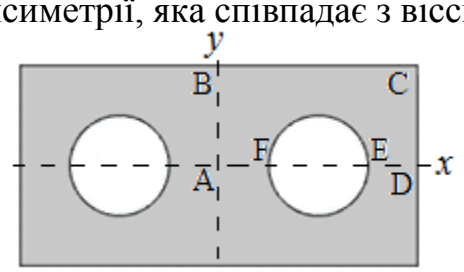
- умови Діріхле визначають значення потенціалу V на границі тіла, яка зазвичай є провідником;
- граничні умови Неймана визначають нормальну компоненту густини струму $\boldsymbol{J}_n = \boldsymbol{n} \cdot (\sigma \nabla V)$;
- узагальнені умови Неймана $\boldsymbol{n} \cdot (\sigma \nabla V) + q V = g$.

Приклад. Два циліндричних металевих провідника проходять крізь просякнутий розсолом промокальний папір, який є плоским тонким провідником з електропровідністю $\sigma = 1$. Математична модель задачі складається з рівняння Лапласа $\nabla \cdot (-\nabla u) = 0$, де u – потенціал поля, і граничних умов: $u = 1$ на контурі лівого провідника, $u = -1$ на контурі правого, та граничних умов Неймана $\boldsymbol{n} \cdot \nabla V = \partial u / \partial n = 0$ на зовнішніх межах паперу.

За допомогою майстра *створіть* 2D *модель задачі*

File->New -> Model Wizard -> 2D -> Mathematics - Classical PDEs - Laplace Equation (lpeq) -> Stationary

Нехай папір є прямокутником зі сторонами 4×2, і циліндричними отворами радіусом 0.5, розташованими симетрично, які утворюють провідники. Тоді модель має горизонтальну вісь симетрії, що співпадає з віссю X. Вона також має вертикальну вісь антисиметрії, яка співпадає з віссю Y.



Для заощадження комп'ютерних ресурсів розрахункову область можна побудувати як чверть повної геометрії, складену з чотирикутника ABCD з вирізаним напівколом EF.

Зауваження. Симетрія моделі означає, що вона повністю однакова з обох сторін роздільної лінії або площини, тобто однакові геометрія, матеріали, джерела поля, зовнішні впливи і граничні умови. Для скалярного поля це означає, що потік поля по нормалі крізь лінію симетрії дорівнює нулю. Антисиметрія означає, що зовнішній вплив, джерела поля і граничні умови протилежно збалансовані (мають протилежні значення або напрямки дії) по різні боки від розділювальної лінії (або площини). Для скалярного поля залежна змінна обертається в нуль на лінії антисиметрії. Якщо модель має подібні властивості, то можна будувати «геометрично повторювальну» частину розрахункової області (половину, чверть і т.д.), а на розділювальних осях застосовувати додаткові граничні умови симетрії або антисиметрії. Розв'язок

задачі в решті геометрії може бути отриманий шляхом «парного» або «непарного» продовження відносно цих осей.

Побудуйте розрахункову область.

Geometry **-> Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

Size – **Width, Height** = 2, 1 (*два поля*)

Geometry -> **Primitives** – Circle

Component 1 - Geometry 1 - Circle 1:

Size and Shape – **Radius:** = 0.5

Position – **x:** = 1

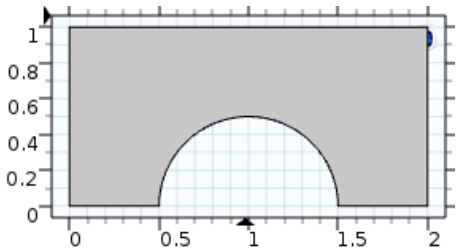
Geometry **-> Booleans and Partitions** – Difference

Component 1 - Geometry 1 - Difference 1:

Difference – **Object to add** = r1 (*прямокутник*)

Union – **Object to subtract** = c1 (*коло*)

LMB > Build All Objects



Зауваження. Досліджувати модель буде простіше, якщо розглядати її геометрію цілком. Але для демонстрування деяких можливостей системи буде корисно розрахункову область обрати як чверть всієї геометрії.

На створеній геометрії *генеруйте* СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** = Finer

LMB> Build All

Рівняння Лапласа, загальний вигляд якого наведено в ПВ вузла **Laplace Equation 1**, не потребує задання коефіцієнтів.

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection ,** <> = Manual,1 (*відрізок* AB)

Dirichlet Boundary Condition – **r** = 0

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection ,** <> = Manual,6,7 (*дуга* EF)

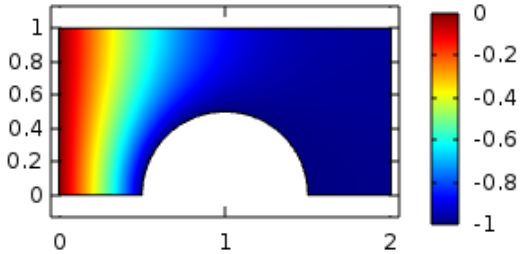
Dirichlet Boundary Condition – **r** = -1

На зовнішній межі (відрізки BC і CD) нормальна компонента густини струму дорівнює нулю, тобто діє умова Неймана $\partial u/\partial n = 0$. Умова симетрії на осі *x*, (граничні відрізки DE і AF) має такий же вигляд $\partial u/\partial n = 0$. Ці умови на ділянках границі, що залишилися, задані за замовчуванням у вузлі **Zero Flux 1**.

Виконайте обчислення.

Study 1 : LMB > Compute

Після завершення обчислень в графічній зоні вузла **2D Plot Group 1 – Surface 1** буде побудована розрахункова область з функціональним забарвленням. Шкала кольорів поруч показує діапазон зміни значень потенціалу *u*.



Побудуйте графік потенціалу на всій геометрії, продовживши розв'язок *u* через осі симетрії і антисиметрії. Для цього потрібно джерело даних, визначене на всій геометрії. Тут воно створюється з використанням джерел даних `Mirror 2D`. Ці джерела призначені для продовження розв'язку з деякої зони на область, що отримується об'єднанням первісної зони з власним віддзеркаленням відносно осі симетрії. При чому розв'язок *u* віддзеркалюється «парно» відносно осі.

Results – Data Sets: RMB > More Data Sets - Mirror 2D

Data Sets – Mirror 2D 1:

Data – **Data set** = `Study 1/Solution 1`

Axis Data – Point 1 – **x, y** = `0,0`

Axis Data – Point 2 – **x, y** = `2,0`

Цим задається горизонтальна вісь симетрії, яка проходить крізь вказані точки.

LMB > Plot (наступний рисунок ліворуч)

Отриману область віддзеркалимо відносно вертикальної осі Y.

Results – Data Sets: RMB > More Data Sets - Mirror 2D

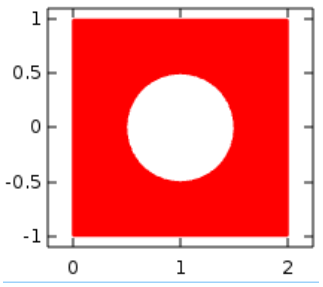
Data Sets – Mirror 2D 2:

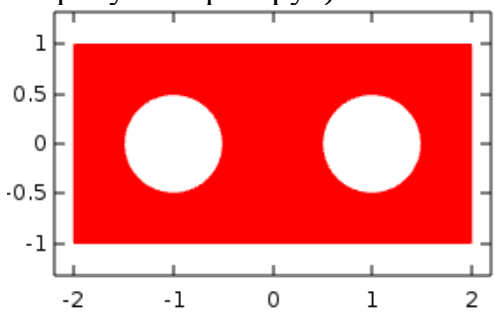
Data – **Data set** = `Mirror 2D 1`

Axis Data – Point 1 – **x, y** = `0,0`

Axis Data – Point 2 – **x, y** = `0,1`

LMB > Plot (наступний рисунок праворуч)



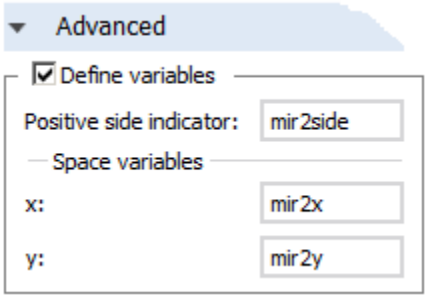


Майте на увазі, що на повній геометрії продовжений розв'язок $U(x,y)$ отримано з розв'язку $u(x,y)$ за формулами $U(x,y)=u(|x|,|y|)$, тобто парним продовженням відносно осей. Якщо на повній геометрії побудувати графік

виразу u, то насправді буде побудовано графік $U(x,y)$. Але це не те, що нам потрібно. Нам потрібно парне продовження відносно осі X, та непарне продовження відносно осі Y. Воно отримується за формулою

$$U(x,y)=\begin{cases} u(x,|y|), & x\ge 0 \\ -u(-x,|y|), & x<0 \end{cases} = \operatorname{sign} x\cdot u(|x|,|y|).$$

Щоб побудувати подібний вираз можна скористатися додатковою змінною mir2side, що визначається у вузлі **Mirror 2D 2**. Встановіть в цьому вузлі прапорець **Define variables**.



В полі **Positive side indicator** встановлено значення mir2side. Це ім'я змінної, яка має значення 1 у вихідній області і 0 – у віддзеркаленій. З цієї змінної можна побудувати вираз 2*mir2side -1, який буде дорівнювати +1 в початковій зоні і -1 (мінус один) – у віддзеркаленій. Тоді вираз, який буде коректно зображати антисиметричне продовження відносно осі *y* набуде вигляду (2*mir2side -1)*u. Саме його графік буде правильно зображати потенціал поля у повній геометрії.

Для спрощення подальших виразів, які будуть використовувати множник 2*mir2side-1, визначимо глобальну змінну pp, обчислювальну за цією формулою.

Global Definitions: RMB > Variables

Variables 1:

Variables – **Name, Expression** = pp, 2*mir2side-1

Не хвилюйтеся щодо помаранчевого кольору формули. На глобальному рівні змінна mir2side невідома, але при застосуванні ідентифікатора pp на даних джерела **Mirror 2D 2** вона набуде конкретного значення.

Щоб змінна стала доступною, потрібно виконати переобчислення моделі.

Study 1: RMB > Update Solution

Тепер перейдемо до побудови графіка потенціалу на всій геометрії.

Results -> 2D Plot Group

2D Plot Group 2:

Data – **Data set** = Mirror 2D 2

2D Plot Group 2: RMB > Surface

2D Plot Group 2 – Surface 1:

Expression – **Expression** = u*pp

LMB > Plot

Доповнимо рисунок лініями сталого значення потенціалу, додавши в графічну групу вузол Contour.

2D Plot Group 2: RMB > Contour

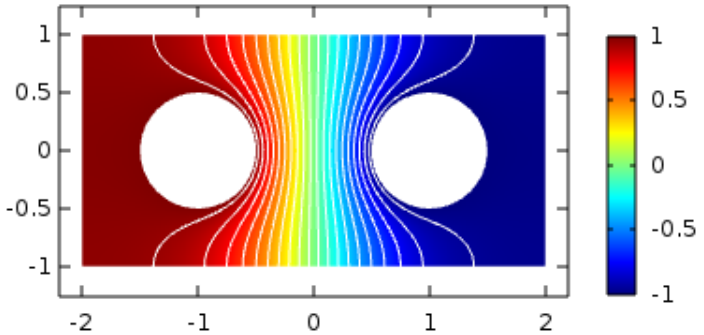
2D Plot Group 2 - Contour 1 :

Expression – **Expression** = u*pp

Coloring and style – **Coloring, Color** = Uniform, White

Coloring and style – **Color legend** = False

LMB > Plot



З графіка видно, що на осі Y потенціал *u* дорівнює нулю, оскільки вона є лінією антисиметрії задачі.

Побудуйте лінії електричного струму, тобто векторні лінії поля $\boldsymbol{J} = (-u_x, -u_y)$. Щоб їх накреслити у повній геометрії, потрібно будувати лінії поля з компонентами -ux*pp та -uy*pp.

Створіть графічну групу.

Results -> Plot Group - 2D Plot Group

2D Plot Group 3:

Data – **Data set** = Mirror 2D 2

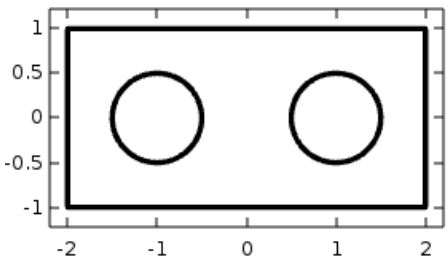
Побудуйте в графічній групі контур повної геометрії.

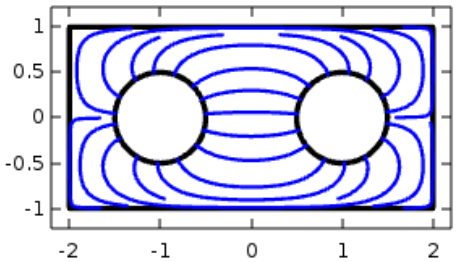
2D Plot Group 3 : RMB > Line

Expression – **Expression** = 1

Coloring and style – **Coloring, Color** = Uniform, Black

LMB > Plot (*наступний рисунок ліворуч*)





Додайте в графічну групу лінії поля.

2D Plot Group 3 : RMB > Streamline

2D Plot Group 3 – Streamline 1:

Expression – **x component, y component** = -ux*pp,-uy*pp

Streamline Positioning – **Positioning** = Uniformly density

Streamline Positioning – **Separation distance** = 0.05

При побудові ліній потрібно задати їх густину; у випадку рівномірного розташування (Uniformly density) число `0.05` (Separation distance) вказує деяку «відносну» відстань між лініями.

`Coloring and style` – Color = `Blue`

LMB > Plot (*попередній рисунок праворуч*)

Накладемо на отриманий рисунок зображення векторного поля $\boldsymbol{J} = (-u_x, -u_y)$ (густина струму) за допомогою `Arrow Surface` графіка.

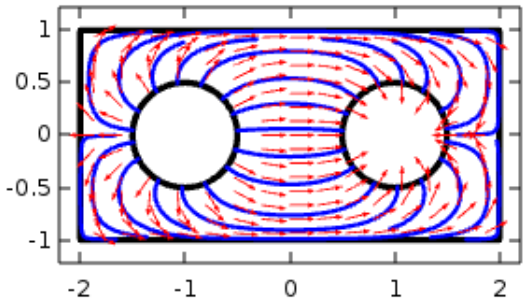
2D Plot Group 3 : RMB > Arrow Surface

2D Plot Group 3 – Arrow Surface 1:

`Expression` – x component, y component = `-ux*pp,-uy*pp`

`Coloring and Style` – Arrow length = `Normalized`

LMB > Plot



Електричний струм, як очкувалося, тече від провідника з позитивним потенціалом до провідника з негативним.

3.15. Дифракція плоскої хвилі на перешкоді.

Поширення хвильових процесів (електромагнітних, пружних, акустичних) супроводжується такими явищами як дифракція, заломлення, відбиття і т.ін. Одним з розділів, який вивчає ці процеси, є математична теорія дифракції [8]. В ній розглядається збурення поля плоскої хвилі в однорідному середовищі, яке має сторонні перешкоди. Хвиля може заломлюватися в опромінювані об'єкти, але тут ми розглянемо випадок повного відбиття.

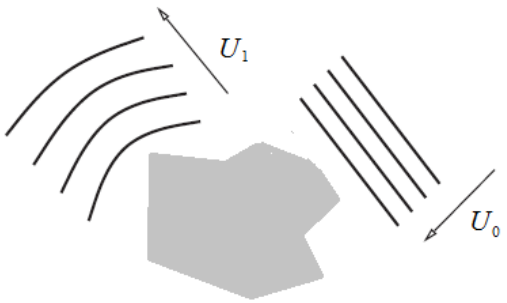
Плоскою хвилею, що розповсюджується в деякому заданому напрямку називається розв'язок U хвильового рівняння

$$\frac{\partial^2 U}{\partial t^2} - a^2 \Delta U = 0, \qquad (1)$$

який залежить від часу і від однієї просторової координати, відлічуваної в напрямку поширення. У разі усталеного коливального режиму, коли залежність від часу визначається множником $e^{-i\omega t}$, плоска хвиля, яка розповсюджується в X напрямку, має вигляд $U(x,t) = Ae^{i(-\omega t - kx)}$, де $k = \omega / a$ - хвильове число, $|A|$ - амплітуда. Плоска хвиля, що розповсюджується в напрямку одиничного вектора $\boldsymbol{l}(l_x, l_y, l_z)$, може бути представлена у вигляді $U(x,y,z,t) = Ae^{i(-\omega t - k\,\boldsymbol{l}\cdot\boldsymbol{r})}$, де $\boldsymbol{r} = (x, y, z)$.

Нехай поле в однорідному середовищі створюється деякими джерелами, розташованими поза зоною інтересів (на нескінченності), і має вигляд плоскої

хвилі $U_0(x,y,z,t)=Ae^{i(-\omega t-k\,\boldsymbol{l}\cdot\boldsymbol{r})}$. Справжнє поле U в зоні інтересів Ω можна представити у вигляді суми $U=U_0+U_1$ падаючої U_0 і відбитої хвилі U_1.



В усталеному процесі результівне поле $U(x,y,z,t)=w(x,y,z)e^{-i\omega t}$, падаюча $U_0(x,y,z,t)=u_0(x,y,z)e^{-i\omega t}$ і відбита хвиля $U_1(x,y,z,t)=u_1(x,y,z)e^{-i\omega t}$ будуть розв'язками хвильового рівняння (1). Підставляючи останні вирази в рівняння (1), отримаємо $-\omega^2 u_i-a^2\Delta u_i=0$, що набуває вигляду рівняння Гельмгольца

$$-\Delta u_i-k^2u_i=0\ \ (i=0,1), \tag{2}$$

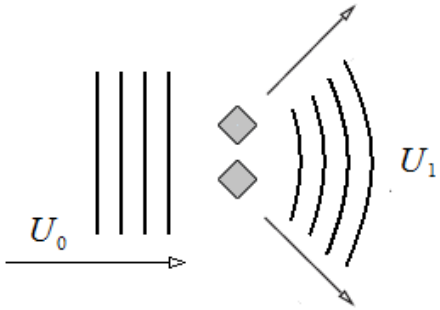
де k – хвильове число, яке пов'язано з частотою f, кутовою частотою ω та довжиною хвилі λ, співвідношеннями $k=\dfrac{\omega}{a}=\dfrac{2\pi f}{a}=\dfrac{2\pi}{\lambda}$. Розв'язки рівняння (2) зазвичай теж називають хвилями. Задача дифракції полягає у розв'язанні рівняння Гельмгольца $-\Delta u_1-k^2u_1=0$ відносно відбитої хвилі u_1, оскільки падаюча хвиля u_0 відома, а для результівної хвилі маємо $w=u_0+u_1$.

Розглянемо розсіювання плоскої хвилі на перешкоді, використовуючи модель «нескінченної» горизонтальної мембрани (коло досить великого радіуса). Перешкоди змоделюємо у вигляді абсолютно жорстких тіл. На границях перешкод $\partial\Omega$ загальний розв'язок w буде перетворюватися на нуль $w=0$. Оскільки $w=u_0+u_1$, то на межі $\partial\Omega$ для відбитої хвилі матимемо $u_1\big|_{\partial\Omega}=-u_0\big|_{\partial\Omega}$.

Відбита хвиля поширюється назовні від об'єктів. Умова на зовнішній межі розрахункової області повинна бути такою, щоб на ній не утворювалися нові відбиті хвилі. Для цього використаємо умову випромінювання Зомерфельда $\lim\limits_{r\to\infty} r^{(n-1)/2}\left(\dfrac{\partial u_1}{\partial r}-iku_1\right)=0$ [6], де n – просторова розмірність і r – відстань по радіусу від перешкоди. Ми її застосуємо на «далекій» (але не безмежно) границі розрахункової області, і для спрощення будемо вважати, що похідній $\partial u/\partial r$ відповідає похідна по зовнішній нормалі $\boldsymbol{n}$ зовнішньої межі обчислювальної зони. Тоді, відкидаючи операцію $\lim$ і дорівнюючи нулю вираз в дужках, отримуємо наближену форму граничної умови Зомерфельда на зовнішній межі у вигляді $\boldsymbol{n}\cdot\nabla u_1=iku_1$.

Приклад. Розглянемо розсіювання плоскої хвилі $U_0(x,y,t)=e^{i(-\omega t-kx)}$, яка розповсюджується в напрямку осі X, на двох абсолютно жорстких квадратних

перешкодах. Квадрати повернуто на 45^0 і зсунуто по вертикалі одне від одного приблизно на довжину хвилі.



Математична модель задачі полягатиме в розв'язанні рівняння Гельмгольца (2) $\Delta u_1 + k^2 u_1 = 0$ з граничною умовою Діріхле $u_1 = -u_0 = -e^{-ikx}$ на контурах квадратів, і узагальненою умовою Неймана $\boldsymbol{n} \cdot \nabla u_1 = ik u_1$ на зовнішній межі розрахункової зони. Саму розрахункову зону зобразимо у вигляді кола з центром в точці (0,0) достатньо великого розміру з отворами у формі двох повернутих квадратів, розміром 0.1, і з центрами в точках (0,0.1), (0,-0.1). Оскільки при моделюванні розрахункової області її зовнішня межа повинна бути розташована на відстані не менше кількох розмірів розсіювальних об'єктів, тому оберемо радіус кола 0.65.

За допомогою майстра *створіть шаблон моделі*

File->New -> Model Wizard -> 2D -> Mathematics - Classical PDEs - Helmholtz Equation (hzeq) -> Stationary

Вимкніть в моделі підтримку розмірностей фізичних величин.

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

Побудуйте розрахункову область.

Geometry -> **Primitives** – Circle

Component 1 - Geometry 1 - Circle 1 (c1):

Size and Shape – **Radius:** = 0.65

Накресліть перешкоди у вигляді квадратів, повернутих на кут 45^0.

Geometry -> **Primitives** – Square

Geometry 1 - Square 1 (sq1):

Size – **Size length** = 0.1

Position – **Base** = Center

Position – **x, y** = 0, 0.1

Rotation Angle – **Rotation** = 45

Geometry 1 - Square 1 (sq1) : RMB >Duplicate

Geometry 1 - Square 2 (sq2):

Position – **x, y** = 0, -0.1

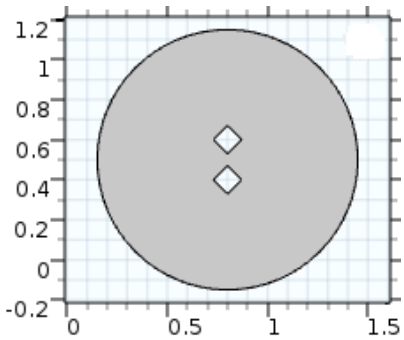
Geometry -> **Booleans and Partitions** – Difference

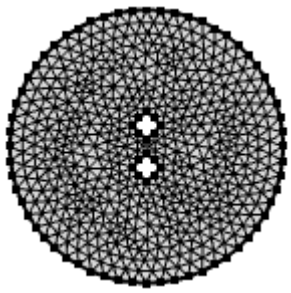
Component 1 - Geometry 1 - Difference 1:

Difference – **Object to add** = c1 (*коло*)

Union – **Object to subtract** = sq1,sq2 (*квадрати*)

LMB > Build All Objects (*наступний рисунок ліворуч*)





На створеній геометрії *генеруйте* СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – Element Size = `Fine`

LMB> Build All (*попередній рисунок праворуч*)

Для того, щоб обчислювання були виконані з достатньою точністю, потрібно мати приблизно 10 скінченних елементів на довжині хвилі, тобто в околиці перешкод варто було б згустити сітку. Але це можна довірити алгоритму адаптивного солвера (`Adaptive solver`). Він враховує конфігурацію області і при необхідності включає здрібнювання СкЕ сітки. Підключіть його.

Study 1 - Step 1: Stationary:

`Study extensions` – Adaptive mesh refinement = `True`

Задайте параметри моделі. Щоб дифракційні ефекти були помітні, довжина хвилі повинна приблизно відповідати розміру перешкод. У нас характерним розміром є 0.1. Враховуючи співвідношення $k = 2\pi/\lambda$, підставимо сюди замість λ розмір 0.1 і отримаємо, що $k \approx 60$. Задайте його в таблиці параметрів.

Global Definitions : RMB > Parameters

Global Definitions – Parameters :

`Parameters` – Name, Expression = `k, 60`

Задайте коефіцієнти *рівняння*.

Helmholtz Equation (hzeq) - Helmholtz Equation 1:

`Diffusion Coefficient` – c = `1`

`Source Term` – f = `0`

`Absorption Coefficient` – a = `-k^2`

Цим задається рівняння $\nabla\cdot(-\nabla u) - k^2 u = 0$, і величина u буде репрезентувати відбиту хвилю (тобто ту, що ми раніше позначали як u_1).

Задайте граничні умови. На межі квадратних перешкод повинна виконуватися умова Діріхле $u_1 = -u_0 = -e^{-ikx}$.

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

`Boundary Selection`– Selection=`1,2,3,4,5,6,7,8` (*контури квадратів*)

`Dirichlet Boundary Condition` – r = `- exp(-i*k*x)`

Умову на зовнішній межі розрахункової області можна задати, використовуючи вузол типу `Flux/Source`. В таких вузлах задається умова $-\boldsymbol{n}\cdot(-c\nabla u) = g - qu$. Щоб вона набула вигляду $\boldsymbol{n}\cdot\nabla u = iku$, потрібно ввести $g = 0, q = -ik$.

`Physics` -> **`Boundaries`** – Flux/Source

Helmholtz Equation (hzeq) - Flux/Source 1:

Boundary Selection – **Selection** = `Manual,9,10,11,12`

(*зовнішня межа кола*)

Boundary Flux/Source – **g** = `0`

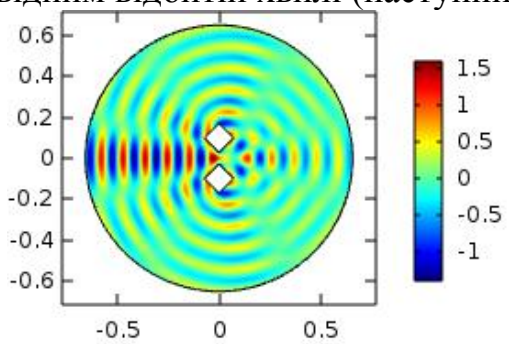
Boundary Absorption/Impedance Term – **q** = `-i*k`

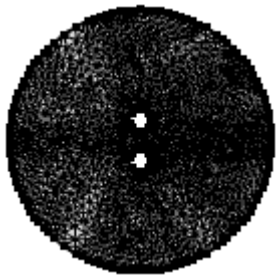
Зверніть увагу, що q є комплексним числом. Не турбуйтеся до цього, бо COMSOL Multiphysics чудово працює з комплексними величинами. Вам тільки треба не забувати, що отриманий розв'язок буде комплексним.

Виконайте обчислення.

Study 1 : LMB > Compute

Після завершення обчислень в графічній зоні вузла **2D Plot Group 1 – Surface 1** буде побудована розрахункова область з функціональним забарвленням, відповідним відбитій хвилі (наступний рисунок ліворуч).





Зазначимо, що дані для графічної гілки **2D Plot Group 1** вибираються з джерела даних **Study 1/Adaptive Mesh Refinement 1**. Воно створено автоматично після підключення адаптивного солвера (`Adaptive solver`). Цей солвер також автоматично створив вузол **Component 1 (comp1) – Meshes – Mesh 2**. Активуйте його і подивіться на створену СкЕ сітку (попередній рисунок праворуч). Вона значно густіша ніж та, що побудована у вузлі **Mesh 1**, і на яку опирається ДД **Study 1/Solution 1**.

Ще зверніть увагу на те, що розв'язок `u` (`u` позначає відбиту хвилю u_1) повинен бути комплексним. Але хоча у вузлі **Surface 1** використано вираз `u`, побудовано графік дійсної функції. Це тому, що коли в графічних вузлах використовуються комплекснозначні вирази, то система будує графіки їх дійсних частин, тобто насправді у вузлі **Surface 1** побудовано графік виразу `real(u)`.

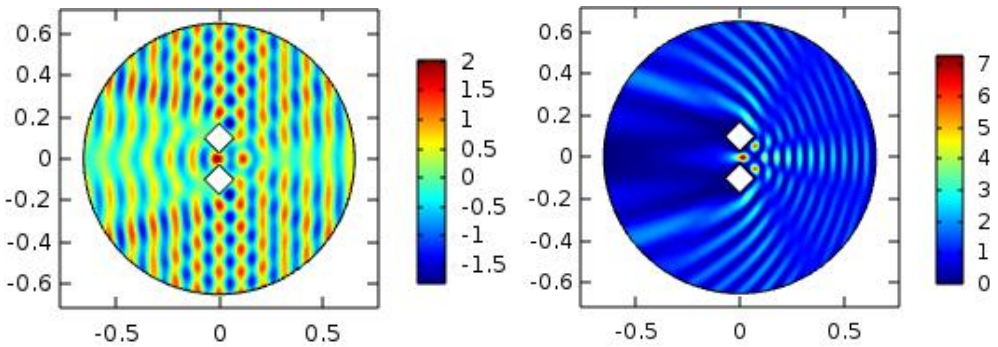
Результівна хвиля обчислюється за формулою $w = u_1 + u_0 = u_1 + e^{-ikx}$. Побудуємо її графік.

2D Plot Group 1: RMB > Duplicate

2D Plot Group 2 – Surface 1:

Expression – **Expression** = `real(u+exp(-i*k*x))`

2D Plot Group 2: LMB > Plot (*наступний рисунок ліворуч*)



Амплітуда падаючої хвилі дорівнює одиниці, але в деяких точках результівна хвиля посилюється (амплітуда зростає по модулю), а в деяких – слабшає.

Зазвичай інтенсивність поля, хвилі якого розповсюджуються, пропорційна квадрату амплітуди хвилі, тобто добутку комплекснозначної амплітуди на величину їй спряжену $I \sim w \cdot \overline{w}$. В нашому випадку величини w, u_0 і u_1 є комплексними амплітудами результівної, падаючої і відбитої хвиль.

Побудуємо графік інтенсивності $I = \alpha\, w \cdot \overline{w}$ результівного поля, поклавши $\alpha = 1$.

2D Plot Group 2: RMB > Duplicate

2D Plot Group 3 – Surface 1:

`Expression` – **Expression** = `(u+exp(-i*k*x))*conj(u+exp(-i*k*x))`

2D Plot Group 3: LMB > Plot (*попередній рисунок праворуч*)

Тут функція `conj(...)` позначає комплексне спряження.

Зауваження про умову Зомерфельда. В поточному прикладі ми зустрілися з необхідністю моделювання області з відкритою зовнішньою межею, через яку хвилі проходили би без відбиття. COMSOL Multiphysics пропонує для цього кілька можливих рішень. Тут ми використали граничну умову випромінювання Зомерфельда. Ця умова є цілком невідбивною, коли межа області моделювання нескінченно віддалена від джерела випромінювання, але моделювання нескінченно великої області неможливо. Тому, хоча ми не можемо застосувати в точності умову Зомерфельда, ми застосували її прийнятне наближення у вигляді $\boldsymbol{n} \cdot \nabla u - i k u = 0$, Воно називається граничною умовою розсіювання першого порядку (scattering boundary condition - SBC). По суті, це гранична умова третього роду з комплекснозначними коефіцієнтами. Але є істотне обмеження для цієї умови, якщо вона застосовується на скінченній відстані. Воно полягає в тому, що умова є невідбивною, тільки коли випромінювання падає на границю уздовж нормалі. Будь-які хвилі, що падають неперпендікулярно на SBC-межу будуть частково відбиватися. Коефіцієнт відбиття плоскої хвилі, що падає на SBC-межу зростає з ростом кута падіння. В деяких інтерфейсах COMSOL Multiphysics включає в себе SBC-умову другого порядку, яка трохи краще імітує повне поглинання, ніж умова першого роду. Однак бажано мати «інструмент», який дозволить мінімізувати будь-які відбиті сигнали. Цього можна досягти, якщо зовнішність розрахункової області зобразити деяким ідеально поглинаючим шаром (perfectly matched layer - PML). Він не є граничною умовою, а областю, пристикованою до зовнішньої межі моделі, яка поглинає всі вихідні хвилі. Хоча PML-шари, суто теоретично, є

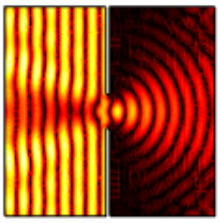
невідбивними, вони, все одно, виявляють деякі відбивні властивості через помилки чисельної дискретизації на СкЕ сітці. Щоб це мінімізувати, необхідно коректно будувати сіткове розбиття розрахункової області. Застосування PML-шару є кращим порівняно з підходом, що використовує SBC-граничні умови, однак, його реалізація потребує більше комп'ютерних ресурсів.

Для використання PML шару в COMSOL Multiphysics, по-перше, потрібно побудувати геометричну зону навкруги області інтересів, яка буде його зображати. По-друге, в гілці **Definitions** компонента потрібно створити вузол типу `Perfectly Matched Layer`, в якому визначити цю зону, як PML шар, та налаштувати його властивості.

Нагадаємо, що в COMSOL Multiphysics існує ще один інструмент – `Infinite Element Domain`. Він імітує нескінченну зону, яка в деяких задачах може замінити моделювання невідбивних граничних умов.

Вправа 1. Дослідіть дифракцію плоских хвиль на єдиній квадратній перешкоді, коли довжина хвилі порівняна з розміром квадрата.

Вправа 2. Дослідіть дифракцію плоских хвиль, що падають на щілину завширшки з довжину хвилі.



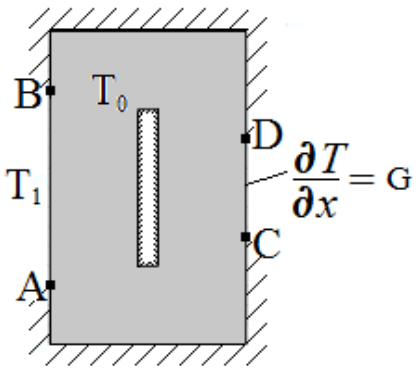
3.16. Розповсюдження тепла в платівці з прямокутним отвором.

Процес розповсюдження тепла описується рівнянням теплопровідності [1]

$$d\frac{\partial T}{\partial t}+\nabla\cdot\left(-c\nabla T\right)=f\,, \qquad (1)$$

де $T\left(x,y,z,t\right)$ шукана функція температури, d – питома теплоємність, c – коефіцієнт теплопровідності, f – питома потужність джерел тепла.

Приклад. Розглянемо задачу розповсюдження тепла в прямокутній платівці з прямокутним отвором всередині. Нехай джерела тепла відсутні, теплоємність і коефіцієнт теплопровідності дорівнюють одиницям (в системі СІ). Середня частина лівої межі платівки підтримується при температурі $T_1=100\ ^0C$, початкова температура платівки дорівнює $T_0=40\ ^0C$, на середній частині правої межі задано тепловий потік в навколишнє середовище $G=-100\,w/m^2$ (G від'ємне, бо тепло витікає з області). Решта границь платівки теплоізольована.



Це призводить до наступних граничних умов:

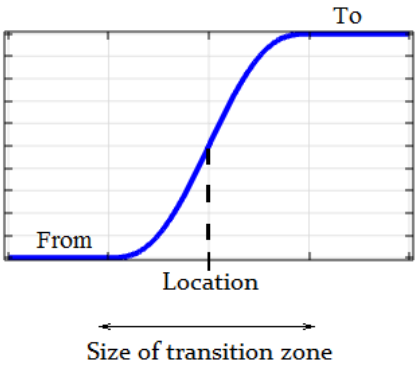
- $T = T_1$ на відрізку AB (умова Діріхле);
- $\dfrac{\partial T}{\partial x} = G$ на відрізку CD (умова Неймана);
- $\dfrac{\partial T}{\partial \boldsymbol{n}} = 0$ на решті частині границі, де $\dfrac{\partial}{\partial \boldsymbol{n}}$ похідна по нормалі.

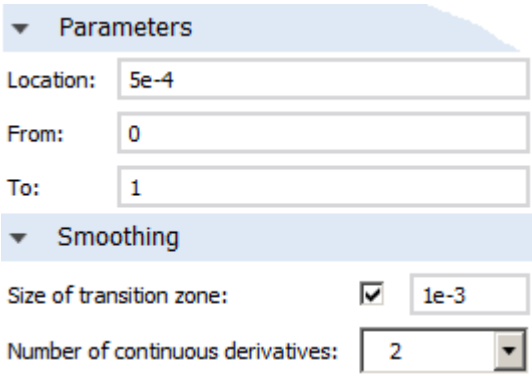
За допомогою майстра *створіть* 2D *модель задачі.*

File -> New -> Model Wizard -> 2D -> Mathematics – Classical PDEs – Heat Equation (hteq) + Dependent variable = T -> Time Dependent

Цим створено шаблон моделі з ім'ям шуканої змінної T і фізичним змістом температури.

Для коректного задання граничної температури, яка в нашому прикладі не погоджена з початковою, знадобитися функція-сходинка. В COMSOL це неперервна гладка функція зі швидким переходом від одного значення до іншого. Вона визначається положенням середини перехідної зони (`Location`), шириною цієї зони (`Size of transition zone`), початковим (`From`) і кінцевим значенням (`To`).





При активному вузлі **Global Definitions** виконайте послідовність інструкцій:

`Home->Functions->Step`

Globsl Definitions - Step 1 (step1):

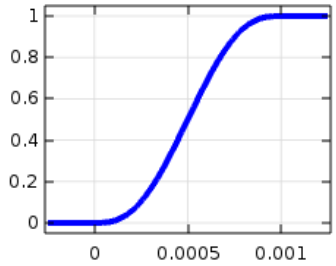
`Parameters` – **Location** = `5e-4`

`Parameters` – **From** = `0`

`Parameters` – **To** = `1`

`Smoothing` – **Size of transition zone** = `0.001`

LMB > Plot



Побудуйте геометрію моделі.

Geometry **-> Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

Size – **Width, Height** = 1, 1.6 (*два поля*)

Position – **x, y** = -0.5,-0.8 (*два поля*)

Geometry **-> Primitives** – Rectangle

Component 1 - Geometry 1 – Rectangle2:

Size – **Width, Height** = 0.1, 0.8 (*два поля*)

Position – **x, y** = -0.05,-0.4 (*два поля*)

Geometry **-> Booleans and Partitions** – Difference

Component 1 - Geometry 1 - Difference 1:

Difference – **Object to add** = r1 (*зовнішній прямокутник*)

Union – **Object to subtract** = r2 (*внутрішній прямокутник*)

LMB > Build All Objects (*наступний рисунок ліворуч*)

В цій задачі потрібно задавати різні граничні умови на різних ділянках межі зовнішнього прямокутника. Щоб пізніше ці частини можна було виділити, нам знадобляться допоміжні точки.

Geometry -> **Primitives** – Point

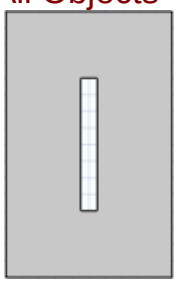
Component 1 - Geometry 1 - Point 1:

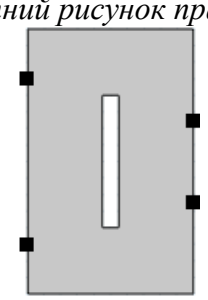
Point – **x** = 0.5, 0.5, -0.5, -0.5

Point – **y** = -0.25, 0.25, -0.5, 0.5

Тут окремо було задано абсциси і окремо ординати точок.

LMB > Build All Objects (*наступний рисунок праворуч*)





На створеній геометрії *генеруйте* СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** = Finer

LMB> Build All

При побудови моделі знадобляться деякі сталі. Задайте їх в таблиці глобальних параметрів.

Global Definitions : RMB > Parameters

Заповніть таблицю параметрів так, як показано на рисунку.

Parameters

Name	Expression	Value	Description
t0	0.001[s]	0.001 s	Перший момент часу
t1	20[s]	20 s	Останній момент часу
N	100	100	Кількість часових інтервалів
dlt	(log(t1[1/s])-log(t0[1/s]))/N	0.099035	Логарифмічний часовий крок
T0	40[degC]	313.15 K	Початкова температура
T1	100[degC]	373.15 K	Температура відрізка AB
G	-100[W/m^2]	-100 W/m²	Тепловий потік на відрізку CD

Стисле пояснення сенсу сталих наведено в стовпці `Description` цієї таблиці.

Задайте рівняння. В ПВ вузла **Heat Equation (hteq) - Heat Equation 1** наведено загальний вигляд рівняння теплопровідності

$$d_a \frac{\partial u}{\partial t} + \nabla \cdot (-c \nabla u) = f, \ \nabla = \frac{\partial}{\partial x}.$$

Коефіцієнти рівняння задаються в полях в нижній частині цієї панелі.

Heat Equation (hteq) - Heat Equation 1 :

Diffusion Coefficient – **c** = 1

Source Term – **f** = 0

Absorption Coefficient – $\mathbf{d_a}$ = 1

Зверніть увагу на те, що одиниці вимірювання параметрів **c**, **f** і $\mathbf{d_a}$ вказані праворуч від відповідних полів. Ці одиниці використовуються за замовчуванням, якщо в полях стоять безрозмірні величини.

Задайте початкові умови. Перевірте, що в ПВ вузла **Initial Values 1** в секції вибору стоїть індикатор 1. Потім введіть

Initial Values 1:

Initial Values – **Initial Value for u:** – **u** = T0

Задайте граничні умови. На відрізку AB (див. рисунок на початку) потрібно задати температуру T_1. Оскільки на боках прямокутника створені допоміжні точки, то при заданні граничних умов можна виділяти частини сторін.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,3 (*відрізок* AB)

Dirichlet Boundary Condition – **r** = T0+step1(t[1/s])*(T1-T0)

Використання формули `T0+step1(t[1/s])*(T1-T0)` призводить до того, що в нульовий момент часу гранична температура дорівнюватиме `T0` (бо `step1(0)=0`), але практично відразу зростатиме до `T1` (бо `step1(0.001)=1`). Тобто температура відрізка в будь-який більший `0.001сек` момент часу дорівнює `T1`, і при цьому усувається стрибок температури в околиці відрізка AB в нульовий момент часу.

Зауваження. Поточна задача з успіхом розв'язується без використання функції-сходинки. Але в деяких задачах наявність стрибка у шуканої величини в початковий момент часу може призводити до великої обчислюваної похибки.

Умова на відрізку CD задається з використанням вузла типу `Flux/Source`. В таких вузлах задається умова $-\boldsymbol{n} \cdot (-c \nabla u) = g - qu$. Щоб вона набула вигляду

$\frac{\partial T}{\partial n} = G$, потрібно ввести $g = G, q = 0$ (G повинно бути від'ємним, оскільки вважаємо, що тепло витікає з області назовні).

`Physics` -> **`Boundaries`** – Flux/Source

Helmholtz Equation (hzeq) - Flux/Source 1:

`Boundary Selection` – **Selection** = `Manual,11` (*відрізок* CD)

`Boundary Flux/Source` – **g** = `G`

`Boundary Absorption/Impedance Term` – **q** = `0`

Виконайте налаштування розв'язувача.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times** =

`exp(range(log(t0[1/s]),dlt,log(t1[1/s])))`

Оскільки процес остигання/нагрівання іде швидко на початку процесу, то список моментів часу ми зробили нерівномірним: спочатку інтервал між значеннями невеликий, а потім поступово зростає. Задавати список, наприклад, зі 100 моментів часу вручну небажано, тому ми вигадали формулу. Якби стала N дорівнювала 10, то формула згенерувала би наступний список:

`0.001,0.0027,0.00723,0.0195,0.052,0.1414,0.381,1.025,2.76,7.43,20`

Його елементи щільніше розташовані на початку. При N=100 ми отримаємо схожу послідовність зі 101 числа, також щільніше розташованих на початку. Якщо проаналізувати введену формулу, то побачимо, що внутрішній список `range(log(t0),dlt,log(t1))` створює послідовність зі сталим кроком `dlt` і значеннями від `log(t0)` до `log(t1)`. Використання функції `exp(range(...))` створює вже нерівномірну послідовність зі значеннями від `t0` до `t1` і «експоненціально» зростаючим кроком.

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Line Graph 1**, що містить графік розв'язку *T* в останній зі списку момент часу (`t=20 сек`). Ввімкніть на графіку відображення екстремальних значень і змініть палітру кольорів.

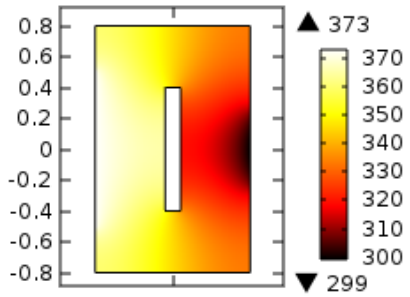
2D Plot Group 1 :

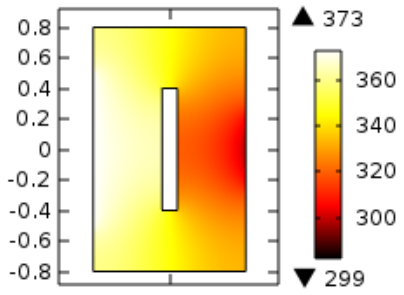
`Color Legend` - **Show maximum and minimum values** = `True`

2D Plot Group 1 - Surface 1 :

`Coloring and style` – **Color Table** = `Thermal`

LMB > Plot (*наступний рисунок ліворуч*)





Майте на увазі, що змінення часу в полі **Time (s)** вузла **2D Plot Group 1**, змінює мінімальне і максимальне значення температури (це екстремуми по області в фіксований момент часу), що приводить до зміни діапазону шкали кольорів. В нашому прикладі чорний колір кожний раз буде відповідати мінімальній температурі. Бажано, щоб шкала кольорів не змінювалася від обраного моменту часу, бо це вводить в оману. Запобігти такій зміні дозволяє ручне налаштування діапазону кольорів. Проклацавши всі моменти часу (простіше спершу побудувавши анімацію), можна визначити що мінімальне значення температури трохи більше за $282\,^0C$. Встановіть це значення в якості мінімального.

2D Plot Group 1 - Surface 1 :

Range– Manual color range = `True`

Range– Minimum = `282`

LMB > Plot (*попередній рисунок праворуч*)

Хоча обидва попередні графіка відповідають одному моменту часу, вони трохи відрізняються, бо на правому графіку чорному кольору відповідає температура $282\,^0C$. Оскільки це значення в розглядуваний момент часу `t=20 сек` не досягається ні в одній точці області, то зовсім чорний колір на графіку відсутній.

Для спостереження за розв'язком можна побудувати анімацію.

`Results` –> Export –> Animation – Player

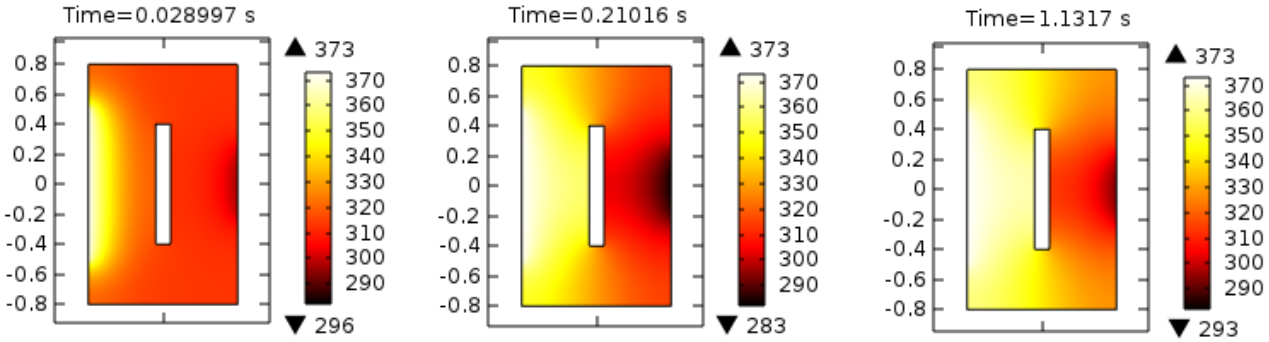
Export - Animation 1:

Scene– Subject = `1D Plot Group 1`

Frames– Frame selection = `All`

LMB> Generate Frame

Запустіть анімацію на виконання, клацнувши мишею по кнопці Play . На наступному рисунку наведено 3 кадри з зображенням розподілу температури в моменти часу, які надруковані вгорі графіків. Причому кольорова палітра цих кадрів однакова.



Побудуйте лінійний графік температури вздовж горизонтальної осі симетрії, тобто вздовж відрізка $y=0$. Спочатку створіть джерело даних, яке відповідає цьому відрізку.

`Results` –> Data Set - Cut Line 2D

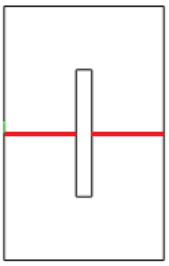
Results – Data Sets – Cut Line 2D 1:

Data – Data set = `Study 1/Solution 1`

Line Data – **Point 1 – x** = -0.5

Line Data – **Point 2 – x** = 0.5

LMB > Plot



Потім створіть групу одновимірних графіків і налаштуйте її.

Results -> Plot Group - 1D Plot Group

1D Plot Group 2 :

Data – **Data set** = Cut Line 2D 1

Data –**Time selection** = Manual

Data –**Times (1-121)** = 6,29,40,50,60,70,98 (*номера моментів часу*)

1D Plot Group 2-> Line Graph

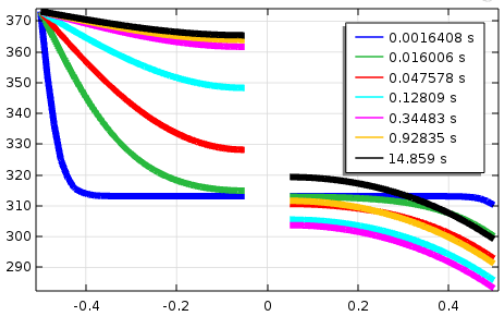
1D Plot Group 2 – Line Graph 1:

y-Axis Data – **Expression** = T

x-Axis Data – **Parameter, Expression** = Expression, x (*два поля*)

Legends–**Show legends** = True

LMB > Plot

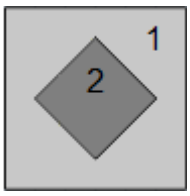


Зверніть увагу на розрив кривої. Він з'являється внаслідок розриву джерела даних, тобто відрізка, вздовж якого будується графік.

Вправа. Побудуйте самостійно анімацію лінійного графіка температури вздовж горизонтальної осі симетрії $y = 0$.

3.17. Теплопровідність в неоднорідному середовищі.

Розглянемо задачу розповсюдження тепла в квадратній платівці з квадратним вкладенням всередині з іншого матеріалу. Внутрішній квадрат повернуто на 45^0 відносно зовнішнього.



Матеріал зовнішнього квадрата має теплопровідність 10 і густину 2 одиниці. Внутрішній квадрат є джерелом тепла потужності 4, з коефіцієнтом теплопровідності 2 і густиною 1. Теплоємність обох матеріалів дорівнює 0.1. Початкова температура 0. Температура зовнішніх границь підтримується нульовою. На внутрішній межі витримується неперервність температури і теплового потоку.

Рівняння теплопровідності для кожного матеріалу має вигляд [1]

$$\rho C \frac{\partial T}{\partial t} - \nabla \cdot (k \nabla T) = Q, \qquad (1)$$

де ρ - густина матеріалу, C – теплоємність, k – коефіцієнт теплопровідності, Q – потужність внутрішніх джерел тепла.

Для розв'язання задач з кількома матеріалами в COMSOL Multiphysics існують різноманітні підходи. Один з них полягає в використанні для всіх частин геометрії єдиного рівняння (1) з відмінними значеннями матеріальних коефіцієнтів в різних частинах розрахункової області. Можна також розв'язувати рівняння (1) для кожної підобласті. Тут ми продемонструємо обидва способи. Спочатку ми побудуємо шаблон моделі з геометрією і збережемо його в два різних файли, а потім почнемо в ДМ цих файлів додавати інструкції, які потрібно виконати при відповідному підході.

За допомогою майстра *створіть* 2D *модель задачі.*

File->New -> Model Wizard -> 2D -> Mathematics – Classical PDEs – Heat Equation (hteq) + Dependent variable = T -> Time Dependent

Цим створено шаблон моделі з ім'ям шуканої змінної T і фізичним змістом температури.

Побудуйте геометрію моделі. Накресліть зовнішній квадрат

Geometry -> **Primitives** – Square

Geometry 1 - Square 1 (sq1):

Size – **Size length** = 3

Накресліть внутрішній квадрат, який повернуто на кут 45^0.

Geometry 1 - Square 1 (sq1) : RMB >Duplicate

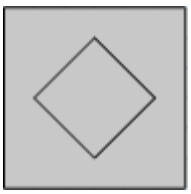
Geometry 1 - Square 2 (sq2):

Size – **Size length** = sqrt(2)

Position – **x, y** = 1.5, 0.5

Rotation Angle – **Rotation** = 45

LMB > Build All Objects



Зверніть увагу на те, що гілка **Geometry 1** за замовчуванням завершується вузлом **Form Union (fin)**, який виконує об'єднання всіх створених в гілці об'єктів.

Збережіть файл з ім'ям, наприклад, `Heat_in_Composite_Env_1.mph`, а потім з іншим ім'ям, наприклад, `Heat_in_Composite_Env_02.mph`. Далі в них будуть реалізовані два різних підходи розв'язання задачі.

Розв'язок 1. Відкрийте файл `Heat_in_Composite_Env_1.mph`. Створіть два вузли `Variables`, які будуть містити значення матеріальних параметрів для зовнішньої і внутрішньої підобластей.

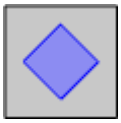
Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 1:

Geometric Entity Selection –
Geometric entity level = Domain
Geometric Entity Selection – Selection = Manual,1
Variables – Name, Expression = C, 0.1
Variables – Name, Expression = rho,2
Variables – Name, Expression = k,10
Variables – Name, Expression = Q,0

Component 1(comp1) – Definitions: RMB > Variables

Definitions – Variables 2:



Geometric Entity Selection –
Geometric entity level = Domain
Geometric Entity Selection – Selection = Manual,2
Variables – Name, Expression = C,0.1
Variables – Name, Expression = rho,1
Variables – Name, Expression = k,2
Variables – Name, Expression = Q,4

Цим задаються параметри C, ρ, k, Q рівняння (1), які мають відмінні значення в різних частинах розрахункової області (насправді C однакове).

Генеруйте СкЕ *сітку* з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Задайте рівняння. Перевірте, що в ПВ гілки **Heat Equation (hyeq)** в розділі **Domain Selection** в секції вибору стоять індикатори 1 і 2, тобто обрано обидві підобласті. В дереві моделі виділіть вузол **Heat Equation 1**. В його панелі в розділі **Equation** наведено загальний вигляд рівняння теплопровідності

$$d_a \frac{\partial T}{\partial t} + \nabla \cdot (-c \nabla T) = f, \quad \nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right],$$

де T позначає шукану функцію температури. Коефіцієнти рівняння задаються в полях в нижній частині цієї панелі.

Heat Equation (hteq) - Heat Equation 1 :

Diffusion Coefficient – c = k
Source Term – f = Q
Absorption Coefficient – d_a = rho*C

Нагадаємо, що в рівнянні теплопровідності коефіцієнти **c** = `k` , $\mathbf{d_a}$ = `rho*C` та величина **f** = `Q` будуть мати різні значення на різних частинах розрахункової області.

Нульові початкові умови задаються за замовчуванням, і їх міняти непотрібно.

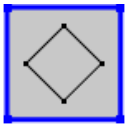
Задайте граничні умови.

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

`Boundary Selection` – **Selection** = `1,2,3,8`

`Dirichlet Boundary Condition` – **r** = `0`



На внутрішніх границях умова неперервності температури і теплового потоку виконується автоматично.

Налаштуйте розв'язувач.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times (s)** = `range(0,0.01,0.1)`

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Surface 1**, що містить графік температури *T* в останній зі списку **Times (s)** момент часу (`t=0.1`). Змініть на графіку палітру кольорів і додайте стрілковий графік, що зображуватиме тепловий потік $\boldsymbol{q} = -k\nabla T = \left(-kT_x, -kT_y\right)$.

2D Plot Group 1 - Surface 1 :

`Coloring and style` – **Color Table** = `Thermal`

2D Plot Group 1 : RMB > Arrow Surface

2D Plot Group 1 – Arrow Surface 1:

`Expression` – **x component, y component** = `-k*Tx,-k*Ty`

`Arrow Positioning` – **x grid points – Entry method, Points** = `Number of points,25` (*два поля*)

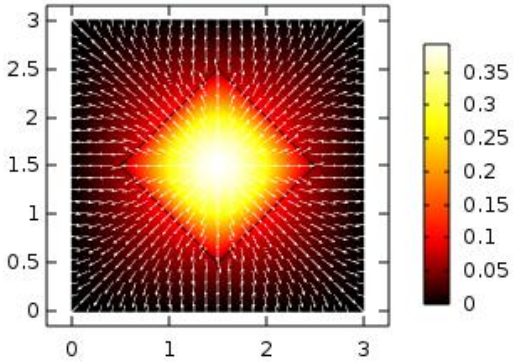
`Arrow Positioning` – **y grid points – Entry method, Points** = `Number of points,25` (*два поля*)

`Coloring and Style` – **Arrow length** = `Normalized`

`Coloring and Style` – **Scale factor** = `True, 0.08`

`Coloring and Style` – **Color** = `White`

LMB > Plot



Розв'язок 2. Відкрийте файл `Heat_in_Composite_Env_2.mph`. В ньому геометрія вже побудована. Зараз ми будемо в одній моделі використовувати вузли двох «фізик», тобто розв'язувати два рівняння теплопровідності, для кожного з яких використаємо власні матеріальні параметри. Для цього спочатку створіть вузол глобальних параметрів, і задайте в ньому параметри C, ρ, k, Q для першої і другої підобластей, додавши індекси до їх назв.

Global Definitions : RMB > Parameters

Заповніть таблицю параметрів так, як показано на рисунку.

Parameters

Parameters

Name	Expression	Value	Descripti
C1	0.1	0.1	
C2	0.1	0.1	
rho1	2	2	
rho2	1	1	
k1	10	10	
k2	2	2	
Q1	0	0	
Q2	4	4	

Генеруйте СкЕ *сітку* з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

В нашому шаблоні вже існує гілка **Heat Equation (hteq)**. В її секції вибору поки що стоятимуть індикатори 1 і 2 обох підобластей.

Heat Equation (hteq) - Heat Equation 1 :

Domain Selection–**Selection**=`All domains,1,2` (індикатор 2 тимчасово)

Diffusion Coefficient – **c** = `k1`

Source Term – **f** = `Q1`

Absorption Coefficient – $\mathbf{d_a}$ = `rho1*C1`

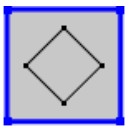
Задайте граничні умови.

`Physics` -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = `1,2,3,8`

Dirichlet Boundary Condition – **r** = `0`



В гілці **Heat Equation (hteq)** створіть вторинний вузол `Heat Equation`.

Heat Equation (hteq) : RMB > Classical PDEs - Heat Equation

Heat Equation (hteq) - Heat Equation 2 :

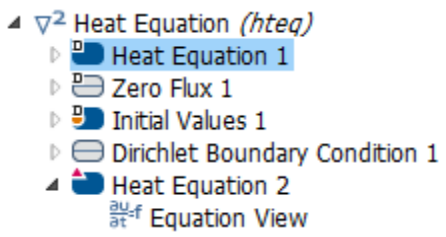
Domain Selection – **Selection** = `Manual,2`

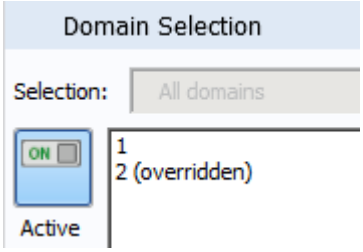
Diffusion Coefficient – **c** = `k2`

Source Term – **f** = `Q2`

Absorption Coefficient – $\mathbf{d_a}$ = `rho2*C2`

Гілка **Heat Equation (hteq)** набуде вигляду, який показано на наступному рисунку ліворуч. Якщо тепер подивитися на секцію вибору ПВ вузла **Heat Equation 1**, то побачимо, що використання підобласті з індикатором 2 анульовано (overridden) (наступний рисунок праворуч).





Це означає, що параметри, задані для рівняння теплопровідності в гілці **Heat Equation 1**, діятимуть лише в області з індикатором 1. А в області з індикатором 2 тепер будуть діяти параметри, які задані в ПВ вузла **Heat Equation 2**.

Налаштуйте розв'язувач.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times (s)** = `range(0,0.01,0.1)`

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Surface 1**, що містить графік температури *T*. Виконайте в ній налаштування, схожі для графіка в першому розв'язку, і додайте стрілковий графік, що зображатиме тепловий потік. Оскільки знадобляться дві формули обчислення теплового потоку $\boldsymbol{q}_1 = -k_1 \nabla T$, $\boldsymbol{q}_2 = -k_2 \nabla T$, то знадобляться два вузли типу `Arrow Surface`, і два джерела даних, обмежених відповідно на підобласті 1 і 2.

2D Plot Group 1 - Surface 1 :

`Coloring and style` – **Color Table** = `Thermal`

Створіть джерела даних.

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,1` (*два поля*)

Results – Data Sets - Study 1/Solution 1 (2) : RMB > Duplicate

Results – Data Sets - Study 1/Solution 1 (3) - Selection :

`Geometric Entity Selection` – **Selection** = `Manual,2` (*два поля*)

До графічної групи **1D Plot Group 1** додайте два вузли `Arrow Surface`.

2D Plot Group 1 : RMB > Arrow Surface

2D Plot Group 1 – Arrow Surface 1:

`Data` – **Data set** = `Study 1/Solution 1 (2)`

`Expression` – **x component, y component** = `-k1*Tx,-k1*Ty`

`Arrow Positioning` – **x grid points – Entry method, Points** = `Number of points,25` (*два поля*)

`Arrow Positioning` – **y grid points – Entry method, Points** = `Number of points,25` (*два поля*)

`Coloring and Style` – **Arrow length** = `Normalized`

`Coloring and Style` – **Scale factor** = `True, 0.08`

`Coloring and Style` – **Color** = `White`

2D Plot Group 1 – Arrow Surface 1: RMB > Duplicate
2D Plot Group 1 – Arrow Surface 2:

Data – **Data set** = Study 1/Solution 1 (3)

Expression – **x component, y component** = -k2*Tx,-k2*Ty

LMB > Plot

Буде побудовано такий же графік, як і в розв'язку 1.

3.18. Власні коливання круглої тремтівки.

При пошуку власних коливань фізичних систем в COMSOL Multiphysics застосовується Eigenfrequency режим дослідження, для якого початковим є рівняння Гельмгольца

$$\lambda^2 e_a u + \nabla \cdot (-c \nabla u) = f \,. \qquad (1)$$

Тут $\lambda = -i\omega$ позначає власне значення, i – уявну одиницю, ω– кутову частоту, яка пов'язана з частотою власних коливань ν формулою $\omega = 2\pi\nu$. В Eigenfrequency дослідженнях шуканими змінними є ν (eigenfrequency – власні частоти) та λ (eigenvalue – власні числа), які пов'язані співвідношенням $\lambda = -2\pi i \nu$. Шуканими є також власні форми (eigenmode) коливань.

В Eigenfrequency дослідженнях для позначення власної частоти ν використовується змінна freq, а для власного числа λ – lambda.

Приклад. Дослідимо власні коливання тремтівки в формі кола Ω одиничного радіуса, у якої зміщення точок границі дорівнюють нулю $u\big|_{\partial\Omega} = 0$. Потрібно знайти всі λ і відповідні їм розв'язки $u_\lambda(x,y)$ однорідного рівняння (1), для яких $u_\lambda(x,y) \neq 0$. Покладемо також $e_a = 1, c = 1$.

Виконайте послідовність команд:

File -> New -> Model Wizard -> 2D ->
Mathematics - Classical PDEs – Wave Equations (waeq)->
Custom Studies - Eigenfrequency

Отже тут вибрано Eigenfrequency тип дослідження.

Геометрія задачі складається з кола одиничного радіуса.

Geometry -> **Primitives** – Circle

Geometry 1 - Circle 1 : LMB > Build All Objects (*параметри кола за замовчуванням*)

На створеному колі *генеруйте* СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

Mesh Settings –**Element size** = Finer

LMB > Build All

Відкрийте ПВ вузла **Wave Equation (waeq) - Wave Equation 1**. В розділі **Equation** цієї панелі наведено рівняння (1). Для розв'язання задачі про власні коливання тремтівки потрібно ввести наступні коефіцієнти:

Wave Equation (waeq) - Wave Equation 1 :

Diffusion Coefficient – **c** = 1

Source Term – **f** = 0

Mass Coefficient – $e_a = 1$

Перевірте також, що в поточній панелі (**Wave Equation 1**) в розділі **Domain Selection** стоїть 1, тобто обрано коло.

Задайте граничні умови.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = All boundaries

Dirichlet Boundary Condition – **r = 0**

Відкрийте ПВ вузла **Step 1: Eigenfrequency**. В полі **Desired number of eigenfrequencies** введіть бажану для обчислення кількість власних частот.

Study 1 - Step 1: Eigenfrequency:

Study Settings – **Desired number of eigenfrequencies** = 20

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results - Data Sets** з'явиться **2D Plot Group 1** група графіків з вузлом **Surface 1**. Налаштуйте його, додавши вертикальні зміщення поверхні тремтівки.

2D Plot Group 1 - Surface 1 : RMB> Height Expression

Height Expression 1:

Axis – **Scale factor** = True ,0.2

LMB > Plot

Вибираючи в ПВ вузла **2D Plot Group 1** в списку **Data - Eigenfrequency** значення власної частоти, ви автоматично будете отримувати графік гармоніки, яка відповідає обраній частоті. Щоб графіки для різних частот використовували однакову палітру і розташовувалися в одній і тій же просторовій області, потрібно задати спільні для всіх графіків діапазони зміни кольорів і даних.

2D Plot Group 1 - Surface 1:

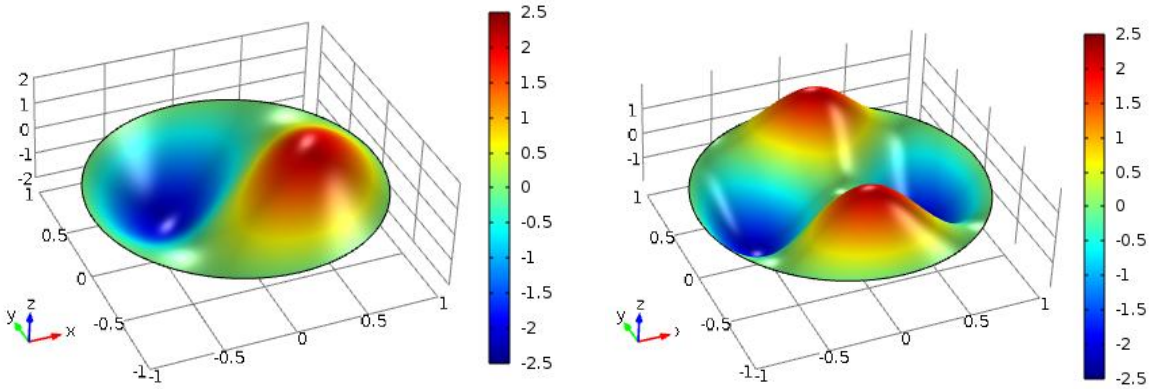
Range – **Manual color range** = True

Range – **Manual color range** – **Minimum, Maximum** = -2.5, 2.5

Range – **Manual data range** = True

Range – **Manual data range** – **Minimum, Maximum** = -2.5, 2.5

Форма тремтівки, яка відповідає другій власній частоті 0.60984 (1), зображена на наступному рисунку ліворуч. Фундаментальна форма коливань мембрани, що відповідає частоті 0.81736 (1) показана праворуч.



Зверніть увагу на те, що деякі власні частоти в списку **Data - Eigenfrequency** повторюються двічі (в дужках стоять індекси 1 або 2). Кожній такій частоті відповідає своя фундаментальна форма, і форми в парі відрізняються тільки поворотом. Деякі частоти пари не мають.

В багатьох підручниках наводяться графіки вузлових ліній основних гармонік круглої тремтівки. Побудуйте їх.

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

`2D Plot Group 2->` Add Plot - Contour

2D Plot Group 2 - Contour 1:

`Levels` –**Entry method** = `Levels`

`Levels` – **Levels** = `0`

Цим вказується необхідність будувати тільки нульові лінії рівня виразу, зазначеного в полі **`Expression`** – **Expression** (ми залишили там `u`). У цьому ж вікні введіть:

`Coloring and style` – **Color type** = `Filled`

`Coloring and style` – **Coloring** = `Color Table`

`Coloring and style` – **Color Table** = `GrayScale`

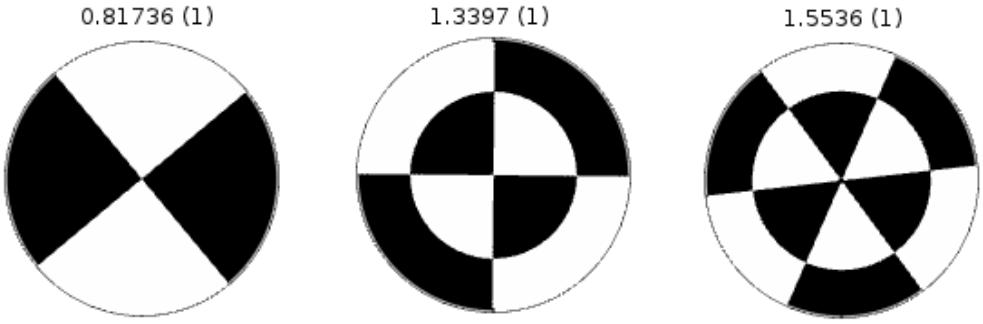
Ці налаштування дозволяють створити монохромне зображення основних гармонік. Темні області будуть відповідати від'ємним зміщенням точок тремтівки, а білі - додатним.

2D Plot Group 2 :

`Data` –**Eigenfrequency** = `0.81736(1)`

LMB > Plot

На наступному рисунку показані зони позитивних і негативних зміщень точок для гармонік, частоти яких вказані зверху.



Зауваження. Останні графіки можна побудувати в інший спосіб, задавши у вузлі **2D Plot Group 1 - Surface 1** вираз `u>0`. Але якість такого зображення гірша, бо вона залежить від розмірів скінченних елементів.

Вправа 1. Побудуйте форми власних коливань прямокутної тремтівки.

Вправа 2. З курсу математичної фізики відомо, що аналітичний розв'язок одновимірного рівняння теплопровідності $u'_t = a^2 u''_{xx}$ з нульовими граничними умовами на відрізку $[0, L]$ має вигляд $u(x,t) = \sum_{n=1}^{\infty} \beta_n u_n(x,t)$, де $u_n(x,t)$ – власні

функції $u_n(x,t)=\sin\left(\frac{\pi n x}{L}\right)e^{-\frac{\pi n}{L}at}$, а β_n – деякі числові коефіцієнти, які визначаються з початкових умов. Побудуйте в COMSOL Multiphysics власні функції задачі теплопровідності на відрізку $[0,1]$ і порівняйте їх з аналітичними виразами для $u_n(x,t)$.

Вправа 3. Побудуйте власні функції задачі теплопровідності в прямокутнику.

3.19. Розповсюдження тепла в радіоактивному стрижні.

Тривимірні задачі потребують значно більше комп'ютерних ресурсів ніж двовимірні. Тому, якщо існує можливість перетворити тривимірну задачу на двовимірну, то її треба використати. Зокрема це можна зробити для вісесиметричних задач, коли геометрія, матеріальні властивості і зовнішні впливи не залежать від кута. Як приклад, розглянемо задачу розповсюдження тепла в стрижні, в якому виділяється тепло внаслідок радіоактивного розпаду.

Розглянемо вертикальний циліндричний стрижень радіусом 1, і 3 одиниці довжиною. На нижньому торці задано тепловий потік. Верхівка стриженя підтримується при сталій температурі. На боковій границі відбувається теплообмін з навколишнім середовищем. Одночасно тепло рівномірно виділяється у всьому об'ємі завдяки радіоактивному розпаду. Початкова температура стрижня відома. Це призводить до неоднорідного рівняння теплопровідності [1]

$$\rho C\frac{\partial T}{\partial t}-\nabla\cdot(k\nabla T)=Q, \tag{1}$$

де ρ - густина матеріалу, C – питома теплоємність, k – коефіцієнт теплопровідності, Q – потужність внутрішніх джерел тепла.

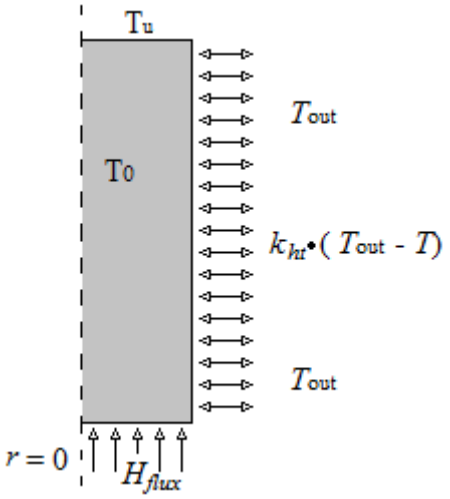
Оскільки задача вісесиметрична, то можна записати її рівняння в циліндричних координатах r,φ,z. Через незалежність моделі від кута φ воно набуває вигляду

$$r\rho C\frac{\partial T}{\partial t}-\frac{\partial}{\partial r}\left(k r\frac{\partial T}{\partial r}\right)-\frac{\partial}{\partial z}\left(k r\frac{\partial T}{\partial z}\right)=Q\cdot r. \tag{2}$$

Нехай густина матеріалу стриженя $\rho=7800\,кг/м^3$, питома теплоємність $C=500\,вт\cdot сек/кг\cdot град$, коефіцієнт теплопровідності $k=40\,вт/м\cdot град$, потужність внутрішніх джерел тепла $Q=20000\,вт/м^3$, початкова температура $T_0=0\,^{\circ}C$, температура зовнішнього середовища $T_{out}=100\,^{\circ}C$, температура верхнього торця $T_u=100\,^{\circ}C$, коефіцієнт теплопередачі $k_{ht}=50\,вт/м^2\cdot град$, тепловий потік $H_{flux}=5000\,вт/м^2$, радіус циліндра $R=1\,м$, а висота $Hgt=3\,м$.

Граничні умови матимуть вигляд:

- на нижньому торці стрижня задано тепловий потік $\boldsymbol{n}\cdot(k\nabla T)=H_{flux}$; оскільки в граничній умові Неймана використовується параметр *c*, який в циліндричній системі координат (див. рівняння (2)) має значення $c=k\cdot r$, то умова повинна набути вигляду $\boldsymbol{n}\cdot(c\nabla T)=H_{flux}\cdot r$;
- на боковій поверхні здійснюється теплообмін з навколишнім середовищем по закону $\boldsymbol{n}\cdot(k\nabla T)=k_{ht}\cdot(T_{out}-T)$, який повинен бути представлений у вигляді $\boldsymbol{n}\cdot(c\nabla T)=g-q\cdot T$; маючи на увазі, що $c=k\cdot r$, потрібно покласти $g=k_{ht}\cdot r\cdot T_{out}$, $q=k_{ht}\cdot r$;
- $T=T_u$ на верхньому торці стрижня;
- на осі циліндра ($r=0$) граничну умову задавати непотрібно (насправді вона матиме штучний вигляд $\boldsymbol{n}\cdot(k\nabla T)=0$).



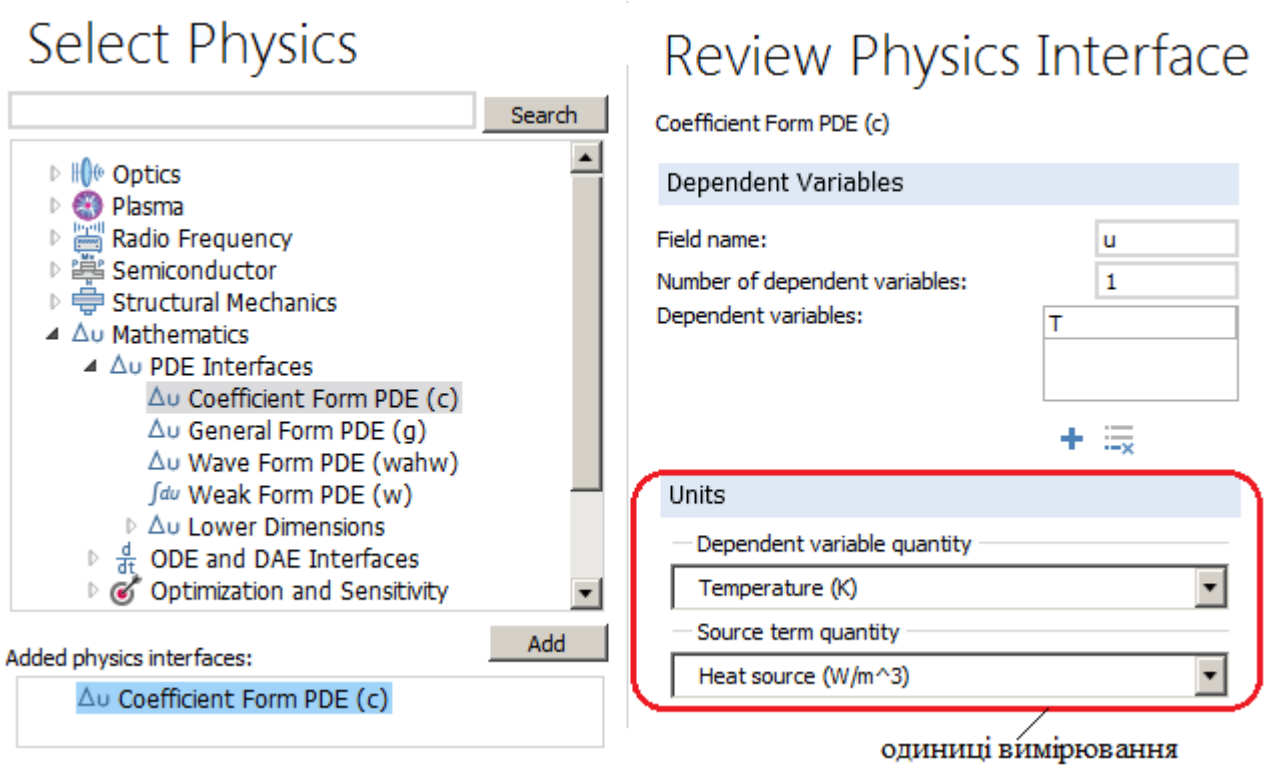
В цій моделі прослідкуємо також за фізичними одиницями вимірювання.

Почніть створення шаблону моделі з вибору `Coefficient From PDE(c)` інтерфейсу.

File->New->Model Wizard->2D Axisymmetric->
Mathematics – PDE Interfaces – Coefficient From PDE(c)

Зауваження. При створенні `2D Axisymmetric` моделі у вікні `Select Physics` майстер Model Wizard не відображає гілку `Classical PDEs`. Тому доводиться вибирати `Coefficient From PDE(c)` інтерфейс.

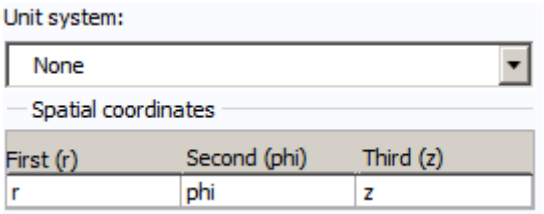
В панелі `Review Physics Interface` в полі **Dependent Variables** змініть ім'я шуканої величини на *T*, а в розділі **`Units`** в випадаючих списках оберіть розмірності для температури та потужності джерел тепла так, як зображено на рисунку.



Друга розмірність (`Source term quantity`) буде використовуватися при перевірці фізичних одиниць в рівнянні вузла `Coefficient Form PDE`.

Оберіть Time Dependent тип дослідження.

В ПВ вузла **Component 1 (comp 1)** в таблиці розділу **`Spatial coordinates`** ви можете дізнатися (або поміняти) імена просторових змінних, які будуть використовуватися в моделі. В нашому прикладі залишимо назви циліндричних координат без змін.



Створіть таблицю параметрів

Global Definitions : RMB > Parameters

і заповніть її так, як зображено на рисунку.

Parameters

Name	Expression	Value	Description
rho	7800[kg/m^3]	7800 kg/m³	густина
C	500[W*s/(kg*degC)]	500 J/(kg·K)	теплоємність
k	40[W/(m*degC)]	40 W/(m·K)	коефіцієнт теплопровідності
Q	20000[W/m^3]	20000 W/m³	потужність тепловиділення
Tout	100[degC]	373.15 K	температура зовнішнього середовища
Tu	100[degC]	373.15 K	темпертура верхнього торця
Kht	50[W/m^2/degC]	50 W/(m²·K)	коефіцієнт теплопередачі
Hflux	5000[W/m^2]	5000 W/m²	тепловий потік
T0	0[degC]	273.15 K	початкова температура
R	1[m]	1 m	радіус стрижня
Hgt	3[m]	3 m	довжина стрижня

Геометрію моделі побудуйте у вигляді прямокутника, обертанням якого утворюється циліндричне тіло стрижня.

Geometry -> **Primitives** – Rectangle

Component 1 - Geometry 1 - Rectangle1:

Size – **Width, Height** = R, Hgt (*два поля*)

Position – **r, z** = 0, – Hgt/2 (*два поля*)

LMB > Build All Objects

Генеруйте СкЕ *сітку*.

Mesh 1: Mesh Settings –**Element size** = Finer

LMB > Build All

Задайте рівняння. В ПВ вузла **Coefficient Form PDE (c) – Coefficient From PDE 1** в розділі **Equation** наведено загальний вигляд коефіцієнтної форми ДРЧП.

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \left(-c \nabla u - \alpha u + \gamma\right) + \beta \cdot \nabla u + \mathrm{a} u = f, \nabla = \left[\frac{\partial}{\partial r}, \frac{\partial}{\partial z}\right].$$

При перевірці фізичних одиниць система вважатиме, що розмірність лівої і правої частини цього рівняння повинна бути такою, яка обрана в панелі Review Physics Interface в розділі **Units** в випадаючому списку **Source term quantity**, тобто мати розмірність $вт/м^3$. Але в рівнянні (2), яке буде використовуватися, права частина $Q \cdot r$ має розмірність $вт/м^2$. Щоб система не відображали формули помаранчевим кольором (це відбувається, коли фізичні одиниці невідповідні) до множника r будемо додавати «множник» розмірності [1/m]. Наприклад, замість виразу $Q \cdot r$ будемо писати $Q \cdot r[1/m]$. Усі доданки лівої частини рівняння повинні мати однакову розмірність з правою частиною. Отже вони теж потребують коректування одиниць вимірювання.

Задайте коефіцієнти рівняння, враховуючи зазначену особливість.

Coefficient Form PDE (c) - CoefficientFromPDE 1:

Diffusion Coefficient – **c** = k*r[1/m]

Absorption Coefficient – **a** = 0

Source Term – **f** = Q*r[1/m]

Mass Coefficient – $\mathbf{e_a}$ = 0

Damping or Mass Coefficient – $\mathbf{d_a}$ = rho*C*r[1/m]

Коефіцієнти α, β і γ залиште незмінними, тобто нульовими. Цим задається рівняння (2) з поправкою на розмірність.

Задайте граничні умови. У вузлах Flux/Source система вважатиме, що ліва і права частина умови повинні мати розмірність теплового потоку $H_{flux}[вт/м^2]$. Поява множника r в виразі $\boldsymbol{n} \cdot (c \nabla T) = H_{flux} \cdot r$, подібно до попереднього, потребує використання «множника» розмірності [1/m].

На нижньому торці задається тепловий потік.

Physics -> **Boundaries** – Flux/Source

Helmholtz Equation (hzeq) - Flux/Source 1:

Boundary Selection – **Selection** = Manual,2 (*нижня межа*)

Boundary Flux/Source – **g** = Hflux*r[1/m]

Boundary Absorption/Impedance Term – **q** = 0

На боковій поверхні задано теплообмін з навколишнім середовищем.

Physics -> **Boundaries** – Flux/Source

Helmholtz Equation (hzeq) - Flux/Source 2:

Boundary Selection – **Selection** = Manual,4 (*бокова сторона*)

Boundary Flux/Source – **g** = Kht*Tout*r[1/m]

Boundary Absorption/Impedance Term – **q** = Kht*r[1/m]

На верхньому торці задано температуру.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = Manual,3 (*верхня межа*)

Dirichlet Boundary Condition – **r** = Tu

Задання граничної умови на лівому боці прямокутника (це вісь обертання), непотрібно.

Задайте початкові умови.

Initial Values 1:

Initial Values – **Initial Value for u:**– **u** = T0

Налаштуйте розв'язувач.

Study 1 - Step 1: Time Dependent:

Study Settings –**Times (s)** = range(0,250,25000)

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Surface 1**, що містить графік температури *T* в останній зі списку **Times (s)** момент часу (t=25000 сек). Ввімкніть на графіку відображення екстремальних значень і змініть палітру кольорів на термічну, загальновживану для теплових задач.

2D Plot Group 1 :

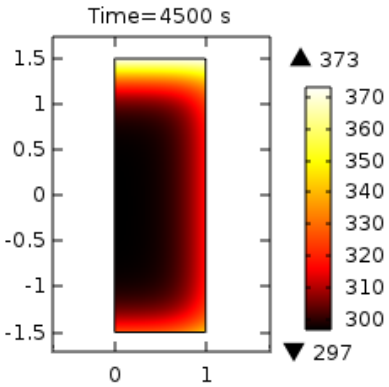
Data – **Time (s)** = 4500

Color Legend - **Show maximum and minimum values** = True

2D Plot Group 1 - Surface 1 :

Coloring and style – **Color Table** = Thermal

LMB > Plot



Пам’ятайте, що змінення часу в полі **Time (s)** вузла **2D Plot Group 1**, змінює мінімальне і максимальне значення температури, що призводить до зміни діапазону шкали кольорів. Тобто шкала кольорів може змінюватися від обраного моменту часу. Відвернути це дозволяє ручне налаштування діапазону кольорів. Виконайте це самостійно.

Після обчислень в ДМ з’явиться тривимірна графічна група, з графіком температури по розтину циліндра. Для неї у вузлі **Data Sets** автоматично було створено джерело даних **Revolution 2D 1**. Змініть на графіку палітру кольорів і ввімкніть відображення екстремальних значень.

3D Plot Group 2 :

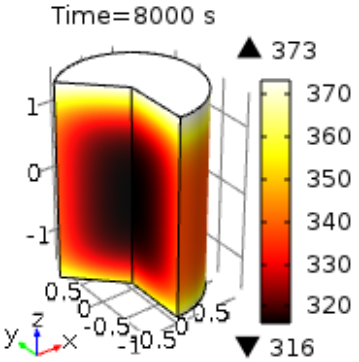
`Data` – **Time (s)** = `8000`

`Color Legend` - **Show maximum and minimum values** = `True`

3D Plot Group 2 - Surface 1 :

`Coloring and style` – **Color Table** = `Thermal`

LMB > Plot

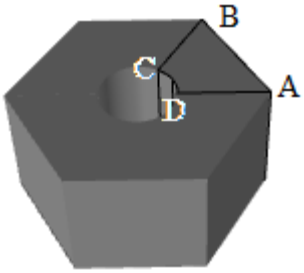


Корисно буде побудувати анімацію розв’язку. Зробіть це самостійно. Ви зможете «побачити потоки тепла» крізь торці стрижня всередину і назовні в залежності від того, температура $T < 100\,^{\circ}C$ чи $T > 100\,^{\circ}C$.

3.20. Теплопровідність в шестикутній призмі з отвором.

Розглянемо задачу теплопровідності в тривимірному тілі з симетрією. Коли кажуть про симетрію, то мають на увазі, що існує площина, відносно якої модель ідентична. Площин симетрії може бути декілька. Для врахування симетрії потрібно, щоб геометрія і матеріальні властивості тіла були симетричні. Необхідно також, щоб граничні умови, зовнішні впливи і внутрішні джерела були прикладені симетрично. Для скалярного поля густина потоку (струму, тепла, тощо) на площині симетрії дорівнює нулю. При виконанні всіх умов симетрії розрахункову область моделі можна зменшити до частини повної області. В цій частині постановка задачі така ж сама, як у повній геометрії, з додаванням граничної умови «нульового потоку» на площинах симетрії.

Приклад. Розглянемо задачу про усталений розподіл температури в шестикутній призмі з круговим отвором, центр якого розташований в початку координат.



Нехай джерела тепла відсутні і початкова температура дорівнює нулю. На поверхні кругового отвору підтримується стала температура $T=T_{in}$. На нижньому торці задан вісесиметричний тепловий потік $\boldsymbol{n}\cdot(c\nabla T)=p\sqrt{x^2+y^2}$. На верхньому торці і зовнішній поверхні виконується теплообмін з навколишнім середовищем по закону $\boldsymbol{n}\cdot(c\nabla T)=q\cdot(T_{out}-T)$. Покладемо p=50, q=0.1, T_{in}=1, T_{out}=1, сторона шестикутника a=1, радіус отвору R=0.3·a, висота призми Hght=1, коефіцієнт теплопровідності c=1.

Задача характеризується наявністю трьох площин симетрії і для заощадження комп'ютерних ресурсів розрахункову область можна побудувати як одну шосту частину повної геометрії, складену з призми, верхній торець якої на попередньому рисунку має вигляд криволінійної трапеції ABCD.

За допомогою майстра створіть тривимірну модель задачі теплопровідності.

File->New -> Model Wizard -> 3D -> Mathematics – Classical PDEs – Heat Equation (hteq) + Variables = T -> Stationary

Цим створено шаблон моделі з ім'ям шуканої змінної *T* і фізичним змістом температури.

Оскільки значення фізичних параметрів умовні, то немає сенсу використовувати їх розмірності.

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

Задайте параметри моделі.

Global Definitions : RMB > Parameters

Заповніть таблицю параметрів так, як показано на рисунку.

Parameters

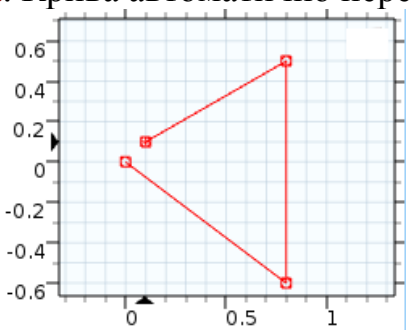
Name	Expression	Value	Description
a	1	1	сторона шестикутника
R	0.3*a	0.3	радіус отвору
Hght	1	1	висота призми
x1	a*sqrt(3)/2	0.86603	абсциса вершини
Tout	1	1	зовнішня температура
Tin	1	1	внутрішня температура
c	1	1	коефіцієнт теплопровідності
p	50	50	
q	0.1	0.1	

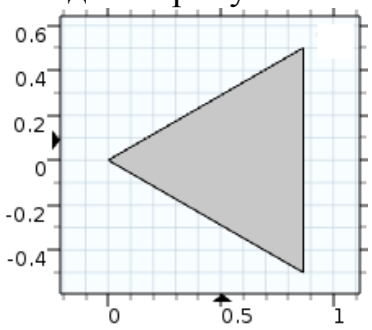
Геометрія моделі складається з шестикутної призми з круговим отвором.

Створіть робочу площину. При активному вузлі **Geometry 1** виконайте команду

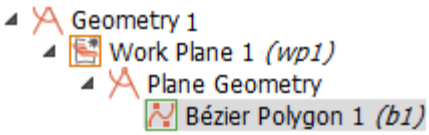
Geometry->**Work Plane**->Work Plane

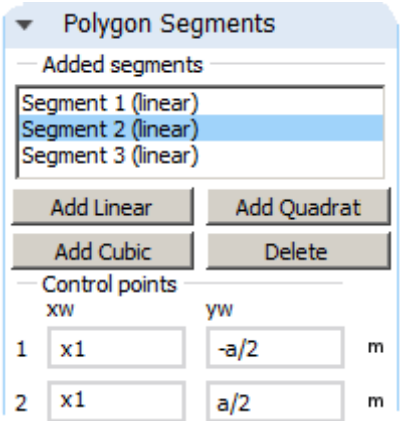
В ПВ створеного вузла **Work Plane 1** всі налаштування залиште без змін. Вони відповідають робочій площині, що співпадає з площиною XY. Клацніть по кнопці Show Work Plane вгорі панелі. Цим ви активуєте вузол **Work Plane 1 – Plane Geometry**. На закладці `Work Plane` інструментальної панелі перевірте, що натиснуті кнопки Snap і Solid. Кнопка Snap вмикає режим «притягання» покажчика миші до вершин точкової решітки (grid) графічної області. В режимі «Solid» всі замкнені криві перетворюються на «солідні» об'єкти (тобто області). На панелі `Geometry` в секції **`Draw`** натисніть кнопку Line, переведіть покажчик миші в графічну область і накресліть трикутник приблизно відповідний, показаному на наступному рисунку праворуч. Закінчіть креслення ламаної (коли вона буде вже замкненою) клацанням правою кнопкою миші в зоні малюнка. Крива автоматично перетвориться на солідний трикутник.





Відредагуйте координати вершин трикутника так, щоб вони дорівнювали (`0,0`), (`x1,-a/2`), (`x1, a/2`). Для цього в розділі **`Polygon Segments`** послідовно виділяйте сегменти ламаної і редагуйте значення координат вершин в полях **Control points**.





Додайте на робочу площину коло.

`Geometry` -> **`Primitives`** – Circle

Component 1 - Geometry 1 - Circle 1:

`Size and Shape` – **Radius:** = `R`

За допомогою операції логічного віднімання видаліть з трикутника область кола.

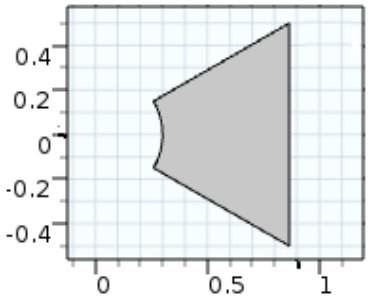
`Geometry` **-> `Booleans and Partitions`** – Difference

Component 1 - Geometry 1 - Difference 1:

`Difference` – **Object to add** = `b1` (*трикутник*)

`Difference` – **Object to subtract** = `c1` (*коло*)

LMB > Build Selected



Посуньте отриману двовимірну фігуру вгору, використовуючи операцію Extrude. Вона витягує солідні об'єкти робочої площини в простір, утворюючи тривимірні тіла. Зміщення профілю виконується в напрямку нормальному до площини. Звичайно, що слід вказати відстань, на яку витягується профіль.

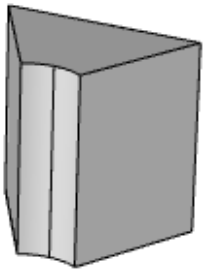
Work Plane 1 (wp1): `Geometry` ->**`Operations`**-> Extrude

Geometry 1 – Extrude 1 (ext1):

`General` – **Input objects** = `wp1` (*робоча площина*)

`Distance from Plane` – **Distance** = `Hght` (*відстань витягування*)

LMB > Build All Objects



Генеруйте СкЕ *сітку* з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB > Build All

Задайте рівняння. Перевірте, що в ПВ гілки **Heat Equation (hteq)** в розділі **`Domain Selection`** в секції вибору стоїть індикатор 1, тобто обрана створена криволінійна призма. В дереві моделі виділіть вузол **Heat Equation 1**. В його панелі в розділі **`Equation`** наведено загальний вигляд рівняння теплопровідності

$$d_a \frac{\partial T}{\partial t} + \nabla \cdot (-c \nabla T) = f\,, \quad \nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right],$$

де T позначає шукану температуру. Коефіцієнти рівняння задаються в полях в нижній частині панелі.

Heat Equation (hteq) - Heat Equation 1 :

`Diffusion Coefficient` – **c** = `c`

`Source Term` – **f** = `0`

`Absorption Coefficient` – $\mathbf{d_a}$ = `0`

Нульові початкові умови задаються за замовчуванням, і їх міняти непотрібно.

Задайте граничні умови. На нижньому торці задайте тепловий потік за допомогою вузла типу `Flux/Source`. В ньому задається умова $-\boldsymbol{n} \cdot (-c \nabla T) = g - qT$, і щоб вона набула бажаного вигляду $\boldsymbol{n} \cdot (c \nabla T) = g(x, y)$ потрібно ввести $g = p\sqrt{x^2 + y^2}$, $q = 0$.

Physics -> **Boundaries** – Flux/Source

Helmholtz Equation (hzeq) - Flux/Source 1:

Boundary Selection – **Selection** = Manual,3 (*нижній торець*)

Boundary Flux/Source – **g** = p*sqrt(x^2+y^2)

Boundary Absorption/Impedance Term – **q** = 0

На верхньому торці і зовнішній поверхні виконується теплообмін з навколишнім середовищем по закону $\boldsymbol{n}\cdot(c\nabla T)=q(T_{out}-T)$.

Physics -> **Boundaries** – Flux/Source

Helmholtz Equation (hzeq) - Flux/Source 2:

Boundary Selection – **Selection,** <> = Manual,4,7

Boundary Flux/Source – **g** = q*Tout

Boundary Absorption/Impedance Term – **q** = q

На межі внутрішнього отвору підтримується фіксована температура.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection ,** <> = Manual,1,5

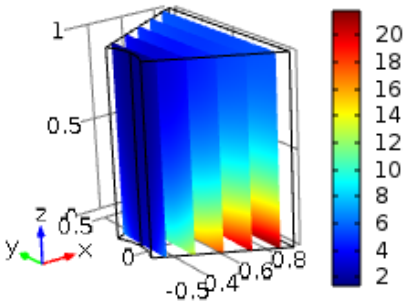
Dirichlet Boundary Condition – **r** =Tin

На бокових гранях призми залишимо умови, що задані за замовчуванням у вузлі **Zero Flux 1**. Нульовий тепловий потік на гранях, це і є потрібні граничні умови симетрії на цих площинах.

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Slice 1**, що містить пошаровий графік температури $T(x,y,z)$.



Більш інформативним буде графік у повному тілі. Для цього побудуйте поверхневий графік температури для всієї призми. Але спершу треба створити відповідне джерело даних Sector 2D. Воно призначено для поширення розв'язку з деякої зони на область, яка отримується шляхом багаторазового повороту «базової» розрахункової області.

Results -> Data Set - More Data Sets -> Sector 3D

Data Sets - Sector 3D 1 :

Data – **Data set** = Study 1/Solution 1

Axis Data –**Axis entry method** = Two points

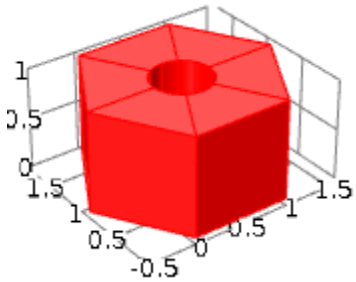
Axis Data – Point 1 – **x, y, z** = 0,0,0 (*три поля*)

Axis Data – Point 2 – **x, y, z** = 0,0,1 (*три поля*)

Цим задається вісь, що проходить через точки (0,0,0) і (0, 0,1), і навколо якої буде виконуватися поворот при створені «кругового» масиву джерел даних.

Symmetry – Number of sectors = 6 (*кількість елементів в масиві ДД*)

Виконайте команду Plot і отримайте зображення джерела даних.



Тепер побудуйте «поверхневий» графік температури.

Results -> Plot Group - 3D Plot Group

3D Plot Group 2 : Data – Data set = Sector 3D 1

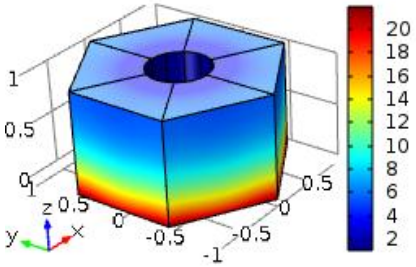
3D Plot Group 2 : RMB > Surface

3D Plot Group 2 - Surface 1 :

Data – Data set = From parent

Expression – Expression = T

Виконайте команду Plot, і отримайте графічне зображення розв'язку у вигляді функціонального забарвлення поверхні шестикутної призми з отвором.



Побудуйте графік температури по похилому перерізу призми. Почніть зі створення джерела даних Cut Plane, що містить дані цього перерізу.

Results -> **Data Set** - Cut Plane

Results – Data Sets – Cut Plane 1:

Data – Data set = Sector 3D 1

Plane Data – Plane type = General (будувати переріз по трьом точкам)

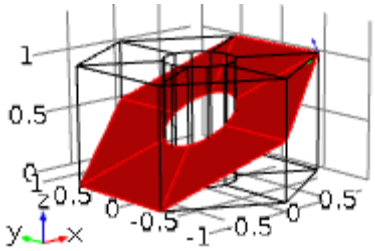
Задайте координати трьох точок площини перерізу.

Plane Data – Point 1 – x, y, z = x1,-a/2, Hght (*три поля*)

Plane Data – Point 2 – x, y, z = x1, a/2, Hght (*три поля*)

Plane Data – Point 3 – x, y, z = -x1, 0, 0 (*три поля*)

LMB > Plot



Використовуючи створене ДД, побудуйте графік температури на перерізі.

Results -> Plot Group - 3D Plot Group

3D Plot Group 3 : Data – Data set = Cut Plane 1

3D Plot Group 3 : RMB > Surface

3D Plot Group 3 - Surface 1 :

Data – Data set = From parent

Expression – Expression = T

3D Plot Group 3 : RMB > Line

3D Plot Group 3 - Line 1 :

Data – Data set = Sector 3D 1

Expression – Expression = 1

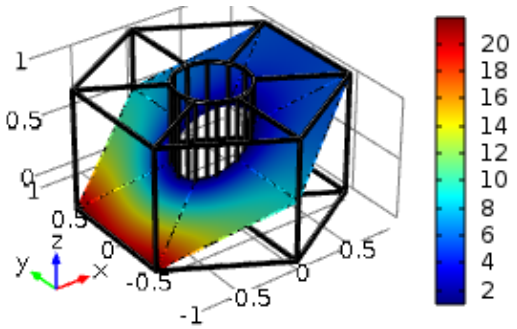
Coloring and style – Line type = Tube

Coloring and style – Tube radius expression = 0.02

Coloring and style – Radius scale factor = True, 1 (*два поля*)

Coloring and style – Coloring, Color = Uniform, Black (*два поля*)

LMB > Plot



3.21. Дві крайові задачі коливання квадратної мембрани.

В цьому параграфі ми продемонструємо використання однієї розрахункової області для розв'язання двох схожих, але різних задач.

Приклад. Дослідимо коливання квадратної тремтівки при двох відмінних початкових умовах. В першій задачі тремтівка буде закріплена по контуру, а початкова форма матиме вигляд чотирибічної піраміди.

За допомогою майстра створіть двовимірну модель задачі.

File -> New -> Model Wizard -> 2D -> Mathematics – Classical PDEs – Wave Equation (waeq) -> Time Dependent

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Накресліть квадрат з центром в початку координат і стороною 2 одиниці $(-1 \le x, y \le 1)$.

Geometry -> **Primitives** – Square

Geometry 1 - Square 1 (sq1):

Size – Size length = 2

Position – Base = Center

Генеруйте СкЕ *сітку*.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** = Extra fine

LMB> Build All

Перевірте, що в ПВ гілки **Wave Equation (waeq)** в розділі **Domain Selection** в секції вибору стоїть індикатор області квадрата 1. В дереві моделі виділіть вузол **Wave Equation 1**. В його панелі в розділі **Equation** наведено загальний вигляд хвильового рівняння.

$$e_a \frac{\partial^2 u}{\partial t^2} + \nabla \cdot (-c \nabla u) = f, \quad \nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right],$$

де u позначає шукану функцію ($u(t, x, y)$- вертикальне зміщення в момент часу t точки тремтівки з координатами (x, y)). Коефіцієнти рівняння задаються в полях в нижній частині цієї панелі.

Wave Equation (waeq) - Wave Equation 1 :

Diffusion Coefficient – **c** = 1

Source Term – **f** = 0

Mass Coefficient – $\mathbf{e_a}$ = 1

Початкову форму тремтівки оберіть у вигляді чотирикутної «пірамідки». Щоб її задати створіть відповідну функцію.

Home -> **Definitions** – Functions -> Global - Analytic

Function name = u0

Definition–**Expression**= (2-abs (x) -abs (y) -abs (abs (y) -abs (x))) /2

Definition – **Arguments** = x, y

Для перевірки виконайте побудову графіка цієї функції в квадраті $[-1,1] \times [-1,1]$.

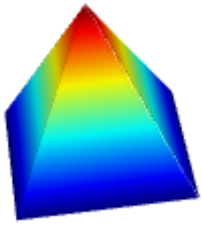
Plot Parameters – **Argument, Low limit, Upper limit** = x, -1, 1

Plot Parameters – **Argument, Low limit, Upper limit** = y, -1, 1

Plot Parameters

Argument	Lower limit	Upper limit
x	-1	1
y	-1	1

LMB > Plot



Використовуючи створену функцію, задайте початкові умови (початкову форму і нульову швидкість). Перевірте, що в ПВ вузла **Initial Values 1** в секції вибору **Selection** обрано індикатори квадрата 1 і введіть

Initial Values 1:

Initial Values – **Initial Value for u:**– **u** = u0 (x, y)

Initial Values – Initial time derivative of u: $-\frac{\partial u}{\partial t} = 0$

В нашому прикладі тремтівка закріплена на контурі.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** = All boundaries

Dirichlet Boundary Condition – **r** = 0 (*умова закріплення*)

Виконайте налаштування розв'язувача, задавши часовий інтервал і похибки обчислень.

Study 1 - Step 1: Time Dependent:

Study Settings –**Times** = range(0,0.1,2.7)

Study Settings –**Relative tolerance** = True, 0.0001

Study 1: RMB > Show Default Solver

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

Absolute Tolerance – **Tolerance** = 0.0001

Зауваження. Досі ми не змінювали налаштування похибок обчислень, бо значень за замовчуванням було достатньо. В цій задачі, щоб отримати прийнятний результат, потрібно зменшити похибки і використати досить дрібну СкЕ сітку.

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **2D Plot Group 1** з вузлом **Surface1**, що містить графіки розв'язку $u(t,x,y)$ в момент часу, заданий в батьківському вузлі в полі **Data** – **Time**.

Змініть у вузлі **Surface1** палітру, фіксуйте для всіх моментів часу однаковий діапазон кольорів, і додайте підвузол Height Expression.

2D Plot Group 1 – Surface 1:

Range – **Manual color range** = True

Range – **Manual color range** – **Minimum, Maximum** = -1, 1

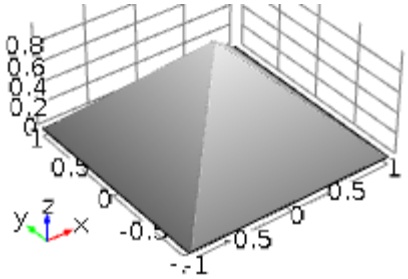
Coloring and style –**Color table** = GrayScale

2D Plot Group 1 – Surface 1: RMB > Height Expression

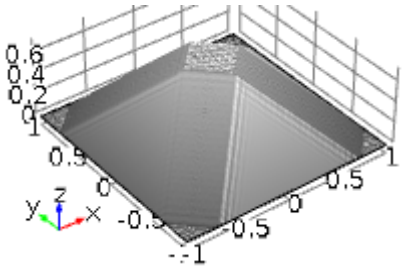
Height Expression 1: Axis – **Scale factor** = True ,1 (*два поля*)

LMB > Plot

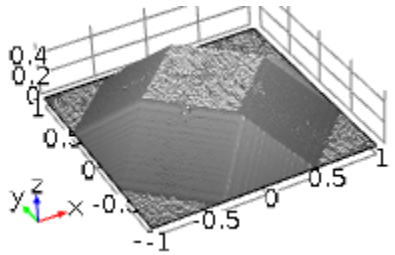
Змінюючи час в полі **Data** – **Time** вузла **2D Plot Group 1**, ви будуватимете форму тремтівки для бажаного моменту. На наступних рисунках зображена тремтівка в моменти часу t=0,0.2,0.4,0.6.



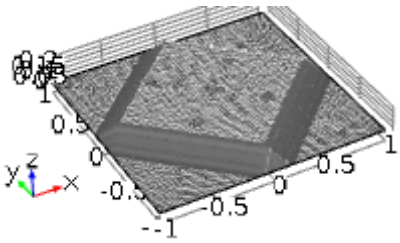
t=0



t=0.2



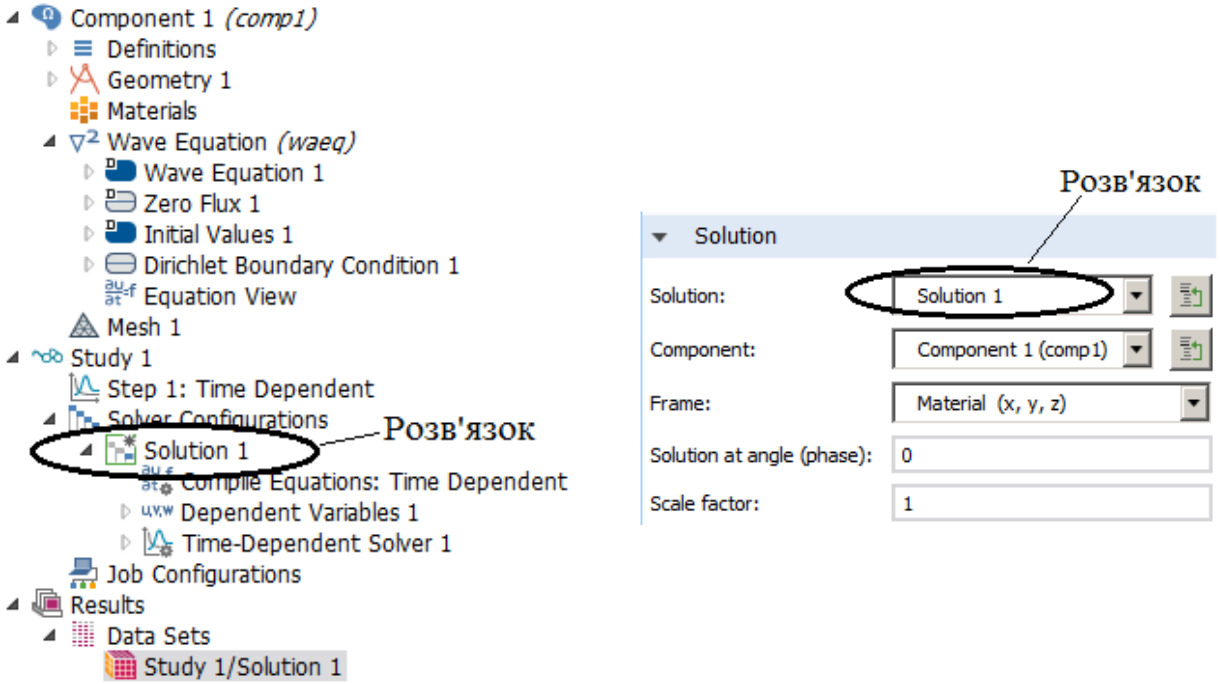
t=0.4



t=0.6

Графічну групу **1D Plot Group 1** можна використати для побудови анімації (зробіть це самостійно).

Зауваження. Зараз в нашій моделі існує один вузол **Study 1** і одне джерело даних **Study 1/Solution 1**, яке «вибирає» з вузла **Study 1** розв'язок **Solution 1**.



Збережіть модель в файл двічі, наприклад, з іменами `VibrSquare_sol1.mph` і `VibrSquare_sol2.mph`. Це потрібно щоб продовжити роботу далі з поточного місця двома різними способами.

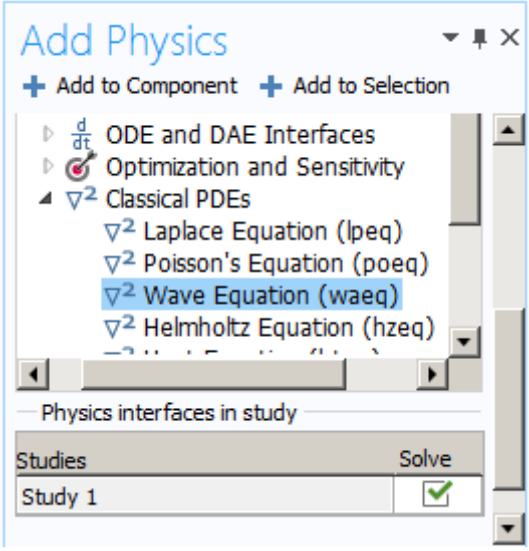
Наступною задачею є обчислення коливань тремтівки при іншому початковому профілі і інших граничних умовах. Звичайно, можна внести зміни в попередню модель і переобчислити її. Інший спосіб полягає в створені додаткових вузлів `Initial Values` і `Dirichlet Boundary Condition`, які анулюють умови перших вузлів **Initial Values 1** і **Dirichlet Boundary Condition 1**. Але якщо треба одночасно мати обидва розв'язки, наприклад, щоб побудувати графік їх різниці, то існують інші способи. Один із них полягає в використанні другої «фізики». В компоненті існує одна розрахункова область, але вузлів «фізик» може бути багато. Ми збираємося продемонструвати використання цього способу.

Інший спосіб полягає в створенні другого вузла `Study`, в ПВ якого можна визначити, які з вузлів `Initial Values` і `Dirichlet Boundary Condition` використовувати при обчисленні (якщо вони частково або повністю анулюють одне одного).

Перший спосіб. Відкрийте файл VibrSquare_sol1.mph. Команда Duplicate для вузлів фізик не передбачена. Тому «вручну» створіть гілку нової фізики, в якій теж буде розв'язуватися хвильове рівняння.

Home **-> Physics**-> Add Physics

В панелі Add Physics, яка з'явиться збоку, виберіть Mathematics – Classical PDEs – Wave Equation (waeq), і закрийте панель.



Вибір у вікні Add Physics виконується в стандартний спосіб подвійним клацанням миші по рядку Wave Equation (waeq) або однократним клацанням по кнопці +Add to Component вгорі вікна. Потім вікно Add Physics закривається хрестиком (X) в правому верхньому куті.

Таку послідовність дій ми кратко записуємо наступним чином.

Component 1: RMB > Add Physics = Add Physics ->
Mathematics – Classical PDEs – Wave Equation (waeq) ->X

В дереві моделі з'явиться гілка **Component 1 - Wave Equation 2 (waeq)**, в якій буде використовуватися та ж сама геометрія. Але в її вузлах можна налаштувати розв'язання іншої задачі. Зверніть увагу, що в цьому вузлі в розділі **Dependent Variables** ім'я нової шуканої змінної позначено як u2.

Повторіть інструкції створення крайової задачі.

Задайте коефіцієнти рівняння (вони можуть відрізнятися від коефіцієнтів першої задачі).

Wave Equation 2 (waeq) - Wave Equation 1 :

Diffusion Coefficient – **c** = 1

Source Term – **f** = 0

Mass Coefficient – $\mathbf{e_a}$ = 1

Задайте початкову форму і початкову швидкість точок тремтівки.

Initial Values 1:

Initial Values – Initial Value for u2: – **u2** = 0.2*atan(cos(pi*x/2))

Initial Values – Initial time derivative of u2: – $\frac{\partial u2}{\partial t}$ = 0.6*sin(pi*x)*exp(sin(pi*y/2))

В другій задачі тремтівка буде закріплена на двох протилежних боках квадрата, а два інших будуть вільними.

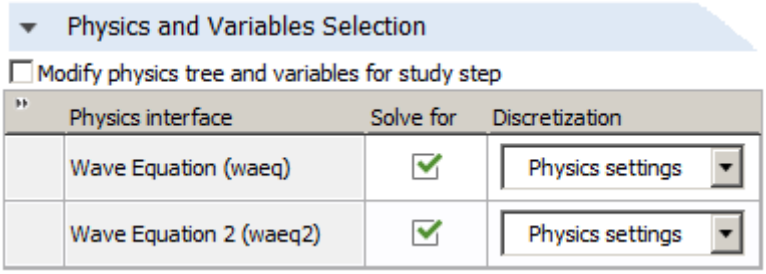
`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

`Boundary Selection` – **Selection,** `<>=Manual,1,4` (*лівий і правий боки*)

`Dirichlet Boundary Condition` – **`r`** `= 0` (*умова закріплення*)

Якщо тепер виділити вузол **Study 1 – Step 1: Time Dependent**, то в його ПВ можна побачити, що обчислюватися будуть обидві «фізики». Це відзначено прапорцями в таблиці завдань розділу **`Physics and Variables Selection`**.



Тепер, якщо виконати обчислення моделі, то в ДД `Study 1/Solution 1` будуть збережені дані як про розв'язок `u` першої задачі, так і про розв'язок `u2` другої.

Study 1 : LMB > Compute

Майте на увазі, що обчислення триватимуть довше, ніж попереднього разу.

Після завершення обчислень можна побудувати графік другого розв'язку у вигляді поверхні.

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

2D Plot Group 2 : `Data` – **Time** = `0.2`

`2D Plot Group 2`-> Add Plot - Surface

2D Plot Group 2 – Surface 1:

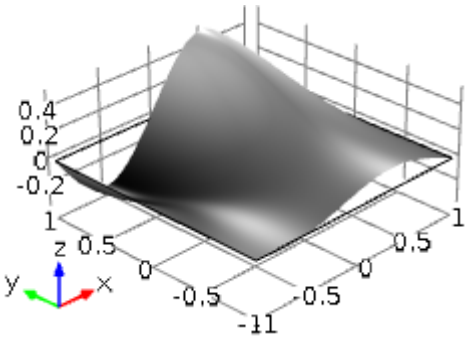
`Expression` – **Expression** = `u2` (*ім'я другого розв'язку*)

`Coloring and style` –**`Color table`** = `GrayScale`

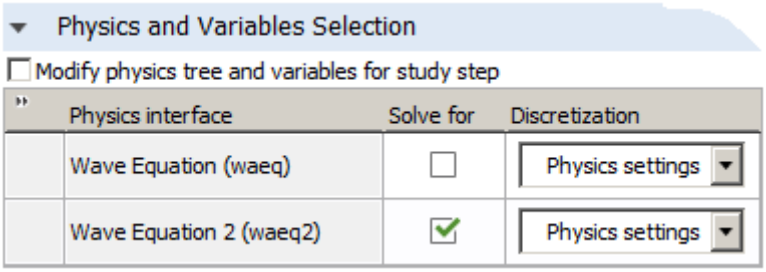
2D Plot Group 2 – Surface 1: RMB > Height Expression

Height Expression 1: `Axis` – **Scale factor** = `True ,1` (*два поля*)

LMB > Plot



Оскільки обчислення тривають достатньо довго, то для скорочення часу ви можете тимчасово не обчислювати першу задачу. Для цього достатньо в ПВ вузла **Study 1 – Step 1: Time Dependent** в таблиці завдань розділу **`Physics and Variables Selection`** зняти прапорець навпроти назви першої «фізики».



Але після цього в ДД `Study 1/Solution 1` буде загублена інформація про перший розв'язок `u`. Тому, після налаштування другого розв'язку, мабуть слід цей прапорець включити и заново обчислити модель (якщо вам потрібні обидва розв'язки `u` і `u2`).

Другий спосіб. Відкрийте файл `VibrSquare_sol2.mph`. В ньому змодельовано розв'язок першої задачі в гілці **Wave Equation(waeq)**. Зараз ми не будемо додавати другу «фізику», а створимо в цій гілці додаткові вузли початкової і граничної умов.

Wave Equation (waeq) - Initial Values 1: RMB > Duplicate

Initial Values 2:

Domain Selection – **Selection,** <> = `Manual,1`

Initial Values – **Initial Value for u:** – **u** = `0.2*atan(cos(pi*x/2))`

Initial Values – **Initial time derivative of u:** – $\frac{\partial u}{\partial t}$ = `0.6*sin(pi*x)*exp(sin(pi*y/2))`

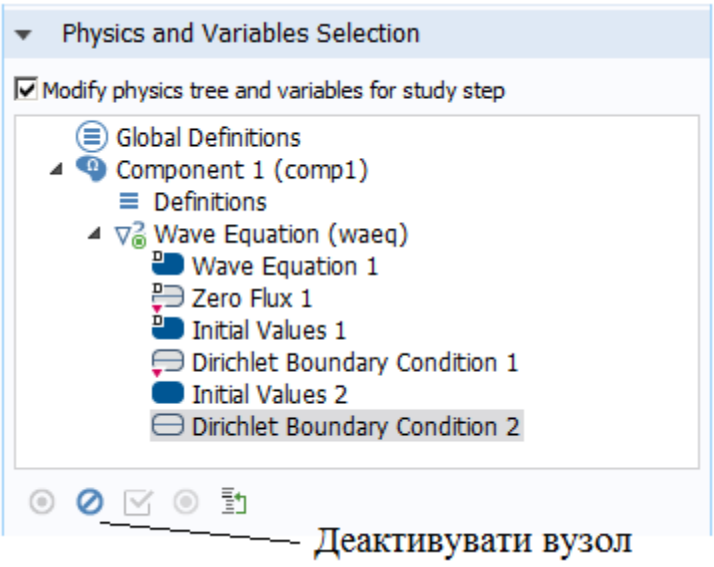
`Physics` -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection,** <>=`Manual,1,4` (*лівий і правий боки*)

Dirichlet Boundary Condition – **r** = `0` (*умова закріплення*)

Активуйте вузол **Study 1 – Step 1: Time Dependent** і в його ПВ в розділі **Physics and Variables Selection** встановіть прапрець **Modify physics tree and variables for study step**. Під ним з'явиться копія гілки **Component 1** дерева моделі.



Виділіть по черзі вузли **Initial Values 2** і **Dirichlet Boundary Condition 2**, та деактивуйте їх за допомогою кнопки Disable ⊘, розташованій нижче. Ці вузли будуть виключені з послідовності вузлів дерева моделі, і не будуть враховуватися при обчисленні гілки **Study 1**.

Study 1 : LMB > Compute

Після завершення обчислень графік в гілці **2D Plot Group 1** залишиться незмінним, бо графічна група підключена до поки що єдиного джерела даних **Study 1/Solution 1**, яке містить розв'язок першої задачі.

Створіть другий вузол `Study`, в якому врахуйте нові початкові і граничні умови.

VibrSquare_sol2.mph (root): RMB > Add Study = `Add Study ->`
Preset Studies - Time Dependent ->X

Налаштуйте створений вузол **Study 2**, задавши діапазон і інтервал зміни часу `t`, а також відносну і абсолютну похибку обчислень

Study 2 - Step 1: Time Dependent:

`Study Settings` –**Times** = `range(0,0.1,2.7)`

`Study Settings` –**Relative tolerance, <>** = `True, 0.0001`

Study 2: RMB > Show Default Solver

В цей момент окрім розкриття гілки **Study 2 – Solver Configurations** в гілці **Results – Data Sets** з'явиться вузол **Study 2/Solution 2**, який буде представляти другий розв'язок (який ще не обчислено).

Study 2 – Solver Configurations – Solution 1 – Time-Dependent Solver 1:

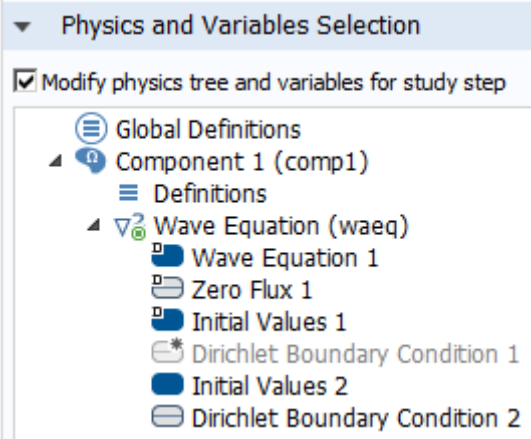
`Absolute Tolerance` – **Tolerance** = `0.0001`

Study 2 – Step 1: Time Dependent:

`Physics and Variables Selection` –

Modify physics tree and variables for study step = `True`

Під прапорцем з'явиться копія гілки **Component 1**, в якій деактивуйте вузол **Dirichlet Boundary Condition 1**. Це приведе до використання на боках квадрата граничних умов вузла **Dirichlet Boundary Condition 2**, а на верхній і нижній межі – умов вузла **Zero Flux 1**. Вузол **Initial Values 1** деактивувати неможливо, але це і непотрібно, бо його налаштування автоматично анулюються налаштуваннями вузла **Initial Values 2**, який в ДМ розташований нижче.



Виконайте обчислення вузла **Study 2**.

Study 2: LMB > Compute

Після завершення обчислень в гілці **Results** з'явиться графічна група **2D Plot Group 2** з вузлом **Surface1**, що містить графік нового розв'язку. Зверніть увагу, що ця графічна група підключена до другого джерела даних **Study 2/Solution 2** і, як наслідок, містить розв'язок другої задачі.

Змініть у вузлі **Surface1** палітру і додайте підвузол `Height Expression`.

2D Plot Group 2 : `Data` – Time = `0.2`

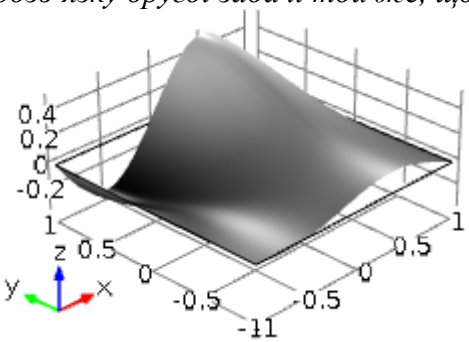
2D Plot Group 2 – Surface 1:

`Coloring and style` –`Color table` = `GrayScale`

2D Plot Group 2 – Surface 1: RMB > Height Expression

Height Expression 1: `Axis` – Scale factor = `True ,1` (*два поля*)

LMB > Plot (*графік розв'язку другої задачі той же, що раніше в першому способі*)



■

Таким чином, для розв'язання двох задач, які відрізняються лише початковими і граничними умовами, можна створити дві «фізики» і один вузол досліджень `Study`, або одну «фізику» і два вузли досліджень. Другий спосіб не підходить, коли ДРЧП задач відрізняються. При використанні першого способу можна в кожній «фізиці» задати власне ДРЧП (інше рівняння або інші коефіцієнти) і власні початкові та граничні умови.

3.22. Коливання струни з обчислювальною початковою формою.

Зустрічаються ситуації, коли комп'ютерний експеримент природно розпадається на етапи, які моделюються різними рівняннями. При чому розв'язок однієї задачі використовується в рівняннях, початкових або граничних умовах другої.

Для розв'язання подібних задач в COMSOL передбачена можливість виконання обчислень/досліджень по крокам. Спочатку розв'язується одна задача. Після неї розв'язується інша. Кроки можна налаштувати таким чином, щоб їх задачі обчислювалися послідовно, і результат попереднього кроку використовувався в наступному. Кроків дослідження може бути декілька.

Розглянемо малі поперечні коливання струни довжини L, яка закріплена на кінцях. Це досить проста задача, коли початковий профіль $\varphi(x)$ і початкова швидкість $\psi(x)$ точок струни відомі. Потрібно розв'язати одновимірне хвильове рівняння [1]

$$U''_{tt}(x,t) = a^2 U''_{xx}(x,t),$$

де $U(x,t)$ являє зміщення з положення рівноваги точки x струни в момент часу t, a – швидкість розповсюдження хвиль вздовж струни. Для однозначного визначення розв'язку необхідно задати початкові умови

$$U(x,0)=\varphi(x),\ U'_t(x,t)=\psi(x),$$

а також граничні умови, які у випадку закріплення кінців мають вигляд $U(0,t)=0, U(L,t)=0$.

Створення початкових умов потребує проведення фізичного експерименту, який зазвичай моделюється іншою крайовою задачею. В нашому прикладі ми вважатимемо, що спершу струна не рухається і знаходиться під впливом точкової зовнішньої сили. Потім, в нульовий момент часу сила зникає, і струна починає коливатися, маючи нульову початкову швидкість і початкову форму, яка визначається статичною задачею прогину. Тобто розв'язок статичної задачі прогину струни визначає початковий профіль $\varphi(x)$ для задачі коливань.

Диференціальне рівняння рівноваги струни має вигляд $u''_{xx}=-q(x)$, де $q(x)=F(x)/T$, T – натяг струни, $F(x)$ – зовнішня (поперечна) сила обчислена на одиницю довжини. Функція $u(x)$ являє вертикальне зміщення точок струни, які мали до навантаження горизонтальну координату x. До рівняння рівноваги потрібно додати граничні умови закріплення кінців струни.

Таким чином, математична модель задачі складається з послідовного розв'язання двох крайових задач:

Задача 1. $u''_{xx}=-q(x),\ u(0)=0, u(L)=0$.

Задача 2. $U''_{tt}(x,t)=a^2U''_{xx}(x,t)$,

$$U(x,0)=u(x),\ U'_t(x,t)=0,$$
$$U(0,t)=0, U(L,t)=0.$$

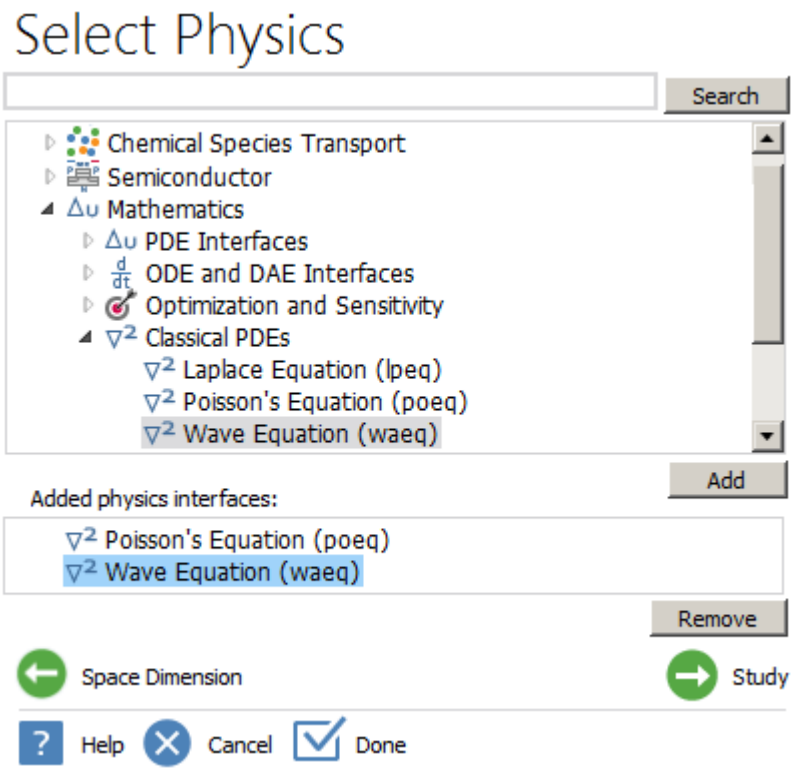
В прикладі покладемо $a=1, L=1$. Вважатимемо також, що натяг струни T=1, а зовнішня сила F=−1 прикладена в одній точці $x_0=0.9$, тобто $q(x)=-\delta(x-x_0)$, де $\delta(x)$ - дельта функція Дірака.

Приклад. За допомогою майстра створіть одновимірну модель задачі.

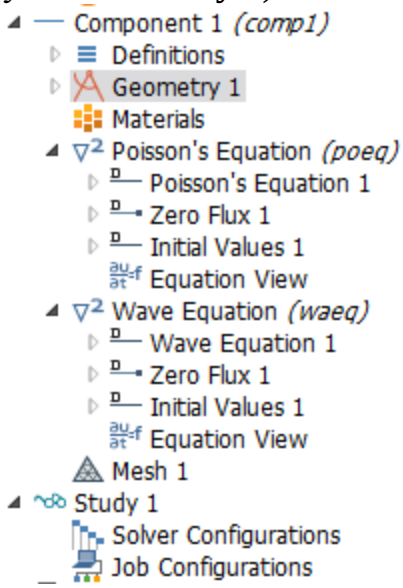
File -> New -> Model Wizard -> 1D

В панелі Select Physics виберіть Mathematics – Classical PDEs – Poisson's Equation (poeq) і натисніть кнопку Add. В поле **Added physics interfaces**, що розташовано нижче, додається назва моделі Poisson's Equation(poeq). В полі **Dependent Variable** залиште ім'я шуканої величини u. Значення інших полів також залиште без змін, але вікно Select Physics не закривайте!

В панелі Select Physics вдруге виберіть Mathematics – Classical PDEs – Wave Equation (waeq) і знову натисніть кнопку Add. В поле **Added physics interfaces** додасться назва ще однієї моделі Wave Equation (waeq). В полі **Dependent Variable** змініть ім'я шуканої величини на U, а значення інших полів залиште без змін.



Натисніть внизу кнопку Study Study. Відкриється вікно `Select Study`. Розкрийте гілку `Custom Study` і виберіть варіант `Empty Study`. Закінчить все натисканням на кнопку Done. Створюється вікно задачі з деревом моделі, що містить вузли двох фізик: `Poisson's Equation (poeq)` і `Wave Equation (waeq)`. При цьому область, на якій діятимуть їх рівняння, буде спільною (відрізком [0,1], заданим у вузлі **Geometry 1**).



Вимкніть в моделі контроль розмірностей фізичних величин.

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

Створіть таблицю параметрів моделі.

Global Definitions : RMB > Parameters

Заповніть таблицю так, як показано нижче.

Parameters

Name	Expression	Value	Description
L	1	1	довжина струни
x0	0.9	0.9	точка прикладення сили

Геометрія моделі складається з відрізка [0, L] і точки. Створіть відрізок.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1) : Interval – Right endpoint = L

Додайте в геометрію точку, до якої пізніше буде прикладено зосереджену силу.

Geometry**-> Primitives –** Point

Geometry 1 - Point 1 : Point – x = x0

LMB > Build All Objects

На створеній геометрії генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – Element Size = Extremely fine

LMB> Build All

Налаштуйте статичну задачу (Poisson's Equation), задавши коефіцієнти рівняння Пуассона і нульові граничні умови. Спочатку перевірте, що в ПВ гілки **Poisson's Equation (poeq)** в розділі **Domain Selection** в секції вибору стоїть 1 (індикатор відрізка). Далі введіть

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – c = -1

Source Term – f = 0 (*точкове зусилля буде задано окремо*)

Physics -> **Boundaries –** Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – Selection,<> = Manual,1,3 (*кінці відрізка*)

Dirichlet Boundary Condition – r = 0

Ми вже казали, що в правих частинах рівняння Пуассона $\nabla\cdot(-c\nabla u)=f$ член f являє розподілений зовнішній вплив. Для задання точкового зусилля (в одновимірних задачах) призначені вузли типу Flux/Source.

Poisson's Equation (poeq):

Physics **-> Boundaries –** Flux/Source

В ПВ створеного вузла **Flux/Source 1** вкажіть точку прикладання зусилля і введіть його величину.

Poisson's Equation (poeq) - Flux/Source 1:

Boundary Selection – Selection, <> = Manual, 2 (*точка* x=$x0$)

Boundary Flux/Source – g =-1 (*величина зосередженого зусилля*)

Налаштування першої задачі (статичного прогину струни) тут закінчується.

Налаштуйте задачу коливань (Wave Equation).

Wave Equation (waeq) - Wave Equation 1:

Diffusion Coefficient – c = 1

Source Term – f = 0

Absorption Coefficient – e_a = 1

Цим задається хвильове рівняння $U''_{tt}-U''_{xx}=0$.

Задайте умову закріплення кінців струни.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection – **Selection,**<> = Manual,1,3 (*кінці відрізка*)

Dirichlet Boundary Condition – **r** = 0

Для задачі коливань потрібно задати початкові умови.

Wave Equation (waeq) - Initial Values 1:

Domain Selection – **Selection,** <> = Manual,1,2 (*увесь відрізок*)

Initial Values – **Initial Value for U:**– **U** = u

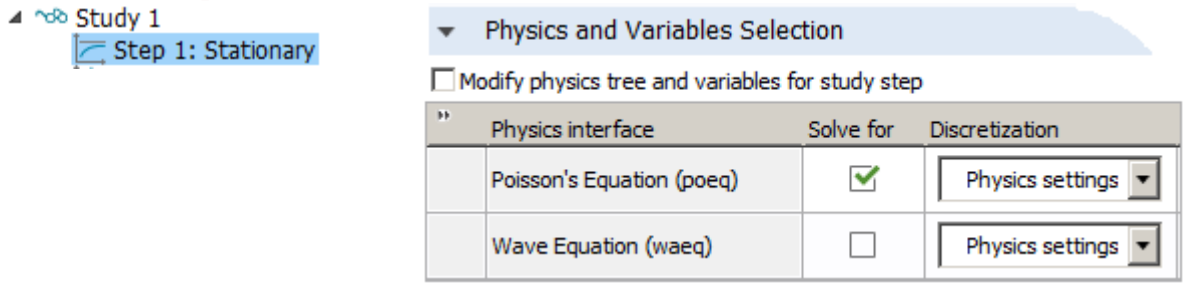
Initial Values – **Initial time derivative of U:**– $\frac{\partial U}{\partial t}$ =0

Зверніть увагу, що в початковій умові використана змінна u, яка буде обчислюватися в першій задачі.

Прийшов час налаштувати вузол Study. Кроки дослідження додаються з випадного меню.

Study 1: RMB> Study Steps – Stationary – Stationary

В ПВ створеного вузла **Step 1: Stationary** в розділі **Physics and Variables Selection** в таблиці завдань зніміть прапорець навпроти задачі Wave Equation(waeq).



Це означатиме, що на першому кроці хвильове рівняння розв'язуватися не буде.

Створіть і налаштуйте другий крок дослідження.

Study 1: RMB> Study Steps – Time Dependent – Time Dependent

Study 1 - Step 2: Time Dependent:

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

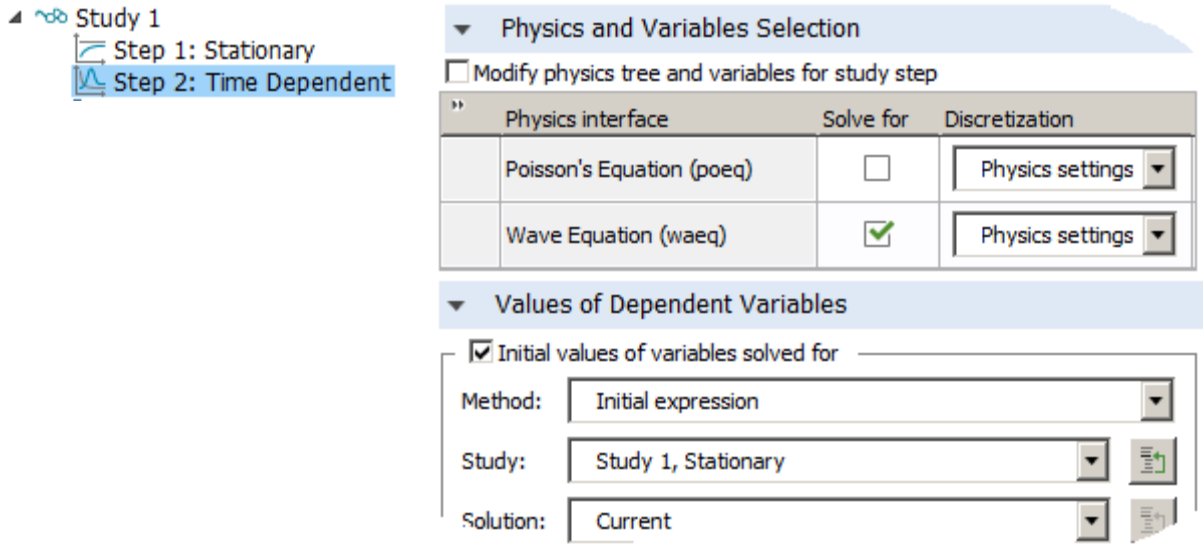
Study Settings –**Relative tolerance,** <> = True, 0.0001

Study 1: RMB > Show Default Solver

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

Absolute Tolerance – **Tolerance** = 0.0001

В ПВ вузла **Step 2: Time Dependent** в розділі **Physics and Variables Selection** в таблиці завдань зніміть прапорець навпроти задачі Poisson's Equation (poeq).



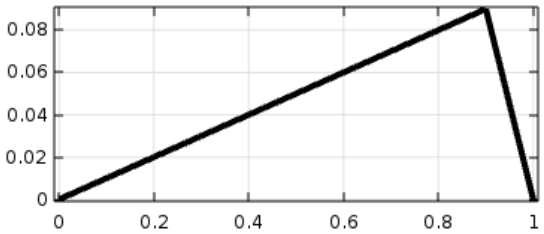
Це означатиме, що на другому кроці рівняння Пуассона розв'язуватися не буде.

В розділі **Values of Dependent Variables** вказується, як буде застосовано розв'язок задачі, що відключена. На рисунку вгорі показані налаштування, які потрібно задати, щоб розв'язок першої задачі `u` можна було використати в якості початкової умови другої задачі. Нагадаємо, що ім'я `u` нами вже один раз використано у вузлі **Wave Equation (waeq) - Initial Values 1** при заданні початкової умови. Вибір `Initial expression` в списку **Method** дає дозвіл на таке використання.

Виконайте обчислення.

Study 1: LMB > Compute

Після їх завершення в гілці **Results** з'являться дві графічні групи. Група **2D Plot Group 1** містить графік розв'язку `u`, тобто графік початкового профілю струни.



Група **2D Plot Group 2** містить графіки форми струни у всі моменти часу. Налаштуйте цю групу.

1D Plot Group 2:

Data –Time selection = `Manual`

Data –Times (1-21) = `4`

Axis –Manual axis limits = `True`

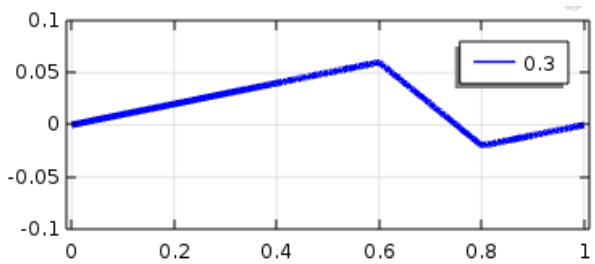
Axis –y minimum, y maximum = `-0.1, 0.1` (*два поля*)

1D Plot Group 2 - Line Graph 1:

Coloring and style - Line style – Width = `4`

Legends–Show legends = `True`

LMB > Plot



Для наочності побудуйте анімацію коливань (самостійно). Спробуйте також змінити параметр `x0`, наприклад покладіть `x0=0.5`, і переобчисліть модель.

Вправа. Розв'яжіть наступну послідовність задач (L=1):

Задача 1. $u''_{xx} = -q(x)$, $u(0)=0$, $u(L)=0$, де $q(x)$ таке ж саме як в прикладі.

Задача 2. $U''_{tt}(x,t) = U''_{xx}(x,t) + u(x)$,

$U(x,0)=0$, $U'_t(x,t)=0$,

$U(0,t)=0$, $U(L,t)=0$.

Вказівка. В налаштуваннях вузла **Step 2: Time Dependent** в розділі **`Values of Dependent Variables`** потрібно зняти прапорець **Initial values of variables solved for** (він знятий за замовчуванням).

3.23. Система двох ДРЧП.

Розглянемо в одиничному колі Ω крайову задачу, яка складається з системи диференціальних рівнянь в частинних похідних [9]

$$\begin{aligned} &\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -q_0 \\ &\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} = -\frac{u}{D} \end{aligned}, \qquad (1)$$

і нульових граничних умов $u|_{\partial\Omega} = 0$, $v|_{\partial\Omega} = 0$. Для цієї задачі відомий точний розв'язок. Він має вигляд

$$\begin{aligned} u(x,y) &= \frac{q_0}{4}\left(1 - x^2 - y^2\right) \\ v(x,y) &= \frac{q_0}{64D}\left(x^4 + 2x^2y^2 + y^4 - 4x^2 - 4y^2 + 3\right) = \\ &= \frac{q_0}{64D}\left(x^2 + y^2 - 1\right)\left(x^2 + y^2 - 3\right) \end{aligned} \qquad (2)$$

Дійсно, $u''_{xx} = -\frac{q_0}{2}$, $u''_{yy} = -\frac{q_0}{2}$, $u''_{xx} + u''_{yy} = -q_0$. Далі $v''_{xx} = \frac{q_0}{64D}\left(12x^2 + 4y^2 - 8\right)$, $v''_{yy} = \frac{q_0}{64D}\left(12y^2 + 4x^2 - 8\right)$ і $v''_{xx} + v''_{yy} = \frac{q_0}{4D}\left(x^2 + y^2 - 1\right) = -\frac{u}{D}$.

Очевидно, що функції (2) задовольняють нульовим граничним умовам.

Ці розв'язки ми зможемо порівняти з наближеними, які будуть побудовані системою.

Зауваження. Система (1) еквівалентна рівнянню Софі Жермен згину тонких платівок $\Delta\Delta v = q_0 / D$ (ДРЧП 4-го порядку).

Системи ДРЧП в COMSOL можна розв'язувати різними способами. Для нашої системи ми наведемо три: перший – за допомогою математичного інтерфейсу `Coefficient Form PDE`, другий – розв'язуючи два з'єднаних рівняння Гельмгольца, третій – створюючи для кожного рівняння окремий компонент.

Приклад. Розв'яжемо задачу (1), в якій покладемо $q_0 = 1, D = 1$.

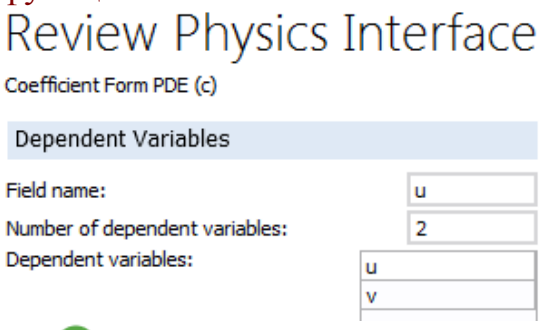
Розв'язок 1. Розв'язання системи двох ДРЧП з використанням математичного інтерфейсу `Coefficient Form PDE` .

Створіть шаблон моделі.

File->New->Model Wizard->2D->

Mathematics- PDE Intefaces - Coefficient Form PDE (c)

В панелі `Review Physics Interface` в розділі **`Dependent Variables`** в полі **Number of dependent variables** введіть 2, а в полі **Dependent variables** введіть імена шуканих функцій `u` і `v`.



Натисніть кнопку Study Study. В панелі `Select Study` виберіть варіант `Stationary`. Завершіть створення шаблона моделі натисканням на кнопку Done.

Нагадаємо, що таку послідовність інструкцій ми домовилися записувати коротко наступним чином

File->New->Model Wizard->2D->

Mathematics- PDE Intefaces - Coefficient Form PDE (c) **->**

Number of dependent variables: = `2` +

Dependent variables: =`[u,v]` **->** Stationary

де символ **+** (плюс) позначає виконання додаткового вводу в поточному вікні або панелі в полі, назва якого стоїть після знака **+**.

Накресліть коло. При активному вузлі **Geometry 1** виконайте команду

`Geometry->`**`Draw`**`->`Circle

Наведіть курсор миші в графічне вікно, намалюйте коло, і в ПВ створеного вузла **Circle 1 (c1)** скорегуйте параметри.

Geometry 1 - Circle 1 (c1):

`Size and Shape` – **Radius** = `1`

`Position` – **x, y** = `0,0` (*два поля*)

LMB > Build All Objects

На створеній геометрії генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Component 1 (comp1) - Mesh 1 : LMB> Build All

В дереві моделі виділіть вузол **Coefficient Form PDE 1**. Справа відкриється його панель. В розділі **`Equation`** цієї панелі наведено рівняння

$$e_a \frac{\partial^2 \boldsymbol{u}}{\partial t^2} + d_a \frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \left(-c \nabla \boldsymbol{u} - \alpha \boldsymbol{u} + \gamma\right) + \beta \cdot \nabla \boldsymbol{u} + a \boldsymbol{u} = f \,, \qquad (3)$$

де $\boldsymbol{u} = \begin{pmatrix} u \\ v \end{pmatrix}$ – шукана вектор–функция і $\nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y}\right]$ – оператор градієнта. Тут $e_a, d_a, \alpha, \beta, \gamma$ тензорні коефіцієнти, які в нашій задачі слід покласти рівними нулю. Тоді (3) зведеться до рівняння

$$\nabla \cdot \left(-c \nabla \boldsymbol{u}\right) + a \boldsymbol{u} = f \,, \qquad (4)$$

В позначеннях Comsol Multyphysics операторна формула $\nabla \cdot \left(c \nabla u\right)$ є коротким записом виразу $\frac{\partial}{\partial x_1}\left(c \frac{\partial u}{\partial x_1}\right) + \ldots + \frac{\partial}{\partial x_n}\left(c \frac{\partial u}{\partial x_n}\right)$. При цьому `u` може бути скалярною або векторної функцією. В нашому прикладі `u` – двовимірна вектор-функція, а c - матриця. Задавши $c = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}$, матимемо

$$\nabla \cdot \left(-c \nabla \boldsymbol{u}\right) = \frac{\partial}{\partial x}\left(\frac{\partial \boldsymbol{u}}{\partial x}\right) + \frac{\partial}{\partial y}\left(\frac{\partial \boldsymbol{u}}{\partial y}\right) = \begin{pmatrix} u''_{xx} + u''_{yy} \\ v''_{xx} + v''_{yy} \end{pmatrix}.$$

Тоді при $a = \begin{pmatrix} 0 & 0 \\ 1/D & 0 \end{pmatrix}$ ліва частина (4) набуде вигляду

$$\nabla \cdot \left(-c \nabla \boldsymbol{u}\right) + a \boldsymbol{u} = \begin{pmatrix} u''_{xx} + u''_{yy} \\ v''_{xx} + v''_{yy} \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1/D & 0 \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} u''_{xx} + u''_{yy} \\ v''_{xx} + v''_{yy} + \frac{1}{D} u \end{pmatrix}.$$

Поклавши в (4) $f = \begin{pmatrix} -q_0 \\ 0 \end{pmatrix}$, приходимо до системи (1).

Таким чином, щоб задати систему рівнянь в ПВ вузла **Coefficient Form PDE 1** в розділах **`Diffusion Coefficient`** (коефіцієнт `c`), **`Absorption Coefficient`** (коефіцієнт a), **`Source Term`** (член `f`), **`Mass Coefficient`** (коефіцієнт e_a) і **`Damping or Mass Coefficient`** (коефіцієнт d_a) слід ввести

$$c = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \quad a = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \quad f = \begin{pmatrix} -1 \\ 0 \end{pmatrix} \quad e_a = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \quad d_a = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

Інші коефіцієнти/тензори α, β, γ залиште нульовими (значення за замовчуванням). Перевірте також, що в поточній панелі **Coefficient Form PDE 1** в розділі **`Domain Selection`** стоїть 1, тобто обрано коло.

Задайте граничні умови.

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection– **Selection,** <> = Manual;1,2,3,4 (*контур кола*)

Dirichlet Boundary Condition–**r1, r2** = 0,0 (*граничні значення u і v*)

Виконайте обчислення.

Study 1 : LMB > Compute

В графічній зоні з'явиться кругова область з функціональним забарвленням, відповідним розв'язку u (наступний рисунок ліворуч). Графік розв'язку v потрібно побудувати окремо.

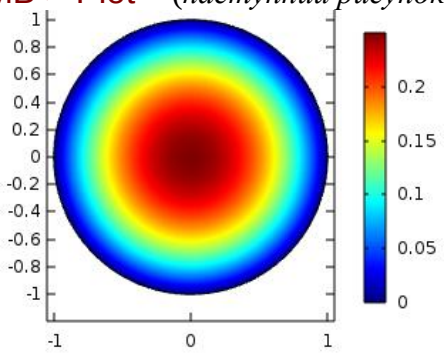
Results -> 2D Plot Group

2D Plot Group 2: RMB > Surface

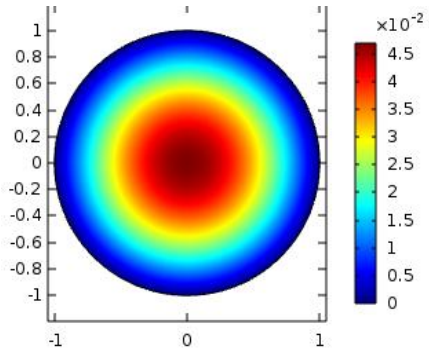
2D Plot Group 2 – Surface 1:

Expression – **Expression** = v

LMB > Plot (*наступний рисунок праворуч*)



Розв'язок u



Розв'язок v

Обчисліть максимуми модулів різниці наближених (u,v) і точних розв'язків (2). Для цього виконайте інструкції

Results->Derived Values- More Derived Values->Surface Maximum

Derived Values – Surface Maximum 1:

Data – **Data set** = Study 1/Solution 1

Selection – **Selection, Selection** = Manual,1 (*індикатор кола*)

Expression – **Expression** = abs(u-(1-x^2-y^2)/4)

LMB > Evaluate

Нижче графічного вікна буде створена таблиця зі значенням максимуму введеного виразу (похибка обчислень величини u). Таким же чином знайдіть похибку обчислень величини v.

Results->Derived Values- More Derived Values->Surface Maximum

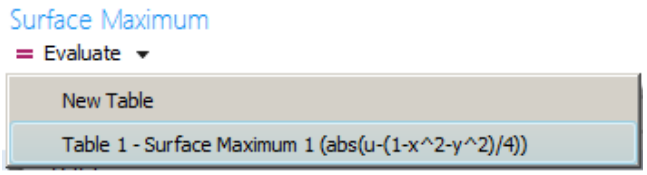
Derived Values – Surface Maximum 2:

Data – **Data set** = Study 1/Solution 1

Selection – **Selection, Active** = Manual,1 (*індикатор кола*)

Expression – **Expression** = abs(v-(x^2+y^2-1)*(x^2+y^2-3)/64)

LMB > Evaluate >Table 1 – Surface Maximum 1



В існуючу таблицю, розташовану під графічним вікном, буде додано друге значення.

abs(u-(1-x^2-y^2)/4)	abs(v-(x^2+y^2-1)*(x^2+y^2-3)/64)
1.2998E-5	6.0977E-6

Як бачите, абсолютна похибка обчислення функції u не перевищує $1.3\cdot10^{-5}$, а функції v – менше ніж $6.1\cdot10^{-6}$. ■

Розв'язок 2. Використання двох пов'язаних «фізик».

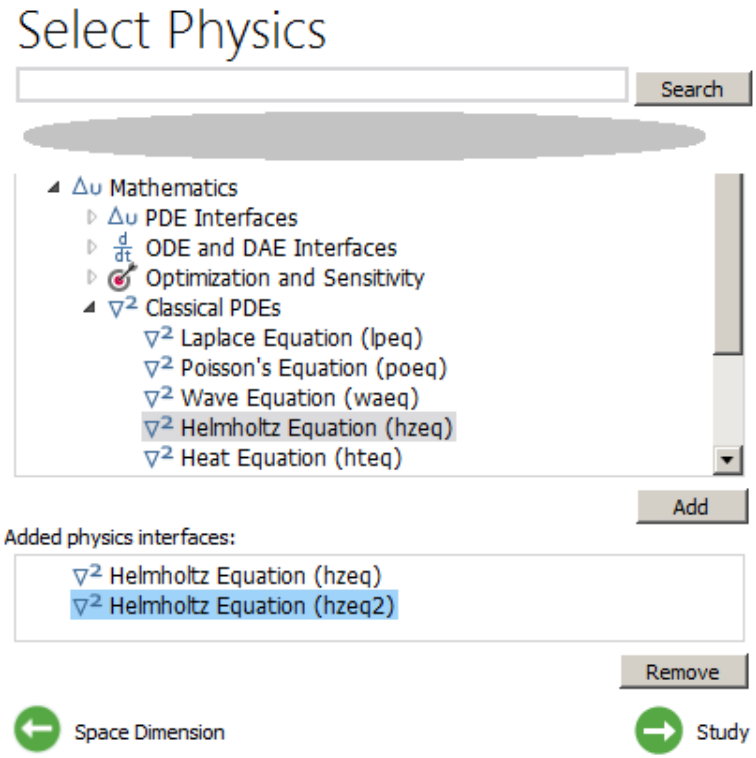
Створіть модель послідовністю команд:

File–>New–>Model Wizard–>2D–>

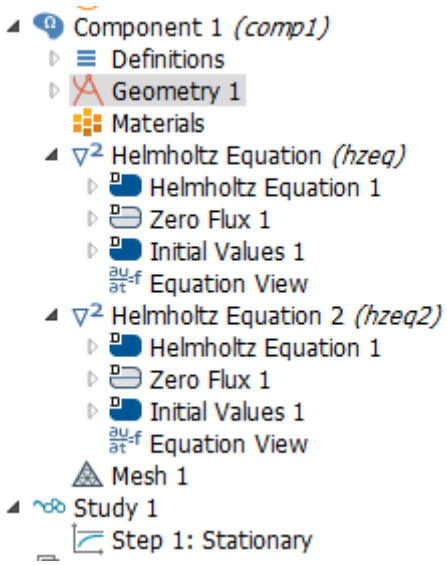
Mathematics– Classical PDEs – Helmholtz Equation (hzeq)

В поле **Added physics interfaces**, розташованому нижче, додасться назва моделі Helmholtz Equation (hzeq). В полі **Dependent Variable** залиште ім'я шуканої функції u. Значення інших полів залиште без змін, але вікно Select Physics не закривайте!

В розділі Select Physics ще раз виберіть Helmholtz Equation (hzeq) і знову натисніть кнопку Add. В поле **Added physics interfaces** додасться назва ще однієї моделі Helmholtz Equation (hzeq2). В полі **Dependent Variable** змініть ім'я шуканої функції на v, а значення інших полів залиште без змін.



Натисніть внизу кнопку Study. Відкриється вікно Select Study. Виберіть варіант Stationary. Завершіть побудову шаблона моделі натисканням на кнопку Done. Створюється вікно задачі. Дерево моделі буде містити дві гілки, які відповідають двом пов'язаним рівнянням Гельмгольца. Зв'язаність означає, що імена невідомих функцій одного рівняння можуть з'являтися в виразах коефіцієнтів іншого. При цьому область, на якій діють ці рівняння, буде спільною, заданої у вузлі **Geometry 1**.



Накресліть коло. При активному вузлі **Geometry 1** виконайте команду

`Geometry->Draw->Circle`

Наведіть курсор миші в графічне вікно, намалюйте коло, і в ПВ створеного вузла **Circle 1 (c1)** скорегуйте параметри.

Geometry 1 - Circle 1 (c1):

Size and Shape – **Radius** = 1

Position – **x, y** = 0,0 (*два поля*)

LMB > Build All Objects

Зауваження. Обидві фізики, тобто обидва рівняння Гельмгольца, мають спільну розрахункову область.

На створеній геометрії генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 : LMB> Build All

Задайте перше рівняння Гельмгольца (перше рівняння системи (1)).

Helmholtz Equation (hzeq):

Domain Selection – **Selection** = 1 (*індикатор кола*)

Цим задається зона дії рівняння. В розділі **Equation** наведено рівняння $\nabla\cdot(-c\nabla u)+au=f$. Щоб воно набуло вигляду $u''_{xx}+u''_{yy}=-1$ першого рівняння (1), введіть наступне

Helmholtz Equation (hzeq) - Helmholtz Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = -1

Absorption Coefficient – **a** = 0

Налаштуйте друге рівняння системи. Перевірте, що в ПВ вузла **Helmholtz Equation 2 (hzeq2)** в розділі **Domain Selection** стоїть 1. Далі введіть

Helmholtz Equation 2 (hzeq2) - Helmholtz Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = -u[1/m^2]

Absorption Coefficient – **a** = 0

Цим ми задали друге рівняння $v''_{xx}+v''_{yy}=-u$ системи (1).

Задайте граничні умови в першій фізиці (для величини u).

Helmholtz Equation (hzeq):

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Helmholtz Equation (hzeq) - Dirichlet Boundary Condition 1:

`Boundary Selection`– **Selection,** <> = `Manual;1,2,3,4` (*контур кола*)

`Dirichlet Boundary Condition`–**r** =`0` (*граничне значення u*)

Задайте граничні умови в другій фізиці (для величини v).

Helmholtz Equation 2 (hzeq2):

`Physics` -> **`Boundaries`** – Dirichlet Boundary Condition

Helmholtz Equation 2 (hzeq2) - Dirichlet Boundary Condition 1:

`Boundary Selection`– **Selection,** <> = `Manual;1,2,3,4` (*контур кола*)

`Dirichlet Boundary Condition`–**r** =`0` (*граничне значення v*)

Виконайте обчислення.

Study 1 : LMB > Compute

Після завершення обчислень в графічній зоні ви побачите кругову область з функціональної забарвленням, яке відповідає розв'язку u. Воно належить вузлу **2D Plot Group 1**. Одночасно в гілці **Results** буде створено вузол **2D Plot Group 2**, який містить зображення функції v. Графіки ті ж самі, що було побудовано в розв'язку 1.

Як і в першому розв'язку знайдіть максимуми модулів різниці наближених (u,v) і точних розв'язків (2). Виконавши всю послідовність дій (самостійно), ви отримаєте таблицю з двома значеннями

abs(u-(1-x^2-y^2)/4)	abs(v-(x^2+y^2-1)*(x^2+y^2-3)/64)
1.2998E-5	6.0977E-6

Оскільки СкЕ сітка в колі була такою ж, як і в першому розв'язку, то похибка виявилася такою ж. Ви можете зменшити розміри клітинок сітки. Тоді зменшиться і похибка обчислень.

Можна побудувати графік похибки.

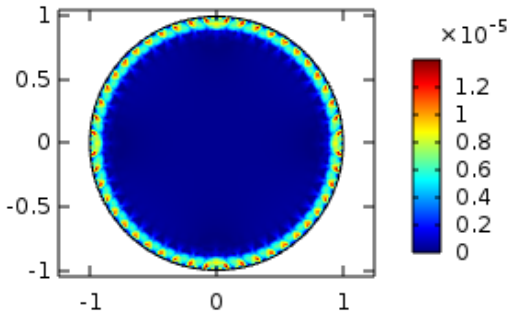
`Results` -> 2D Plot Group

2D Plot Group 3: RMB > Surface

2D Plot Group 2 – Surface 1:

`Expression` – **Expression** = `abs(u-(1-x^2-y^2)/4)`

LMB > Plot



По шкалі кольорів, розташованій праворуч, можна оцінити значення похибки в різних частинах кола. Для тренування побудуйте графік цієї похибки на межі кола. Для цього спочатку треба створити джерело даних `Edge 2D`.

Results – Data Sets: RMB > Edge 2D

Data Sets – Edge 2D 1:

Data – **Data set** = Study1/Solution 1

Selection – **Selection** =All boundaries (*контур кола*)

Тепер можна будувати графік.

Results -> 1D Plot Group

1D Plot Group 4:

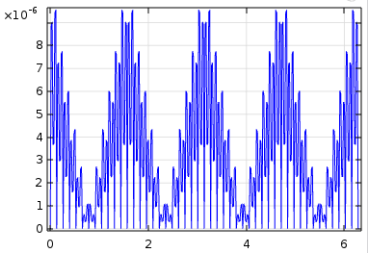
Data – **Data set** = Edge 2D 1

RMB > Line Graph

1D Plot Group 4 – Line Graph 1:

y-Axis Data – **Expression** = abs(u-(1-x^2-y^2)/4)

LMB > Plot



■

Зауваження 1. Оскільки в перше рівняння (1) функція v не входить, то розв'язок задачі можна виконувати по крокам так, як описано в попередньому прикладі.

Зауваження 2. Використану тут модель, що містить два пов'язаних рівняння Гельмгольца, можна побудувати по-іншому. Спочатку можна створити модель з одним рівнянням інструкціями

File->New->Model Wizard->2D->

Mathematics- Classical PDEs - Helmholtz Equation (hzeq)->Stationary

В дереві моделі в гілці **Component 1 (comp1)** буде створений один вузол **Helmholtz Equation (hzeq)**. Потім, коли шаблон моделі відкритий, можна додати ще одне рівняння Гельмгольца, виконавши команду

Home **-> Physics ->** Add Physics

Праворуч графічного вікна відкриється панель Add Physics, в якій слід вибрати ім'я ще однієї задачі Helmholtz Equation (hzeq). В результаті буде створено шаблон моделі з двома фізиками/рівняннями Гельмгольца, як і вище.

Розв'язок 3. Ще раз розв'яжемо систему (1)

$$\begin{cases} u''_{xx} + u''_{yy} = -1 \\ v''_{xx} + v''_{yy} = -u \end{cases}, \qquad (1)$$

розміщуючи кожне рівняння в окремому компоненті. Компонент – це фізична модель з власною геометрією і ДРЧП. Кожне з наведених рівнянь (1) буде розглядатися як компонент – окрема крайова задача для рівняння Пуассона. Компоненти будуть пов'язані за допомогою спеціального оператора загальної екструзії (General Extrusion operator).

Створіть модель (поки що складену з одного компонента) послідовністю команд:

File->New->Model Wizard->2D->

Mathematics- Classical PDEs – Poisson's Equation (poeq) **->**Stationary

Зверніть увагу, що ім'я шуканої функції u ми залишили незмінним.

Вимкніть в моделі контроль розмірностей фізичних величин.

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

Геометрія моделі складається з одиничного кола. Створіть його.

Geometry 1 :> Geometry->**Draw**->Circle

Наведіть курсор миші в графічне вікно і намалюйте коло. В ПВ створеного вузла **Circle 1 (c1)** скорегуйте параметри.

Geometry 1 - Circle 1 (c1):

Size and Shape – **Radius** = 1

Position – **x, y** = 0,0 (*два поля*)

LMB > Build All Objects

На створеному колі генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 : LMB> Build All

Щоб рівняння Пуассона $\nabla\cdot(-c\nabla u)=f$ набуло вигляду першого рівняння (1), введіть

Poisson's Equation (poeq) - Poisson's Equation 1:

Diffusion Coefficient – **c** = -1

Source Term – **f** = -1

Задайте граничну умову для величини u.

Poisson's Equation (poeq):

Physics -> **Boundaries** – Dirichlet Boundary Condition

Poisson's Equation (poeq) - Dirichlet Boundary Condition 1:

Boundary Selection– **Selection,** <> = Manual;1,2,3,4 (*контур кола*)

Dirichlet Boundary Condition–**r** =0 (*граничне значення u*)

Ми ще не закінчили працювати з гілкою **Component 1**. Але до неї ми повернемося трохи пізніше. Зараз створіть компонент з другим рівнянням (1).

Home->**Add Component**->2D

Кожний компонент представляє окрему задачу з власною розрахунковою областю. Тому в кожному компоненті геометрія створюється окремо. Друге рівняння (1) діятиме на такому ж колі, як в першому компоненті. Створіть його.

Component 2 - Geometry 2 :> Geometry->**Draw** - Circle

Намалюйте мишею коло і в ПВ створеного вузла **Circle 1 (c1)** скорегуйте його параметри.

Component 2 - Geometry 2 - Circle 1 (c1):

Size and Shape – **Radius** = 1

Position – **x, y** = 0,0 (*два поля*)

LMB > Build All Objects

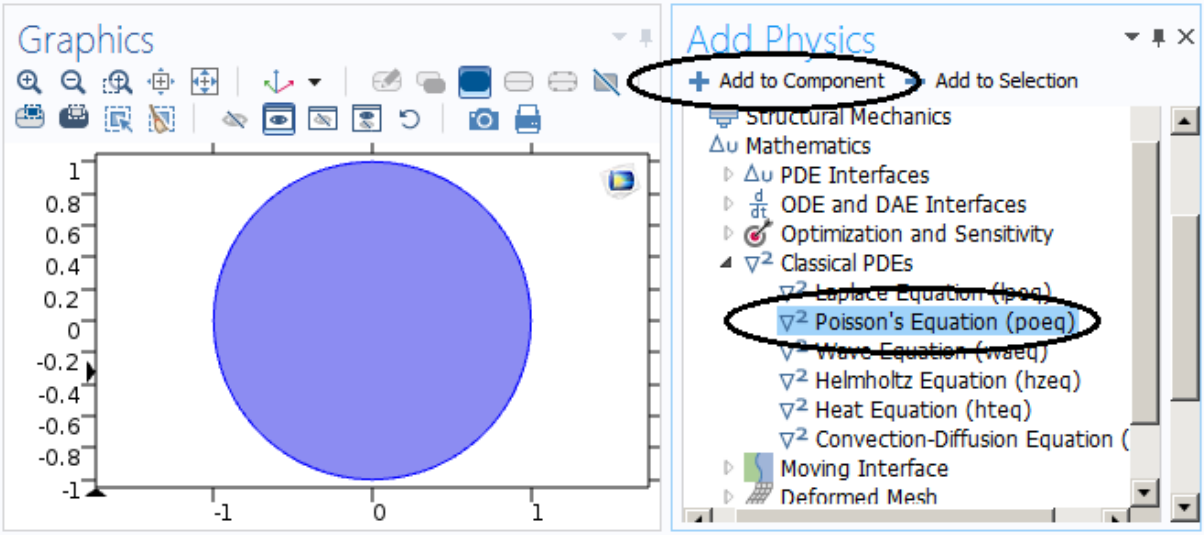
На створеному колі генеруйте СкЕ сітку.

Component 2 (comp1) - Mesh 2 : LMB> Build All

В новому компоненті ще немає ніякої математичної моделі. Додайте її.

Component 2 :> Home->**Physics**->Add Physics

Праворуч графічного вікна відкриється панель вибору фізичної/математичної моделі Add Physics.



Виберіть в ньому варіант Poisson's Equation (poeq) і натисніть кнопку Add to Component, яка розташована вгорі панелі. Панель Add Physics закрийте кнопкою-хрестиком в правому верхньому куті. В дереві моделі виділіть створений вузол **Poisson's Equation 2 (poeq2)**, і в його ПВ в розділі **Dependent Variables** в полі **Dependent Variable** змініть ім'я шуканої функції на v.

Задайте рівняння.

Component 2 - Poisson's Equation 2 (poeq2) - Poisson's Equation 1:

В розділі **Equation** наведено рівняння $\nabla\cdot(-c\nabla v)=f$. Щоб воно набуло вигляду $v''_{xx}+v''_{yy}=-u$ другого рівняння (1), введіть наступне

Diffusion Coefficient – **c** = -1

Source Term – **f** = -u

Однак, остання інструкція хибна. Нижче ми до неї повернемося і виправимо.

Задайте граничну умову для величини v.

Component 2 - Poisson's Equation 2 (poeq2) :

Physics -> **Boundaries** – Dirichlet Boundary Condition

Component 2 - Poisson's Equation 2 (poeq2) - Dirichlet Boundary Condition 1:

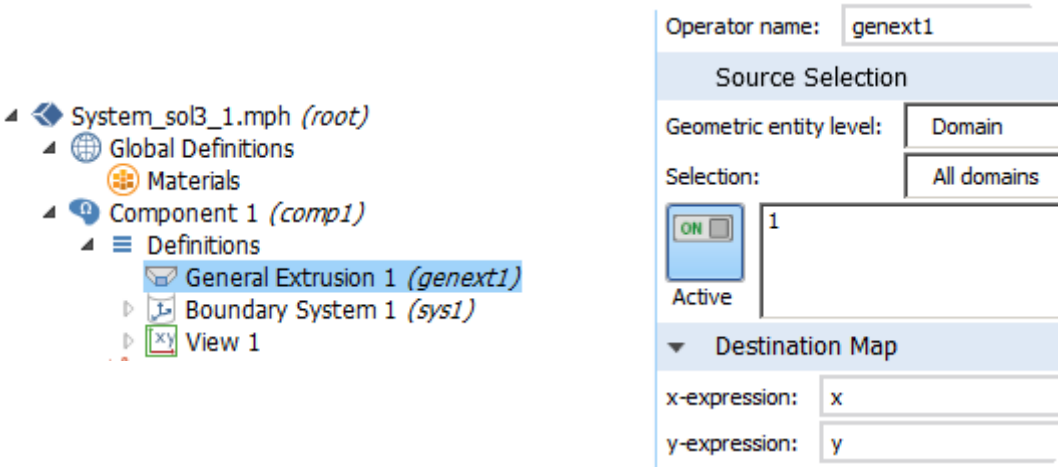
Boundary Selection– **Selection** = All boundaries (*контур кола*)

Dirichlet Boundary Condition – **r** =0 (*граничне значення v*)

Здається, що ми ввели всі необхідні дані для розв'язання задачі. Однак це не так. Справа в тому, що компоненти (гілки **Component**) дерева моделі не пов'язані. Зокрема, ім'я шуканої функції u з гілки **Component 1** недосяжне в **Component 2**. В принципі, рівняння $v''_{xx}+v''_{yy}=-u$ можна було б записати у вигляді $v''_{xx}+v''_{yy}=-\mathrm{comp1}.u$, використовуючи повне ім'я змінної $\mathrm{comp1}.u$. Повне ім'я включає префікс, і складається з імені компонента і імені функції, розділених крапкою. Однак цього недостатньо. Справа в тому, що геометрії компонентів також не пов'язані. Функція u визначена в точках (x, y) з кола в гілці **Component 1**. А в рівнянні $v''_{xx}+v''_{yy}=-\mathrm{comp1}.u$ використовуються координати (x, y) з геометрії в гілці **Component 2**. Будь-яка точка з

координатами (x, y) в розрахунковій області першого компонента ніяк не пов'язана з точкою, що має такі ж числові значення координат в другому компоненті. Для вирішення цієї проблеми в системі COMSOL Multiphysics передбачені спеціальні Component Couplings оператори. Тут ми будемо говорити тільки про один їх різновид – `General Extrusion` оператор.

Щоб зв'язати дві задачі, в тому компоненті, який буде надавати свої дані, потрібно створити `General Extrusion` оператор. В панелі властивостей створюваного вузла `General Extrusion 1(genext1)` вказується геометрія/зона, з якої оператор вибиратиме дані. В розділі **`Destination Map`** задаються формули відповідності між координатами розрахункових областей компонента джерела і компонента призначення. Назва/ім'я оператора вказується в полі **Operator name**.



Якщо `General Extrusion 1(genext1)` оператор налаштувати так, як показано на попередньому рисунку, то точки з координатами (x, y) першого компонента будуть ототожнюватися з точками (x, y) будь-якого компонента, в якому існують координати з такими позначеннями. Точніше, імена просторових координат іншого компонента, які задані в полях **x-expression** і **y-expression**, при використанні оператора будуть збігатися з координатами (x, y) в першому компоненті.

Якщо через (x_s, y_s) позначити просторові координати в розрахунковій області гілки **Component 1**, а через (x_d, y_d) позначити координати в розрахунковій області **Component 2**, то запис comp1.genext1(u) ідентичний математичному запису $u(x_s(x_d, y_d), y_s(x_d, y_d))$. При цьому в поточному прикладі $x_s = x_d$, $y_s = y_d$.

Зауваження. Коли зв'язок між координатами джерела (першого компонента) і області дії (другого компонента) має вигляд $x_s = X(x_d, y_d)$, $y_s = Y(x_d, y_d)$, то в полях **x-expression** і **y-expression** записуються вирази $X(x, y)$ і $Y(x, y)$, в яких розуміється, що x і y змінюються в області призначення/дії.

Після створення оператора `genext1`, у виразах другого компонента можна використовувати цей оператора з аргументом, обчисленим в першому компоненті. При чому бажано оператор використовувати з префіксом – іменем

моделі. Таким чином, рівняння в другому компоненті потрібно писати у вигляді $v''_{xx}+v''_{yy}=-\text{comp1.genext1(comp1.u)}$ або простіше $v''_{xx}+v''_{yy}=-\text{comp1.genext1(u)}$. Цим другий компонент повідомляється, що в правій частині рівняння стоїть функція u, яка визначається в першому компоненті в ідентичних точках.

Таким чином, в нашу модель слід внести дві зміни.

1. В гілці **Component 1 - Definitions** потрібно створити вузол General Extrusion оператора.

Component 1 :> Definitions > Component Couplings - General Extrusion

Component 1 - Definitions - General Extrusion 1:

Source Selection – **Selection,** <> =Manual,1 (*індикатор кола*)

Destination Map – **x-expression** = x (*відповідність координат*)

Destination Map – **y-expression** = y (*відповідність координат*)

2. В **Component 2** потрібно змінити вираз в правій частині рівняння Пуассона.

Poisson's Equation 2 (poeq2) - Poisson's Equation 1:

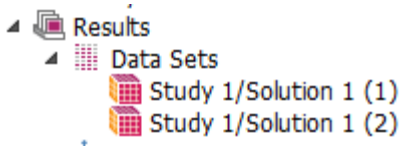
Source Term – **f** = -comp1.genext1(u)

Тепер можна виконати обчислення.

Home **-> Study** - Compute

Після їх завершення в графічній зоні ви побачите кругову область з функціональним забарвленням, відповідним розв'язку u. Це зображення належить вузлу **2D Plot Group 1**. В гілці **Results** також буде створений вузол **2D Plot Group 2**, який містить зображення функції v.

Зверніть увагу на те, що в гілці **Results - Data Sets** створено два вузла/розв'язку - по одному для кожного компонента.



При цьому в ПВ графічної групи **2D Plot Group 1** в полі **Data** – **Data set** використовується розв'язок Study 1/Solution 1(1). А група **2D Plot Group 2** в полі **Data** – **Data set** використовує розв'язок Study 1/Solution 1(2). В першому вузлі ДД Study 1/Solution 1(1) містяться значення функції u (вона визначена в першому компоненті), а в другому ДД - значення функції v.

Перевіримо обчислення. Для цього побудуємо графіки похибок для обох розв'язків u і v.

Results -> 2D Plot Group

2D Plot Group 3: Data – **Data set** = Study 1/Solution 1 (1)

2D Plot Group 3: RMB > Surface

2D Plot Group 3 – Surface 1:

Expression – **Expression** = abs(u-(1-x^2-y^2)/4)

LMB > Plot (*наступний рисунок ліворуч*)

Аналогічно побудуйте графік похибки для функції v.

Results -> 2D Plot Group

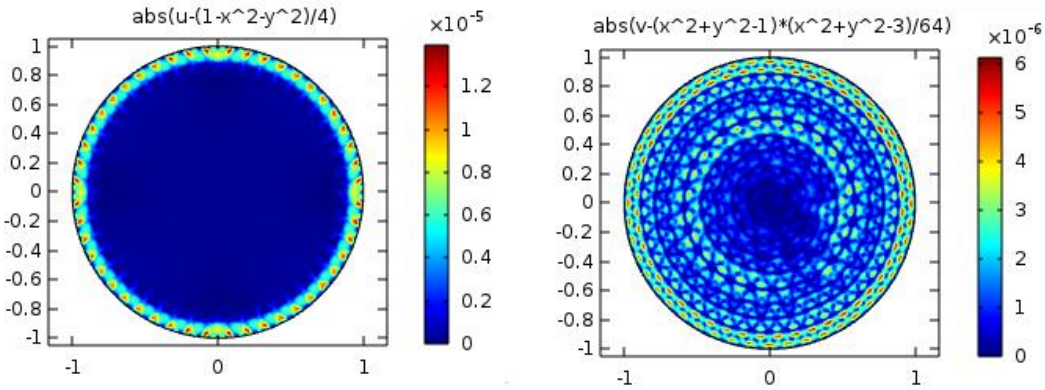
2D Plot Group 4: Data – **Data set** = Study 1/Solution 1 (2)

2D Plot Group 4: RMB > Surface

2D Plot Group 4 – Surface 1:

Expression – **Expression** = `abs(v-(x^2+y^2-1)*(x^2+y^2-3)/64)`

LMB > Plot (*наступний рисунок праворуч*)



За шкалою кольорів можна оцінити значення похибок в різних частинах кола.

Для тренування подивимося, як побудовані в різних компонентах розв'язки, використовувати в одному виразі. Наприклад, побудуємо графік суми `u+v`.

Створіть нову групу двовимірних графіків.

`Results` -> 2D Plot Group

2D Plot Group 5: **Data** – **Data set** = `Study 1/Solution 1 (2)`

2D Plot Group 5: RMB > Surface

2D Plot Group 5 – Surface 1:

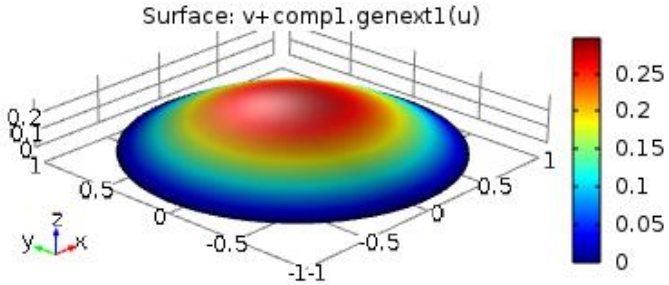
Expression – **Expression** = `v+comp1.genext1(u)`

RMB> Height Expression

2D Plot Group 5 – Surface 1 - Height Expression 1:

Axis – **Scale factor** = `True ,1`

LMB > Plot



Як бачите, оператор загальної екструзії можна використовувати і при обробці результатів обчислень. ■

Зауваження. Ми навели три розв'язки однієї задачі. Якщо фізичні процеси розгортаються в одній і тій же розрахункової області, то перші два способи здаються кращими за третій. Однак третій спосіб дозволяє розглядати задачі з різною геометрією розрахункових областей. Наприклад, в нашому прикладі ми могли б перше рівняння розв'язувати в області квадрата. Потрібно лише, щоб формули перетворення $x_s = x_s(x_d, y_d)$, $y_s = y_s(x_d, y_d)$, які використовуються в виразі $u(x_s(x_d, y_d), y_s(x_d, y_d))$, були законними для всіх точок (x_d, y_d) розрахункової області другого компонента (нагадаємо, що (x_s, y_s) позначають просторові координати в розрахунковій області першого компонента).

3.24. Згин круглої плити, затисненої за контуром.

Диференціальне рівняння згину плити має вигляд [10]

$$\Delta\Delta u = \frac{1}{D} q(x, y), \tag{1}$$

де D – циліндрична жорсткість, $q(x, y)$ – навантаження на одиницю площі плити, $u(x, y)$ - зміщення точок серединної поверхні плити, $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ – оператор Лапласа.

В полярних координатах r, φ диференціальне рівняння (1) має вигляд

$$\left(\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{1}{r^2}\frac{\partial^2}{\partial \varphi^2}\right)\left(\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u}{\partial \varphi^2}\right) = \frac{1}{D} q(r, \varphi). \tag{2}$$

У випадку круглої плити з вісесиметричною навантагою $q = q(r)$ диференціальне рівняння набуває вигляду

$$\left(\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r}\right)\left(\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r}\right) = \frac{1}{D} q(r). \tag{3}$$

Тут ми розглянемо задачу згину суцільної круглої плити Ω радіуса R, зовнішній край якої затиснутий. Тоді на ньому повинні задовольнятися умови $u\big|_{\partial\Omega} = 0$ і $u'_r\big|_{\partial\Omega} = 0$.

Якщо ввести позначення $m = -\left(\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r}\right)$ і покласти для спрощення D=1, R=1, то задачу можна записати у вигляді системи:

$$\begin{cases} \frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} = -m(r) \\ \frac{\partial^2 m}{\partial r^2} + \frac{1}{r}\frac{\partial m}{\partial r} = -q(r) \end{cases} \tag{4}$$

з граничними умовами $u(1) = 0, u'(1) = 0$. При чому в центрі (при $r = 0$) повинні виконуватися умови симетрії $u'(0) = 0, m'(0) = 0$.

В COMSOL для такої задачі можна побудувати `1D Axisymmetric` шаблон. Але система не зможе її розв'язати, бо для шуканої величини $u(r)$ ми маємо одну зайву умову, а для $m(r)$ однієї умови не вистачає. До того ж вона повинна бути умовою Діріхле, оскільки в друге рівняння (4) входять лише похідні $m(r)$ і, якщо на границях будуть відомі тільки похідні, то значення шуканої величини $m(r)$ буде відомо лише з точністю до константи. Щоб цього уникнути, потрібно знати значення $m(r)$ хоча б в одній точці. Але цього немає.

Отже задачу треба модифікувати. Залишимо граничну умову $u(1) = 0$, а для `m` задамо граничну умову $m(1) = p$, ввівши додаткову невідому p – значення

$m(r)$ на контурі кола (правому кінці відрізка обертання, що представляє розрахункову область). Тоді потрібно підібрати p так, щоб умова $u'(1)=0$ виконувалась. Наступний приклад продемонструє описаний підхід.

Спочатку треба переписати систему (4) у вигляді, який потрібен COMSOL.

$$\frac{\partial}{\partial r}\left(\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}\frac{\partial}{\partial r}\begin{pmatrix}u\\ m\end{pmatrix}\right)+\begin{pmatrix}1/r & 0\\ 0 & 1/r\end{pmatrix}\frac{\partial}{\partial r}\begin{pmatrix}u\\ m\end{pmatrix}=\begin{pmatrix}-m\\ -q\end{pmatrix}. \qquad (5)$$

Загальний вигляд коефіцієнтної форми наступний

$$e_a\frac{\partial^2\boldsymbol{u}}{\partial t^2}+d_a\frac{\partial\boldsymbol{u}}{\partial t}+\nabla\cdot(-c\nabla\boldsymbol{u}-\alpha\boldsymbol{u}+\gamma)+\beta\cdot\nabla\boldsymbol{u}+a\boldsymbol{u}=f\,, \qquad (6)$$

де $\boldsymbol{u}=\begin{pmatrix}u\\ m\end{pmatrix}$ – шукана вектор-функция і $\nabla=\frac{\partial}{\partial r}$. Щоб (6) набуло вигляду (5) слід покласти

$$c=\begin{pmatrix}-1 & 0\\ 0 & -1\end{pmatrix} \qquad \beta=\begin{pmatrix}1/r & 0\\ 0 & 1/r\end{pmatrix} \qquad f=\begin{pmatrix}-m\\ -q\end{pmatrix}$$

Інші тензорні/матричні коефіцієнти $e_a, d_a, a, \alpha, \gamma$ слід покласти рівними нулю.

Приклад. Для розв'язання систем ДРЧП призначений математичний інтерфейс `Coefficient Form PDE`.

File->New->Model Wizard->2D->

Mathematics– PDE Intefaces – Coefficient Form PDE (c) **->**

Number of dependent variables: = `2` +

Dependent variables: =`[u,m]` **->** Stationary

Нагадаємо, що тут символ **+** (плюс) позначає виконання додаткового вводу в поточному вікні/панелі в полі, назва якого стоїть після знака **+**.

Вимкніть в моделі використання розмірностей фізичних величин.

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

Створіть функцію $q(r)$ зовнішнього навантаження.

Global Definitions: RMB > Functions - Analytic

Function name = `q`

`Definition` – **Expression** = `1`

Геометрія моделі складатиметься з відрізка [0, 1].

Geometry 1:> `Geometry->`**`Primitives->`** Interval

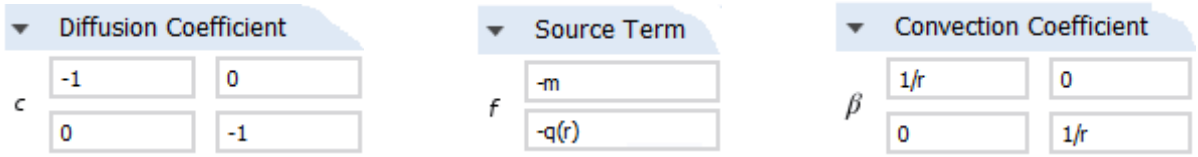
LMB > Build All Objects

На створеному відрізку генеруйте СкЕ сітку з налаштуваннями за замовчуванням.

Mesh 1 : LMB> Build All

В дереві моделі виділіть вузол **Coefficient Form PDE 1**. Справа відкриється його панель. В розділі **`Equation`** цієї панелі наведено загальний вигляд (6) коефіцієнтної форми. Щоб вона набула бажаного виду, введіть

$$c = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \qquad f = \begin{pmatrix} -m \\ -q(r) \end{pmatrix} \qquad \beta = \begin{pmatrix} 1/r & 0 \\ 0 & 1/r \end{pmatrix}$$



Інші коефіцієнти $e_a, d_a, a, \alpha, \gamma$ слід покласти рівними нулю. Перевірте також, що в поточній панелі **Coefficient Form PDE 1** в секції вибору розділу **`Domain Selection`** стоїть 1, тобто обраний відрізок.

Перейдемо до задання граничних умов. Припустимо, що на зовнішній межі $(r = 1)$ шукана функція m відома і $m(1) = p$. Насправді p не задана і ми вважатимемо її додатковою невідомою задачі. При чому ця змінна повинна бути «прив'язана» до точки $r = 1$.

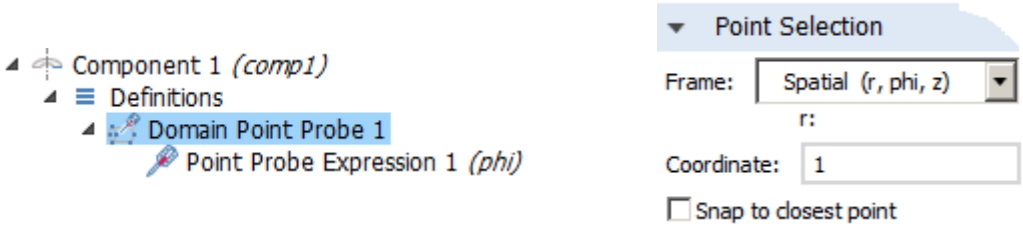
В COMSOL скалярну величину в точці можна реалізувати за допомогою зондової змінної. Існують різні типи зондів. Ми використаємо `domain point` зонд, який схожий на давач, що контролює фізичний параметр в певному місці моделі, і являє змінну, що містить значення «полевої» величини в точці.

Отже в нашій моделі потрібен зонд, який буде відслідковувати нахил поверхні на межі кола, точніше значення $u_r'(1)$. Створимо його.

Component 1 (comp1) – Definitions : RMB: Probes – Domain Point Probe

Component 1 – Definitions - Domain Point Probe 1:

`Point Selection` – **Coordinate** = 1 (*координата точки розташування зонда*)



Definitions - Domain Point Probe 1 – Point Probe Expression 1 (ppb1):

Variable name = `phi`

`Expression` – **Expression** = `ur` (похідна величини *u* по *r*)

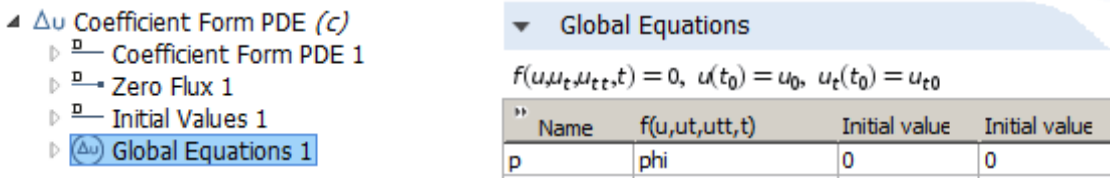
Додайте вузол `Global Equations`, в якому вкажіть невідому *p*, а в якості рівняння напишіть ім'я зондової змінної `phi`.

`Physics` **`-> Global`** – Global Equations

Coefficient Form PDE (c) - Global Equations 1:

`Global Equations` – **Name** = `p`

`Global Equations` – **f(u,ut,utt,t)(1)** = `phi`



Розуміти запис в таблиці потрібно наступним чином. Задача буде розв'язуватися відносно невідомої *p* (граничне значення `m`) одночасно з величинами `u` та `m`, доки вираз в комірці **f(u,ut,utt,t)** не буде дорівнювати

нулю, тобто доки зондова змінна phi (а вона співпадає зі значенням $u_r'(1)$) не стане нулем.

Тепер можна додавати в модель граничну умову Діріхле $u(1)=0$, $m(1)=p$.

Physics -> **Boundaries** – Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** , <> = Manual,2 (*правий кінець*)

Dirichlet Boundary Condition–**r1, r2**=0,p (*граничні значення u і m*)

Задавати граничну умову в центрі (в точці $r=0$) непотрібно.

Таким чином, в нашій задачі створено три шукані величини: u,m та p. При чому u та m – це полеві змінні (функції від r), а p – невідомий скаляр (число).

Виконайте обчислення.

Home -> **Study** - Compute

Після їх завершення в графічній зоні ви побачите одновимірний графік величини u, який належить вузлу **2D Plot Group 1 - Line Graph 1**.

У розглянутої задачі існує аналітичний розв'язок

$$u=\frac{1}{64}(1-r^2)^2 \ , \quad m=\frac{1}{8}(1-2r^2).$$

Його можна порівняти з отриманим наближеним розв'язком.

1D Plot Group 1 -> **Add Plot** – Line Graph

1D Plot Group 1 - Line Graph 2:

Selection – **Selection** ,< > =Manual,1 (індикатор відрізка [0,1])

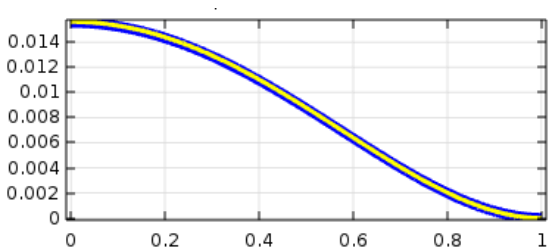
y-Axis Data – **Expression** = (1-r^2)^2/64

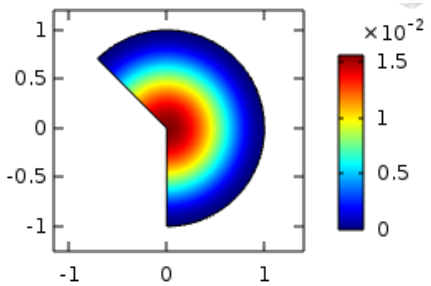
x-Axis Data – **Parameter** = Expression

x-Axis Data – **Expression** = r

Coloring and Style – **Color, Width** = Yellow, 2 (*два поля*)

LMB > Plot (*наступний рисунок ліворуч*)





Як бачите, аналітичний і наближений розв'язки співпадають.

В гілці **Results** також створено вузол **2D Plot Group 2**, який містить кольоровий графік поверхні обертання функції u (попередній рисунок праворуч).

В гілці **Probe Plot Group 3** також створено графік зонда. Оскільки у нас модель статична, то він є просто точкою з майже нульовою ординатою, значення якої можна дізнатися з автоматично створеної таблиці **Probe Table 1**.

Gradient of u, r component, Point Probe Expression 1
6.5052E-18

Побудуйте графік другої шуканої величини m і відповідного аналітичного виразу.

`Results` -> 1D Plot Group

1D Plot Group 4: `Data` – Data set = `Study 1/Solution 1`

1D Plot Group 4: RMB > Line Graph

1D Plot Group 4 – Line Graph 1:

`Selection` – < > = `1` (*індикатор відрізка* [0,1])

`y-Axis Data` – Expression = `m`

`x-Axis Data` – Parameter = `Expression`

`x-Axis Data` – Expression = `r`

`Coloring and Style` – Color, Width = `Blue, 4` (*два поля*)

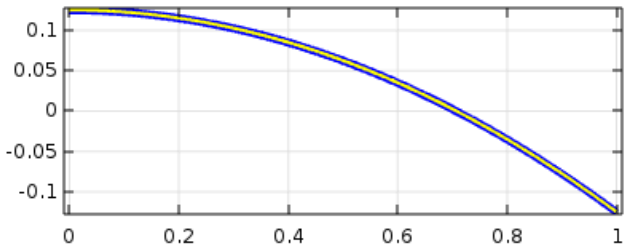
1D Plot Group 4 – Line Graph 1: RMB > Duplicate

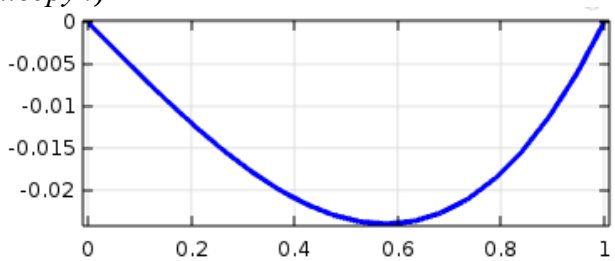
1D Plot Group 4 – Line Graph 2:

`y-Axis Data` – Expression = `(1-2*r^2)/8`

`Coloring and Style` – Color, Width = `Yellow, 2` (*два поля*)

LMB > Plot (*наступний рисунок ліворуч*)





Побудуйте (самостійно) графік похідної u_r' (попередній рисунок праворуч). Він буде належати графічній групі **1D Plot Group 5**.

Побудуйте графік поверхні плити. Спочатку створіть джерело даних типу `Revolution 1D` таке, щоб воно представляло повне коло.

Results – Data Sets - Revolution 1D 1: RMB > Duplicate

Data Sets - Revolution 1D 2:

`Revolution Layers` – Start angle, Revolution angle = `0, 360`

Тепер можна будувати графік поверхні.

`Results` -> 2D Plot Group

2D Plot Group 6: `Data` – Data set = `Revolution 1D 2`

2D Plot Group 6: RMB > Surface

2D Plot Group 6 – Surface 1:

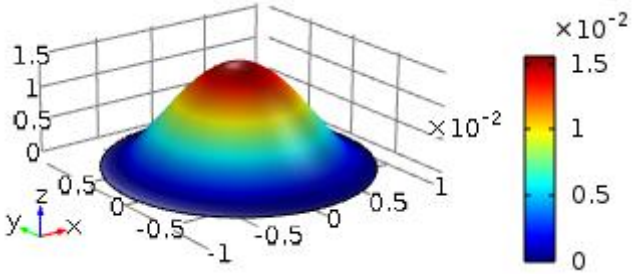
`Expression` – Expression = `u`

RMB> Height Expression

2D Plot Group 5 – Surface 1 - Height Expression 1:

`Axis` – Scale factor = `True ,50`

LMB > Plot



Ви можете самостійно модифікувати зовнішнє навантаження, змінивши функцію $q(r)$, і переобчисливши модель.

Зауваження. Для задачі вісесиметричного прогину плити з вільно опертим краєм система рівнянь (4) залишається незмінною, але граничні умови набувають вигляду $u(1)=0, m(1)=0$, і введення додаткової невідомої непотрібно.

3.25. Система нелінійних параболічних рівнянь.

Розв'яжемо на відрізку [0,1] нелінійну параболічну систему ДРЧП

$$\begin{aligned} \frac{\partial u_1}{\partial t} &= \frac{\partial^2 u_1}{\partial x^2} + u_1 \cdot (1 - u_1 - u_2) \\ \frac{\partial u_2}{\partial t} &= \frac{\partial^2 u_2}{\partial x^2} + u_2 \cdot (1 - u_1 - u_2) \end{aligned}, \qquad (1)$$

з граничними умовами

$$u'_{1x}(t,0)=0, u_2(t,0)=0,$$
$$u_1(t,1)=1, u'_{2x}(t,1)=0.$$

і початковими умовами

$$u_1(0,x)=x^2, u_2(0,x)=x(x-2).$$

Особливість в COMSOL Multiphysics граничних умов Діріхле для систем рівнянь полягає в тому, що вони потребують одночасного задання значень обох шуканих величин на межі області, а в нашій задачі це не так. В точці $x=0$ для першої шуканої величини u_1 відома похідна $u'_1(0)$, а для другої u_2 – її значення $u_2(0)$. В точці $x=1$, навпаки, відома перша функція $u_1(1)$, а для другої задано похідну $u'_2(1)$. Подолати ці труднощі допомагає гранична умова типу `Constraint`, яка більш гнучка, і дозволяє для системи задати два різні вирази на одній граничній ділянці.

Приклад. Для розв'язання систем ДРЧП призначений математичний інтерфейс `Coefficient Form PDE`.

File->New->Model Wizard->1D ->
Mathematics– PDE Intefaces – Coefficient Form PDE (c) ->
Number of dependent variables: = 2 +
Dependent variables: =[u1,u2] -> Time Dependent

Нагадаємо, що тут символ **+** (плюс) позначає виконання додаткового вводу в поточному вікні/панелі в полі, вказаному після знака **+**.

Вимкніть в моделі використання розмірностей фізичних величин.

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

Геометрія моделі складатиметься з відрізка [0, 1].

Geometry 1:> Geometry->**Primitives->** Interval
LMB > Build All Objects

На створеному відрізку генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** = Finer

LMB> Build All

Система (1) в матричній формі має вигляд

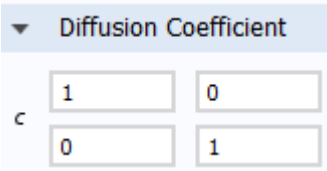
$$\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}\frac{\partial}{\partial t}\begin{pmatrix}u_1\\ u_2\end{pmatrix}+\frac{\partial}{\partial x}\left(-\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}\frac{\partial}{\partial x}\begin{pmatrix}u_1\\ u_2\end{pmatrix}\right)=\begin{pmatrix}u_1\cdot(1-u_1-u_2)\\ u_2\cdot(1-u_1-u_2)\end{pmatrix}.$$

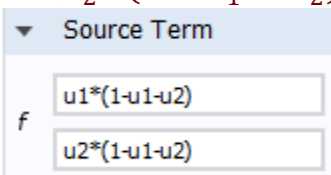
Щоб загальний вид коефіцієнтної форми

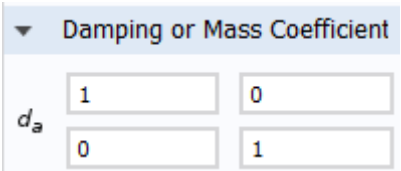
$$e_a\frac{\partial^2\boldsymbol{u}}{\partial t^2}+d_a\frac{\partial\boldsymbol{u}}{\partial t}+\nabla\cdot(-c\nabla\boldsymbol{u}-\alpha\boldsymbol{u}+\gamma)+\beta\cdot\nabla\boldsymbol{u}+a\boldsymbol{u}=f\text{, де }\boldsymbol{u}=[u_1,u_2]^T,\nabla=\partial/\partial x,$$

набув потрібного вигляду слід покласти

$$c=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}\qquad f=\begin{pmatrix}u_1\cdot(1-u_1-u_2)\\ u_2\cdot(1-u_1-u_2)\end{pmatrix}\qquad d_a=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$$

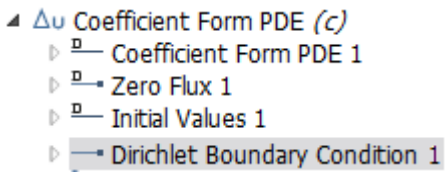


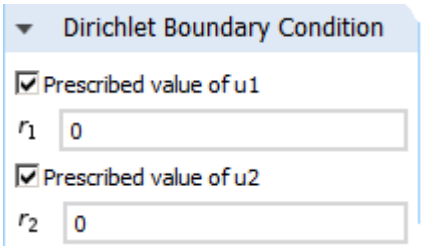




Інші матричні/векторні коефіцієнти $e_a, a, \alpha, \beta, \gamma$ слід покласти рівними нулю.

Перейдемо до задання граничних умов. Якщо спробувати використати граничні умови Діріхле, наприклад, для лівого кінця, то в ПВ вузла Dirichlet Boundary Condition 1 ми отримаємо наступний шаблон вводу граничних значень:





В розділі **Dirichlet Boundary Condition** потрібно одночасно задавати значення обох шуканих величин. Але наші граничні умови сформульовані інакше. Тому їх вводити будемо за допомогою вузлів типу Constraint (обмеження). Гранична умова Constraint обмежує розв'язок на обраній ділянці геометрії, використовуючи вираз R, який повинен обертатися там на нуль (R=0). Нагадаємо, що використання вузлів типу Constraint вмикається в налаштуваннях майстра моделей опцією **Advanced Physics Options** в списку, який розкривається кнопкою Show .

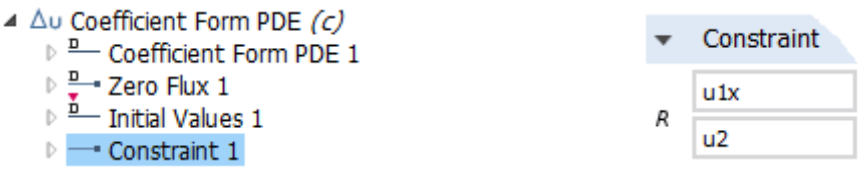
Якщо ви створили вузол **Dirichlet Boundary Condition 1**, то видаліть його. Перевірте, що опція **Advanced Physics Options** увімкнена і виконайте наступні інструкції

Physics -> **Boundaries** – Constraint

Constraint 1:

Boundary Selection – **Selection,** <> = Manual,1 (точка x=0)

Constraint – **R** = u1x / u2



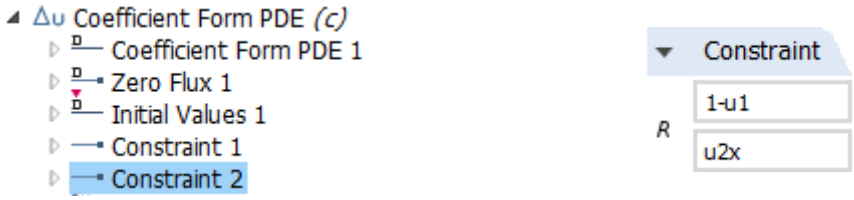
Оскільки вираз R повинен обертатися на нуль, то в точці $x=0$ задаються граничні умови $u'_{1x}(t,0)=0$, $u_2(t,0)=0$. Аналогічно задається гранична умова на правому кінці.

`Physics` -> **`Boundaries`** – Constraint

Constraint 2:

`Boundary Selection` – **Selection,** <> = `Manual,2` (точка x=1)

`Constraint` – **R** = $\begin{matrix} 1-\mathrm{u1} \\ \mathrm{u2x} \end{matrix}$



Цим задаються граничні умови $u_1(t,1)=1$, $u'_{2x}(t,1)=0$.

Задайте початкові умови.

Initial Values 1:

`Initial Values` – **`Initial Value for u1:`** – **u1** = `x^2`

`Initial Values` – **`Initial Value for u2:`** – **u2** = `x*(x-2)`

Налаштуйте розв'язувач, задавши часовий інтервал і похибку обчислень.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times** = `range(0,0.05,1)`

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

Виконайте обчислення.

`Home` -> **`Study`** - Compute

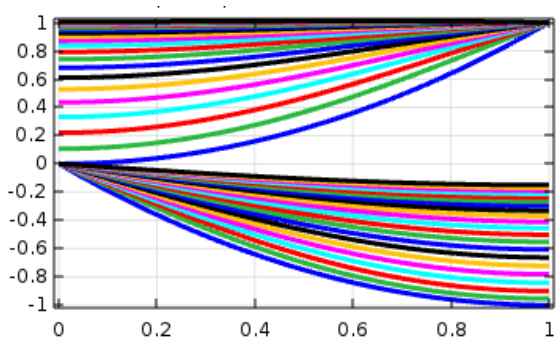
Після їх завершення в графічній зоні вузла **1D Plot Group 1 - Line Graph 1** буде побудовано графіки розв'язку `u1` для всіх моментів часу з послідовності `range(0,0.05,1)`. Додайте в групу **1D Plot Group 1** графіки другої шуканої величини `u2`.

1D Plot Group 1 – Line Graph 1 : RMB> Duplicate

1D Plot Group 1 – Line Graph 2:

`y-Axis Data` – **Expression** = `u2`

LMB > Plot



Тут верхній жмуток кривих зображає графіки величини u1, а нижній – u2. Щоб побачити графіки в конкретний момент часу, дублюйте створену графічну групу, скорегуйте іі налаштування, і задайте момент часу.

1D Plot Group 1 : RMB> Duplicate

1D Plot Group 2 :

Data –Time selection = From list

Data –Times = 0.2

1D Plot Group 2 - Line Graph 1:

Legends–Show legends = True

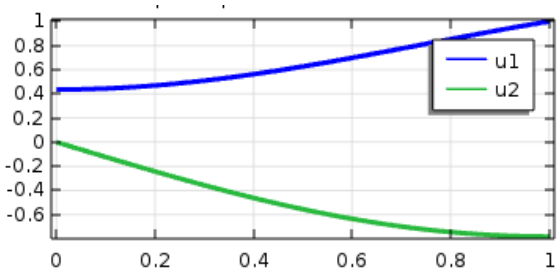
Legends– Legends, <> = Manual,u1

1D Plot Group 2 - Line Graph 2:

Legends–Show legends = True

Legends– Legends, <> = Manual,u2

LMB > Plot



Самостійно побудуйте анімацію розв’язків.

3.26. Рівняння Пуассона в генеральній формі.

Загальна (генеральна) форма має вигляд

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \Gamma = F\,. \qquad (1)$$

Тут Γ може бути векторною або матричною величиною, що залежать від шуканих функцій і їх просторових похідних, а F - скаляром або вектором. Коефіцієнта форма може бути отримана з (1) спеціальним вибором Γ і F. Наприклад, щоб (1) набуло вигляду двовимірного рівняння Пуассона $\nabla \cdot \nabla u = 1$ потрібно покласти $e_a = 0, d_a = 0$, $\Gamma = \nabla u = [u'_x, u'_y]$ і $F = 1$. Форму (1) зручно використовувати для нелінійних рівнянь, коли коефіцієнтна форма непридатна.

При розв’язанні одного рівняння (1) коефіцієнт F є скаляром, а терм Γ – вектором, як правило залежним від просторових координат і розв’язку u. При застосуванні генеральної форми граничні умови також використовують терм Γ, який зазвичай є вектором потоку «збережуваної» величини (тепла, заряду, маси, моменту). На межі тіла цей потік може дорівнювати нулю або будь-якому значенню. Звичайно, що замість потоку на границі може задаватися значення шуканої функції u або який-небудь вираз, залежний від u і її похідних.

Тут ми продемонструємо використання генеральної форми на прикладі розв’язання двовимірного рівняння Пуассона.

Приклад. Створіть шаблон моделі послідовністю інструкцій.

File->New->Model Wizard->2D->

Mathematics – PDE Interfaces – GeneralForm PDE(g)->Stationary

Вимкніть в моделі підтримку розмірностей фізичних величин.

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

Створіть геометрію моделі. Накресліть прямокутник.

Geometry 1 :> Geometry->**Draw->Rectangle** – Rectangle(Center)

Помістіть покажчик миші в точку з координатами (0,0), і при натиснутій лівій кнопці перемістить його в точку (2,1). Буде накреслено прямокутник з центром в точці (0,0), шириною 4, висотою 2, і його сторони будуть паралельні координатним осям. Перевірте ці значення в ПВ створеного вузла **Rectangle 1** і скорегуйте їх, якщо щось не так. Завершіть побудову прямокутника натисканням на кнопку Build Selected.

Намалюйте коло.

Geometry 1 :> Geometry->**Draw**->Circle

Помістіть покажчик миші в точку з координатами (0,0). Утримуючи ліву кнопку, переміщайте покажчик миші від центру доки радіус кола не стане рівним 0.5. Відпускання кнопки миші призведе до створення кола з центром в точці (0,0) і радіусом 0.5. Перевірте, що коло побудовано із заданими параметрами. Для цього активуйте створений вузол **Circle 1 (c1)** та в його ПВ перевірте налаштування (координати центру кола і його радіус). Якщо щось не так, то виправте значення.

Виконайте вирізання кола з прямокутника.

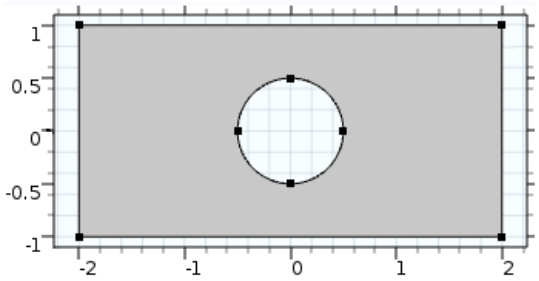
Geometry **-> Booleans and Partitions** – Difference

Component 1 - Geometry 1 - Difference 1:

Difference – **Object to add** = r1 (*прямокутник*)

Difference – **Object to subtract** = c1 (*коло*)

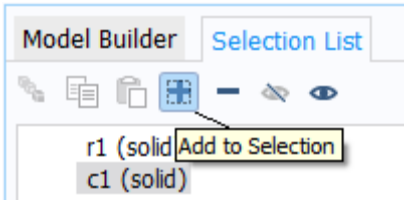
LMB > Build Selected



Зауваження. Інколи виникають складнощі при виборі об’єкта, що видаляється, бо кілька фігур може бути розташовано одне над одним. Щоб обрати одну з таких перекривних фігур, розмістіть покажчик миші над спільною зоною і крутніть коліщатко миші. Цим ви змістите фокус на наступний об'єкт зі «стопки» фігур, що перекриваються. Коли виділиться потрібний об'єкт, клацніть по ньому мишею.

Крім того, вибрати «закритий» об'єкт можна також на закладці Selection List, яка зазвичай розташована за панеллю дерева моделей. Якщо її там немає, то для відображення виконайте команду

Home **-> Layout** -> Windows - Selection List



Потрібно виділити ім'я необхідного об'єкта і виконати команду Add to Selection.

На лівій стороні прямокутника зробимо фаски, а на правій - скруглимо кути.

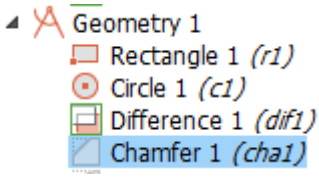
Geometry 1 :> Geometry-> **Operations**-> Chamfer

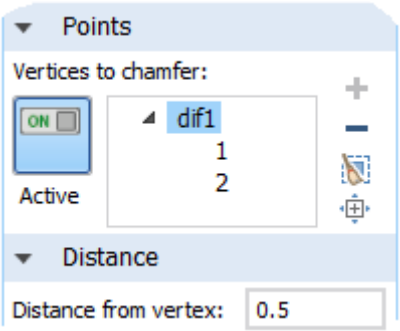
В ДМ з'явиться вузол **Chamfer 1 (cha1)**. В його ПВ в розділі **Points** натисніть кнопку Active. Перемістіть курсор миші в графічну область, і клацніть по черзі по лівим вершин прямокутника. Номери вершин (1 і 2) з'являться в списку **Vertices to chamfer.** В розділі **Distance** в поле **Distance from vertex** введіть значення 0.5 і клацніть мишею по кнопці Build Selected, завершуючи створення фасок. Ці дії кратко позначаються наступним чином.

Geometry 1 - Chamfer 1 (cha1):

Points – **Vertices to chamfer** = 1, 2

Distance – **Distance from vertex** = 0.5





LMB > Build Selected

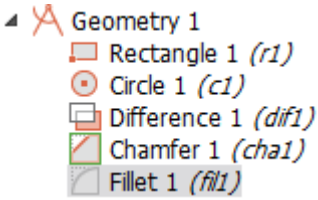
Для додавання скруглень виконайте наступні інструкції.

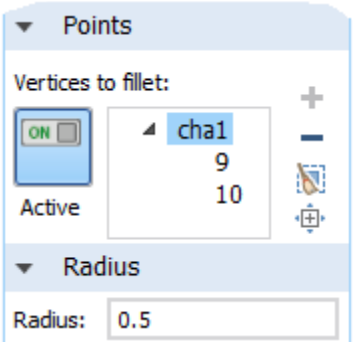
Geometry **-> Operations** -> Fillet

Geometry 1 - Fillet 1(fil1):

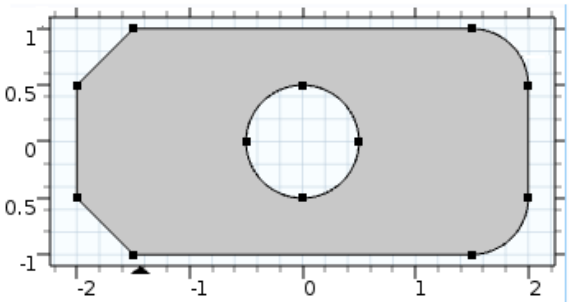
Points – **Vertices to Fillet** = 9, 10

Radius – **Radius** = 0.5





LMB > Build All Objects



Генеруйте СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** = Finer

LMB> Build All

Відкрийте ПВ вузла **General Form PDE (g) - General Form PDE 1**. В розділі **Equation** цієї панелі наведено рівняння (1)

$$e_a \frac{\partial^2 u}{\partial t^2} + d_a \frac{\partial u}{\partial t} + \nabla \cdot \Gamma = F \text{, де } \nabla = \left[\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right].$$

Щоб воно набуло виду рівняння Пуассона необхідно ввести наступні значення.

Conservative Flux – $\Gamma = \begin{matrix} -ux \\ -uy \end{matrix}$

Source Term – **f** = 1

Damping or Mass Coefficient – $\mathbf{d_a}$ = 0

Mass Coefficient – $\mathbf{e_a}$ = 1

Перевірте також, що в поточній панелі (General Form PDE) в розділі **Domain Selection** стоїть 1, тобто обрана сконструйована нами область.

Почніть задавати граничні умови.

Physics->**Boundaries**->Dirichlet Boundary Condition

Dirichlet Boundary Condition 1:

Boundary Selection – **Selection** , <> = Manual,7,8,9,10

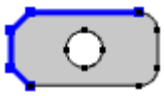
Dirichlet Boundary Condition – **r** = 3*(1-x^2+y^2)

Physics->**Boundaries**->Dirichlet Boundary Condition

Dirichlet Boundary Condition 2:

Boundary Selection – **Selection** , <> = Manual,1,2,3,5

Dirichlet Boundary Condition – **r** = 0



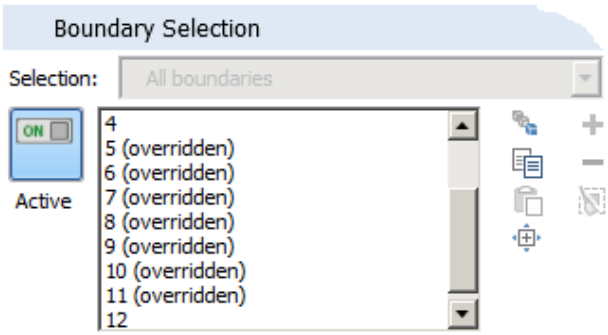
Physics->**Boundaries**->Dirichlet Boundary Condition

Dirichlet Boundary Condition 3:

Boundary Selection – **Selection** , <> = Manual,6,11

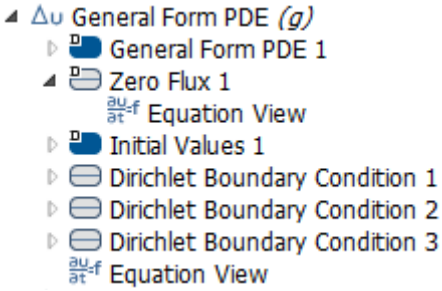
Dirichlet Boundary Condition – **r** = 2

Зайдіть в ПВ вузла **Zero Flux 1** і переконайтеся, що для всіх ділянок межі, крім двох з номерами 4 і 12, граничні умови перевантажені.



На нижньому боці і правому верхньому скругленні гранична умова $-\mathbf{n}\cdot\Gamma=0$, яка задається за замовчуванням, визначає нульовий потік по нормалі $\mathbf{n}$ до межі. При цьому вираз для терма Γ береться з ПВ вузла **General Form PDE 1**, тобто $\Gamma=[-u'_x, -u'_y]$.

В результаті гілка **General Form PDE (g)** дерева моделі набуває наступного вигляду

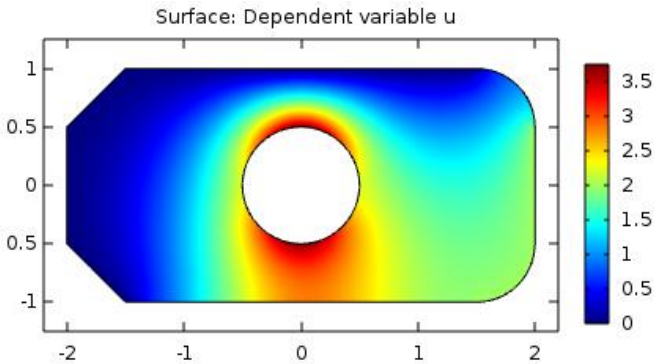


Як бачите, для кожної ділянки межі, на якій задається окрема гранична умова, створюється власний вузол.

Тепер можна виконувати обчислення.

Home -> **Study** - Compute

Після їх завершення в графічній зоні ви побачите розрахункову область з функціональним забарвленням, відповідним значенням розв'язку u.



3.27. Рівняння Кортевега – де Фріза.

Рівняння Кортевега – де Фріза [7]

$$\frac{\partial u}{\partial t}+6u\frac{\partial u}{\partial x}+\frac{\partial^3 u}{\partial x^3}=0 \qquad (1)$$

являє собою модель розповсюдження хвиль на поверхні мілкого каналу. Відповідною зміною масштабу незалежної і залежної змінних перед кожним з трьох членів в рівнянні (1) можна поставити довільні сталі коефіцієнти.

COMSOL Multiphysics не працює з третіми похідними. Тому для дослідження цього рівняння задача потребує модифікації за допомогою введення допоміжної функції $v = u''_{xx}$. Тоді замість рівняння (1) можна розв'язувати систему рівнянь

$$\begin{cases} \dfrac{\partial u}{\partial t} + 6u\dfrac{\partial u}{\partial x} + \dfrac{\partial v}{\partial x} = 0 \\ \dfrac{\partial^2 u}{\partial x^2} - v = 0 \end{cases}. \qquad (2)$$

Припускається, що хвилі розповсюджуються в необмеженому одновимірному середовищі. В Comsol ми не можемо розглядати задачу в нескінченній області, але можемо змоделювати підходящі граничні умови так, щоб на скінченному відрізку розв'язок не відчував вплив кінців. Це можна зробити кількома способами. Один з них полягає в заданні періодичних граничних умов.

Для рівняння (1) відомі розв'язки у вигляді усамітнених хвиль незмінної форми. Щоб їх отримати потрібно задати спеціальні початкові умови. Тут ми використаємо початковий профіль хвилі у вигляді функції $6\,\mathrm{sech}^2(x)$. Якщо розглядати довільні початкові профілі, то відокремлені хвилі не утворюються.

Зауваження. Розглядуваний приклад можна знайти в навчальній моделі, опис якої відкривається послідовністю інструкцій

File -> Help->Documentation ->Application Library –
COMSOL Multiphysics – Equation Based – The KdV Equation and Solitons

Приклад. Створіть шаблон моделі послідовністю інструкцій.

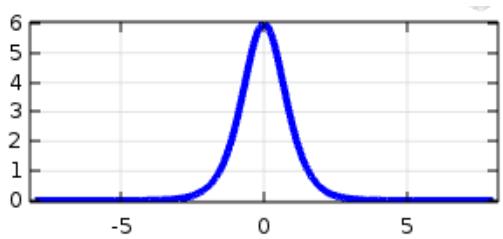
File–>New–>Model Wizard–>1D–>
Mathematics – PDE Interfaces – GeneralForm PDE(g)–>
Number of dependent variables: = 2 +
Dependent variables: =[u,v] **->** Time Dependent

Зверніть увагу, що шукані величини ми позначили ідентифікаторами u і v.
Вимкніть в моделі використання розмірностей фізичних величин.

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

Задача потребує задання початкових профілів величин u і v, тобто функцій $u(0,x) = u_0(x)$, $v(0,x) = v_0(x)$. Оскільки $v = u''_{xx}$, то повинно бути $v_0(x) = u''_{0xx}(x)$. Створіть ці функції $u_0(x), v_0(x)$.

Global Definitions: RMB > Functions - Analytic
Function name = u0
Definition – **Expression** = 6*sech(x)^2
Для перевірки побудуйте графік функції на відрізку [-8,8].
Plot Parameters – **Low limit, Upper limit** = -8, 8
LMB> Plot



Global Definitions: RMB > Functions - Analytic

Function name = v0

Definition – **Expression** = 24*sech(x)^2*tanh(x)^2-12*sech(x)^4

Геометрія моделі складається з відрізка [−8,8]. Створіть його.

Geometry->**Primitives->** Interval

Geometry 1 - Interval 1 (i1):

Interval – **Left endpoint** = −8

Interval – **Right endpoint** = 8

Interval 1: LMB > Build All Objects

На створеному відрізку генеруйте дрібну СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

Mesh Settings – **Element Size** = Extremely fine

LMB> Build All

Задайте рівняння. Відкрийте ПВ вузла **General Form PDE (g) - General Form PDE 1**. В розділі **Equation** цієї панелі наведено загальний вид генеральної форми

$$e_a \frac{\partial^2 \boldsymbol{u}}{\partial t^2} + d_a \frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \Gamma = F \text{, де } \boldsymbol{u} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \text{ і } \nabla = \frac{\partial}{\partial x}.$$

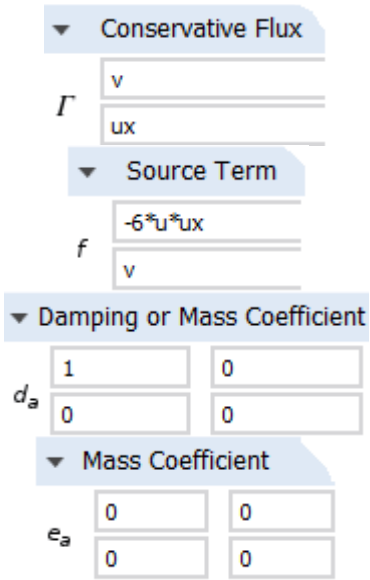
Щоб отримати систему (2) необхідно ввести наступні значення.

Conservative Flux – $\boldsymbol{\Gamma} = \begin{matrix} v \\ ux \end{matrix}$

Source Term – $\mathbf{f} = \begin{matrix} -6 * u * ux \\ v \end{matrix}$

Damping or Mass Coefficient – $\mathbf{d_a} = \begin{matrix} 1 & 0 \\ 0 & 0 \end{matrix}$

Mass Coefficient – $\mathbf{e_a} = \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix}$



Дійсно, ці позначення призводять до системи рівнянь

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \frac{\partial}{\partial t} \begin{pmatrix} u \\ v \end{pmatrix} + \frac{\partial}{\partial x} \begin{pmatrix} v \\ u'_x \end{pmatrix} = \begin{pmatrix} -6 u u'_x \\ v \end{pmatrix},$$

яка співпадає з (2).

Перевірте також, що в поточній панелі (`General Form PDE`) в розділі **`Domain Selection`** стоїть 1.

Задайте початкові умови.

Initial Values 1:

`Initial Values` – **`Initial Value for u1:`** – **u1** = `u0(x)`

`Initial Values` – **`Initial Value for u2:`** – **u2** = `v0(x)`

Оскільки в рівняння (1) входить лише перша похідна по часу, то значення початкової швидкості для визначення розв'язку непотрібні, і ми залишимо відповідні поля $\frac{\partial u}{\partial t}$ і $\frac{\partial v}{\partial t}$ незмінними, тобто нульовими.

Задайте періодичні граничні умови. Вони фактично ототожнюють ліву і праву межу розрахункової області, роблячи її відрізком періодичності шуканих величини u і v.

`Physics` -> **`Boundaries`** – Periodic Condition

Periodic Condition 1:

`Boundary Selection` – **Selection,** <> = `Manual,1,2` (*кінцеві точки*)

Тут в полі **Selection** треба вказати обидві кінцеві точки, які будуть ототожнюватися, інакше система не буде сприймати умову. Зверніть увагу на те, що після створення вузла **Periodic Condition 1** автоматично створюється вузол **Definitions – Explicit 1**.

Виконайте налаштування розв'язувача, задавши часовий інтервал і похибки обчислень.

Study 1 - Step 1: Time Dependent:

`Study Settings` –**Times** = `range(0,0.025,2)`

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

Study 1: RMB > Show Default Solver

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

`Absolute Tolerance` – **Tolerance** = `1e-6`

Тепер можна було б виконати обчислення, але вони триватимуть достатньо довго.

При чисельному інтегруванні динамічних задач за замовчуванням обирається BDF (backward differentiation formula) метод Адамса зворотного диференціювання. Але для розв'язання хвильових рівнянь більш придатний generalized-α метод. До того ж він прискорює обчислення. Виберіть і налаштуйте його.

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

`Time Stepping` – **Method** = `Generalized alpha`

`Time Stepping` – **Amplification for high frequency** = `0.98`

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1**, що містить графіки розв'язку $u(t,x)$ в усі задані моменти часу. Кривих багато, тому цей графік малоінформативний. Щоб краще проаналізувати поведінку хвилі побудуйте анімацію. Але спочатку внесіть деякі зміни в налаштування графіків.

1D Plot Group 1:

Axis –**Manual axis limits** = `True`

Axis –**y minimum, y maximum** = `0,8.5` (два поля)

Legend – **Show legends, Position** = `True, Upper left` (два поля)

1D Plot Group 1 - Line Graph 1:

Coloring and style – **Line style - Width** = `3`

Legends–**Show legends** = `True`

Тепер побудуйте анімацію.

`Results` –> Export –> Animation – Player

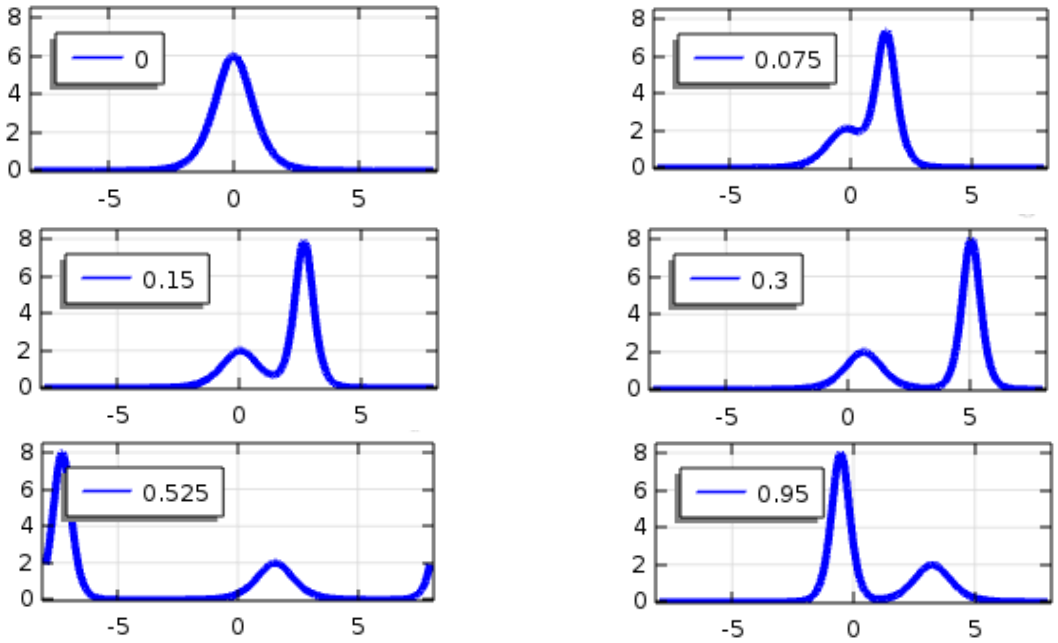
Export - Animation 1:

Scene– **Subject** = `1D Plot Group 1`

Frames– **Frame selection** = `All`

LMB> Generate Frame

Почнеться процес генерування кадрів. По його завершенні натисніть кнопку Play на панелі керування, яка розташована вгорі лівого кута графічного вікна. Цим ви запустите анімацію. На наступному рисунку наведено кілька кадрів, на яких зображено графік розв'язку у деякі моменти часу.



Анімація показує, як солітони (усамітнена хвиля незмінної форми) спочатку розділюються. Більш високий рухається праворуч з більшою швидкістю і виходить за межі рисунка. Зліва з'являється інший, і наздоганяє низький. Солітони зіштовхуються і знову розділюються.

По-іншому продемонструвати рух солітонів можна за допомогою «Parametric Extrusion 1D» графіка, який може зображувати профіль хвилі над прямокутником $[-8,8]\times[0,2]$ площини $X\times t$ (хрестик $\times$ позначає декартовий добуток). Для цього спочатку створіть відповідне джерело даних.

Results - Data Sets : RMB > More Data Sets –> Parametric Extrusion 1D

Data Sets - Parametric Extrusion 1D 1 :

Data – **Data set** = `Study 1/Solution 1`

Data –**Time selection** = Manual

Data –**Times (1-81)** = range(1,4,81) (кожний четвертий момент часу)

Settings – **Level scale factor** = True,16 (пропорції 1×2)

В прямокутнику $[-8,8]\times[0,2]$ площини $X\times t$ при масштабному множнику 16 «часова» довжина, яка дорівнює 2, множиться на 16, і розмір прямокутника джерела даних стає 16×32.

Використайте створене джерело даних для побудови профілю хвилі.

Results -> Plot Group - 2D Plot Group

2D Plot Group 2 :

Data – **Data set** = Parametric Extrusion 1D 1

2D Plot Group 1 -> Add Plot - Line

2D Plot Group 2 - Line 1 :

Data – **Data set** = From parent

Expression – **Expression** = u

В розділі **Coloring and Style** зробіть бажані налаштування.

Coloring and style – **Line type** = Tube

Coloring and style – **Tube radius expression** = 0.2

Coloring and style – **Radius scale factor** = True, 1

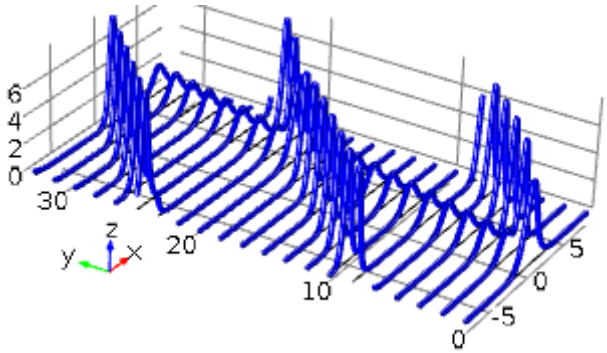
Coloring and style – **Coloring, Color** = Uniform,Blue

Додайте підвузол Height Expression.

2D Plot Group 2 – Line 1 : RMB > Height Expression

Line 1 – Height Expression 1 : Axis – **Scale factor** = True, 1 (*два поля*)

LMB > Plot



Тут вздовж осі y відкладено час t. Профіль хвилі зображено над відрізками [-8,8] по часових шарах, які відповідають кожному четвертому (див. налаштування вузла **Parametric Extrusion 1D 1**) моменту часу t зі списку range(0,0.025,2).

Вправа. Наявність в одновимірній задачі періодичних граничних умов можна інтерпретувати як задачу над кільцем. Зобразіть профіль хвилі (і анімацію її розповсюдження) над кільцем.

Зауваження. Спроба використання довільних початкових умов призведе до майже випадкових «коливальних» профілів розв'язку. Відокремлені хвилі не утворюються. Точніше, вони утворюються, але відслідкувати їх важко із-за великої кількості.

3.28. Сферично-симетрична задача теплопровідності

Багато природничих моделей відзначаються сферичною симетрією. Це дозволяє значно спростити процес обчислень, звівши його до розв'язання одновимірних задач. Для використання переваг сферичної симетрії потрібно щоб:

- розрахункова область мала сферичну форму;
- граничні умови не залежали від зенітного θ і азимутального φ кутів;
- матеріальні властивості і потужність внутрішніх джерел залежали лише від відстані *r* до центра кулі (не залежали від кутів θ і φ);
- в динамічних задачах початкові умови залежали від відстані *r* до центра і не залежали від кутів θ і φ.

Тут ми покажемо, як можна розв'язати сферично симетричну задачу теплопровідності, використовуючи `General Form PDE` інтерфейс.

Рівняння теплопровідності має вигляд

$$\rho c \frac{\partial T}{\partial t} + \nabla \cdot \left(-k \nabla T\right) = Q,$$

де ρ – густина $\left(kg/m^3\right)$, c – теплоємність $\left(J/\left(kg \cdot K\right)\right)$, k – коефіцієнт теплопровідності $\left(W/\left(m \cdot K\right)\right)$, Q – потужність внутрішніх джерел $\left(W/m^3\right)$.

В сферичних координатах r, θ, φ рівняння має вигляд

$$\rho c \frac{\partial T}{\partial t} - k\left[\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2 \frac{\partial T}{\partial r}\right) + \frac{1}{r^2 \sin\theta}\frac{\partial}{\partial \theta}\left(\sin\theta \frac{\partial T}{\partial \theta}\right) + \frac{1}{r^2 \sin^2\theta}\frac{\partial^2 T}{\partial \varphi^2}\right] = Q.$$

Припускаючи, що тіло має сферичну форму, початкова і гранична температура та зовнішні теплові потоки не залежать від кутів θ і φ, отримуємо

$$\rho c \frac{\partial T}{\partial t} + \frac{1}{r^2}\frac{\partial}{\partial r}\left(-k r^2 \frac{\partial T}{\partial r}\right) = Q.$$

Щоб уникнути ділення на нуль при $r = 0$ помножимо це рівняння на r^2.

$$r^2 \rho c \frac{\partial T}{\partial t} + \frac{\partial}{\partial r}\left(-k r^2 \frac{\partial T}{\partial r}\right) = r^2 Q.$$

Позначимо радіус сфери *R*. Ввівши безрозмірну координату $x = \frac{r}{R}, \frac{\partial}{\partial r} = \frac{1}{R}\frac{\partial}{\partial x}$, на відрізку $x \in \left[0,1\right]$ отримаємо наступне ДРЧП

$$x^2 \rho c \frac{\partial T}{\partial t} + \frac{\partial}{\partial x}\left(\frac{-k x^2}{R^2}\frac{\partial T}{\partial x}\right) = x^2 Q. \qquad (1)$$

Граничні умови. Будемо вважати, що на поверхні кулі $r = R$ має місце конвективний теплообмін з навколишнім середовищем

$$q_{in} = h_s \left(T_{ext} - T\right), \qquad (2)$$

де q_{in} – потужність теплового потоку через одиницю площі ззовні поверхні кулі $\left(W/m^2\right)$, h_s – коефіцієнт теплопередачі $\left(W/\left(m^2 K\right)\right)$, T_{ext} – зовнішня температура.

Внаслідок сферичної симетрії в центрі кулі $r = 0$ тепловий потік дорівнює нулю. Тому в центрі $\partial T / \partial r = 0$.

В поточному прикладі вважатимемо, що початкова температура кулі T_0 стала і внутрішні джерела тепла відсутні.

В моделі використаємо наступні значення геометричних і фізичних параметрів.

Позначка	Значення	Пояснення
R	$0.005\ m$	радіус кулі
ρ	$2000\ kg/m^3$	густина
c	$300\ J/(kg \cdot K)$	теплоємність
k	$0.5\ W/(m \cdot K)$	коефіцієнт теплопровідності
h_s	$1000\ W/(m^2 \cdot K)$	коефіцієнт теплопередачі
T_0	$25^0 C$	початкова температура
T_{ext}	$95^0 C$	зовнішня температура

Приклад. Створіть шаблон моделі послідовністю інструкцій.

File->New->Model Wizard->1D->

Mathematics – PDE Interfaces – GeneralForm PDE(g)->

Dependent variables = `T` **->** Time Dependent

Зверніть увагу, що шукану величине ми позначили ідентифікатором `T`.

Вимкніть в моделі використання розмірностей фізичних величин.

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	Description
R	0.005	0.005	радіус кулі
rho	2000	2000	густина
c	300	300	теплоємність
k	0.5	0.5	коефіцієнт теплопровідності
hs	1000	1000	коефіцієнт теплопередачі
T0	273+25	298	початкова температура
Text	273+95	368	зовнішня температура

Геометрія моделі складається з відрізка `[0,1]`. Створіть його.

Geometry->**`Primitives`->** Interval

Interval 1: LMB > Build All Objects

На створеному відрізку генеруйте дрібну СкЕ сітку.

Component 1 (comp1) - Mesh 1 :

`Mesh Settings` – **Element Size** = `Extra fine`

LMB> Build All

Задайте рівняння. Відкрийте ПВ вузла **General Form PDE (g) - General Form PDE 1**. В розділі **Equation** цієї панелі наведено загальний вид генеральної форми

$$e_a \frac{\partial^2 T}{\partial t^2} + d_a \frac{\partial T}{\partial t} + \nabla \cdot \Gamma = f \text{, де } \nabla = \frac{\partial}{\partial x}.$$

Щоб отримати рівняння (1) необхідно ввести наступні значення.

Conservative Flux – $\boldsymbol{\Gamma}$ = `-k*x^2/R^2*Tx`

В якості Γ ми записали вираз, який в рівнянні (1) знаходиться в круглих дужках другого члена.

Source Term – **f** = 0

Damping or Mass Coefficient – $\mathbf{d_a}$ = `x^2*rho*c`

Mass Coefficient – $\mathbf{e_a}$ = 0

Задайте початкові умови.

Initial Values 1:

Initial Values – **Initial Value for T:**– **T** = `T0`

Задайте граничні умови. Закон теплопровідності Фур'є стверджує, що потужність теплового потоку через одиницю площі ззовні поверхні кулі дорівнює $q_{in} = k \nabla T = k \frac{\partial T}{\partial r} = \frac{k}{R} \frac{\partial T}{\partial x}$. Підставляючи це в (2), при $x = 1$ отримуємо $\frac{k}{R} \frac{\partial T}{\partial x} = h_s (T_{ext} - T)$. Враховуючи, що $\Gamma = -k \frac{x^2}{R^2} \frac{\partial T}{\partial x}$, маємо $\frac{\partial T}{\partial x} = -\Gamma \cdot \frac{R^2}{k x^2}$. Тоді $\frac{k}{R}\left(-\Gamma \cdot \frac{R^2}{k x^2}\right)_{x=1} = h_s (T_{ext} - T)_{x=1}$, або $-\Gamma|_{x=1} = \frac{h_s}{R}\left(T_{ext} - T|_{x=1}\right)$. Таку умову на правому кінці відрізка [0,1] можна задати, використовуючи вузол типу `Flux/Source`. В цих вузлах задається умова $-\boldsymbol{n} \cdot \Gamma = g - qu$. Щоб вона (в точці $x = 1$) набула потрібного вигляду, можна покласти $q = 0$ і $g = \frac{h_s}{R}(T_{ext} - T)$ (або $g = \frac{h_s}{R} T_{ext}$, $q = \frac{h_s}{R}$).

`Physics` -> **Boundaries** – Flux/Source

General Form PDE (g) - Flux/Source 1:

Boundary Selection – **Selection** = `Manual,2` (*точка* x=1)

Boundary Flux/Source – **g** = `hs*(Text-T)/R`

Boundary Absorption/Impedance Term – **q** = 0

Крізь центральну точку $r = 0$ потік тепла повинен дорівнювати нулю. Тому нулю дорівнюватиме вираз $k T_r'$, що еквівалентно умові $T_x' = 0$ при $x = 0$. Задамо її у вузлі типу `Constraint`. Гранична умова `Constraint` використовує вираз `R`, який повинен обертатися на нуль (`R=0`), обмежуючи розв'язок на обраній частині геометрії. Вираз `R` можна будувати досить довільним, і ніяк не пов'язаним зі значенням терму Γ.

`Physics` -> **Boundaries** – Constraint

Constraint 1:

Boundary Selection – **Selection,** <> = Manual, 1 (точка x=0)

Constraint – **R** = Tx

Цим задається гранична умова $T'_x\big|_{x=0} = 0$.

Налаштуйте розв'язувач.

Study 1 - Step 1: Time Dependent:

Study Settings –**Times (s)** = range(0,0.25,10)

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Results** з'явиться графічна група **1D Plot Group 1** з вузлом **Line Graph 1**, що містить графік температури T в усі моменти часу зі списку **Times (s)**. Змініть налаштування цієї групи, показавши кожну четверту криву.

1D Plot Group 1:

Data –**Time selection** = Interpolated

Data –**Times** = range(0,1,10) (*список моментів часу*)

Plot Settings –**x-axis label,** <> = True, radius

Plot Settings –**y-axis label,** <> = True, T(K)

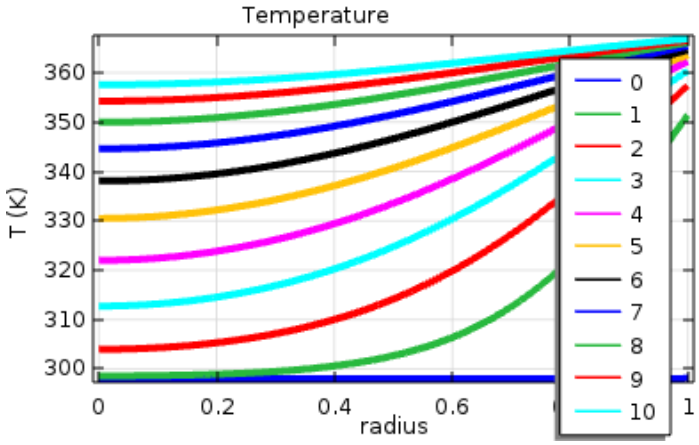
Останні дві інструкції задають на графіку підписи/позначки біля осей координат.

1D Plot Group1 - Line Graph 1:

Coloring and style – **Width** = 3

Legends–**Show legends** = True

LMB > Plot



Як бачимо, температура кулі поступово вирівнюється, наближуючись до температури зовнішнього середовища.

Побудуємо графік залежності температури центру кулі від часу.

Results -> Plot Group - 1D Plot Group

1D Plot Group 2:

Plot Settings –**x-axis label,** <> = True, t(s)

Plot Settings –**y-axis label,** <> = True, T(K)

1D Plot Group 2-> Point Graph

1D Plot Group 2 - Point Graph 1:

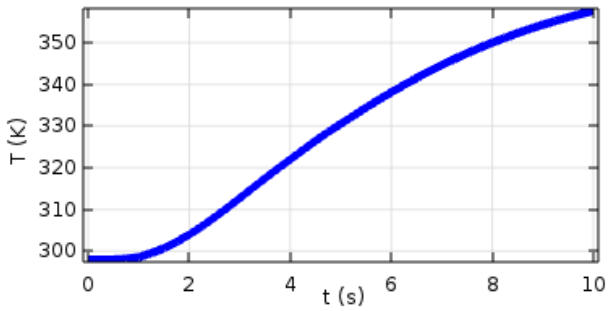
Selection – **Selection,** <> = Manual, 1 (точка x=0)

y-Axis Data – **Expression** = T

x-Axis Data – **Parameter** = Time

Coloring and style – **Width** = 3

LMB > Plot



Останні графіки відображають суть теплових процесів, що відбуваються, але не показують розподіл температури в тривимірній кулі. Для цього потрібна тривимірна модель. Якби ми відразу розв'язували 3D задачу теплопровідності, то це можна було б зробити. Та ми спеціально використали сферично-симетричну модель, щоб не виконувати тривимірні обчислення. Бажано розпорядитися вже знайденим сферично-симетричним розв'язком $T(r)$ так, щоб побудувати об'ємний графік розподілу температури в кулі. Для цього в існуючому файлі можна створити додатковий тривимірний компонент і, не виконуючи в ньому ніяких обчислень, використати його геометрію для побудови графіка.

Створіть другий компонент і в ньому власну геометрію.

Home **-> Model ->** Add Component - 3D

Активуйте гілку нового компонента і побудуйте в ньому кулю і блок з параметрами за замовчуванням.

Component 2 - Geometry 2 :> Geometry->**Primitives** – Sphere

LMB > Build Selected

Component 2 - Geometry 2 :> Geometry->**Primitives** – Block

LMB > Build Selected (*наступний рисунок ліворуч*)

Видаліть блок з кулі.

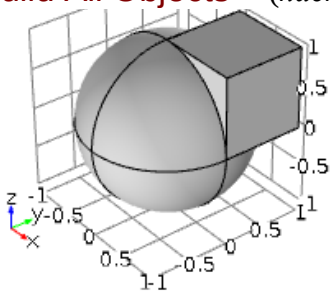
Geometry **->Booleans and Partitions** – Difference

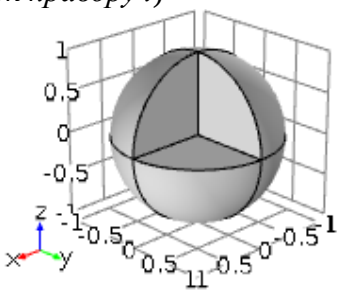
Component 2 - Geometry 2 - Difference 1:

Difference – **Object to add** = sph1 (*куля*)

Union – **Object to subtract** = blk1 (*блок*)

LMB > Build All Objects (*наступний рисунок праворуч*)





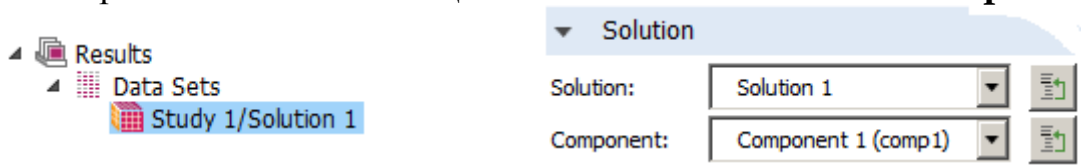
На створеній геометрії генеруйте СкЕ сітку з параметрами за замовчуванням.

Component 2 (comp1) - Mesh 2 : LMB> Build All

Виконайте обчислення.

Study 1 : LMB > Compute

В гілці **Result** оновиться джерело даних **Study 1/Solution 1**. Але воно буде пов'язано з першим компонентом. Це вказано в його ПВ в полі **Component.**

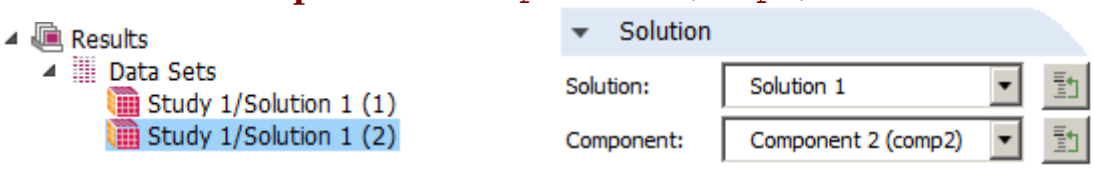


Ми можемо дублювати це джерело даних і в створеній копії змінити базовий компонент.

Result – Data Sets - Study 1/Solution 1: RMB> Duplicate

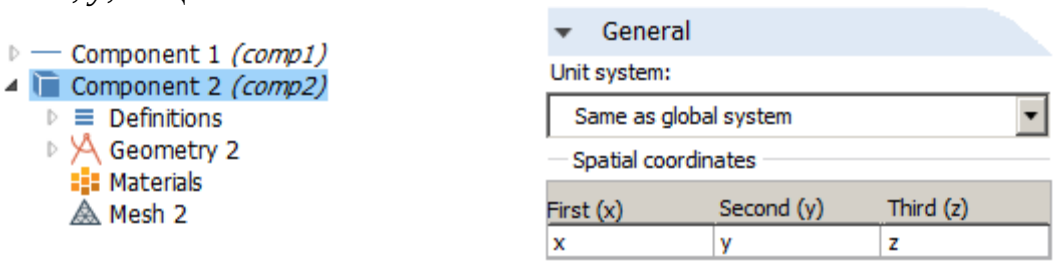
Result – Data Sets - Study 1/Solution 1(2):

Solution – **Component** = Component 2(comp2)



Поверхневий графік, який спиратиметься на ДД **Study 1/Solution 1(2)**, буде зафарбовувати поверхню тіла, створеного в компоненті **Component 2**.

В тривимірній геометрії другого компонента використовуються просторові координати *x*, *y*, *z*. Це можна побачити в його ПВ.



Для фарбування поверхню кулі (з вирізаною частиною одного октанту) відповідно сферично-симетричного розв'язку $T(r)$, потрібно побудувати графік функції $T\left(\sqrt{x^2+y^2+z^2}\right)$. Але величина T обчислена в першому компоненті, а координати точок *x*, *y*, *z* беруться з другого. Крім того, треба зіставити вираз $\sqrt{x^2+y^2+z^2}$ з координатою x (нормований радіус r) першого компонента. Для подолання подібних складностей використовуються Component Coupling оператори, які створюються в тому компоненті, з якого будуть вибиратися дані. Нам потрібен General Extrusion оператор, за допомогою якого можна створити довільне відображення $f(x,y,z) \Leftrightarrow x$, зокрема тут ми створимо відображення $\sqrt{x^2+y^2+z^2} \Leftrightarrow x$. Оскільки буде використовуватися розв'язок T з першого компонента, то оператор потрібно створювати в ньому.

Component 1 (comp1) – Definitions:

RMB > Component Couplings - General Extrusion

Component 1 (comp1) – Definitions -General Extrusion 1 (genext1):

Source Selection – **Selection, <>** = Manual,1 (*відрізок* [0,1])

Destination Map – **x-expression** = sqrt(x^2+y^2+z^2)

Вираз sqrt(x^2+y^2+z^2) буде обчислюватися в розрахунковій області адресата (будь-якій геометрії, в якій існують координати x, y, z), і буде

ототожнюватися з координатою x вздовж відрізка [0,1] першого компонента. В полі **Opertor name** залиште ім'я `genext1` без змін. Як наслідок, оператор `genext1(...)` буде задавати відповідність абсцис x точок відрізка [0,1] і виразу $\sqrt{x^2+y^2+z^2}$ в геометрії кулі. Використання «в тривимірній геометрії» оператора `genext1(T)` буде означати обчислення виразу $T\left(\sqrt{x^2+y^2+z^2}\right)$.

Оператор відповідності `General Extrusion` можна застосовувати в будь-яких місцях моделі, і для уточнення перед його іменем слід використовувати префікс `comp1` (тобто писати `comp1.genext1(T)`), що являє ім'я компонента, в якому оператор створений, і з якого вибирається величина `T`.

Щоб оператор став доступним, переобчисліть модель командою Update Solution, або заново виконайте команду Compute.

Тепер можна будувати бажаний графік.

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

3D Plot Group 3 :

`Data` – **Data set** = `Study 1/Solution 1(2)`

`Data` –**Time** = `10`

3D Plot Group 3 : RMB > Surface

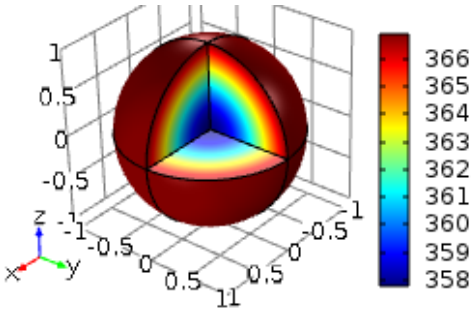
3D Plot Group 3 - Surface 1 :

`Data` – **Data set** = `From parent`

`Expression` – **Expression** = `comp1.genext1(T)`

Це те місце, де використовується створений оператор `General Extrusion`.

LMB > Plot



На базі графічної групи **3D Plot Group 3** можна побудувати анімацію (зробіть це самостійно).

Література.

1. Тихонов А.Н., Самарский А.А. Уравнения математической физики. – М.: Наука, 1977. – 736с.
2. Меркин Д.Р. Введение в механику гибкой нити. – М.: Наука. Главная редакция физико-математической литературы, 1980. – 240 с.
3. Карслоу Х.С. Теория теплопроводности. – М.–Л.: Гос. изд. техн.-теор. литературы, 1947.
4. Кошляков Н.С., Глинер Э.Б., Смирнов М.М. Основные дифференциальные уравнения математической физики. – М.: Гос. изд. физ.-мат. лит., 1962.
5. Сегерлинд Л. Применение метода конечных элементов. – М.: Мир, 1979. – 392 с.
6. Купрадзе В.Д. Основные задачи математической теории дифракции, ОНТИ Л.-М., 1935.
7. Абловиц М., Сигур Х. Солитоны и метод обратной задачи. – М.: Мир, 1987. – 479 с.
8. Франк Ф., Мизес Р. Дифференциальные и интегральные уравнения математической физики. – М.–Л.: Главная редакция общетехнической литературы, 1937.
9. Тимошенко С.П., Войновский-Кригер С. Пластинки и оболочки. – М.: Гос. изд. физ.-мат. лит., 1963.
10. Лурье А.И. Операционное исчисление и его приложение к задачам механики. – М.–Л.: Гос. изд. техн.-теор. литературы, 1950.