Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

магістра
(назва освітньо-кваліфікаційного рівня)

студента	*Титаренко Марії Олександрівні*
	(ПІБ)
академічної групи	*121М-19-1*
	(шифр)
спеціальності	*121 Інженерія програмного забезпечення*
	(код і назва спеціальності)
освітньої програми	*«Інженерія програмного забезпечення»*
	(назва освітньої програми)
на тему:	*Розробка та дослідження ефективності впровадження програмного забезпечення двовимірної задачі теплопровідності*

_____*М.О. Титаренко*

Керівники	Прізвище, ініціали	Оцінка за шкалою		Підпис
		рейтинговою	інституційною	
розділів кваліфікаційної роботи				
спеціальний	*Проф. Бердник М.Г.*			
економічний	*Проф. Вагонова О.Г.*			
Рецензент	*Доц. Шедловський І.А.*			
Нормоконтролер	*Проф. Лактіонов І.С.*			

Дніпро
2022

Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
Завідувач кафедри
Програмного забезпечення комп'ютерних систем
(повна назва)

М.О. Алексєєв
(підпис) (прізвище, ініціали)

« » 20 22 Року

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності *121 Інженерія програмного забезпечення*
(код і назва спеціальності)

студенту *121м-20-1* *Титаренко Марії Олександрівні*
(група) (прізвище та ініціали)

Тема кваліфікаційної роботи *Розробка та дослідження ефективності впровадження програмного забезпечення двовимірної задачі теплопровідності*

1 ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Наказ ректора НТУ «Дніпровська політехніка» від 31.10.2022 р. № 1200-с

2 МЕТА ТА ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Об'єкт досліджень – . теплові процеси, які відбуваються в двовимірних тілах.

Предмет досліджень – математичні моделі та методи розрахунку теплових процесів, які відбуваються в двовимірних тілах.

Мета НДР – підвищення ефективності обчислення та аналізу температурних полів в двовимірних тілах.

Вихідні дані для проведення роботи – теоретичні та експериментальні дослідження, методи скінченних елементів і Гальоркіна, методи обробки експериментальних даних.

3 ОЧІКУВАНІ НАУКОВІ РЕЗУЛЬТАТИ

Новизна запропонованих рішень полягає в обґрунтуванні та розв'язанні проблеми математичного моделювання температурних полів в двовимірних тілах, що дає можливість більш точно, ефективно і економно проектувати тіла для використання у різноманітних механізмах та пристроях, які знаходяться під інтенсивним впливом температур.

Практична цінність полягає в розробці чисельних методів і програмного забезпечення розрахунку полів температури в двовимірних тілах, що дозволяє знаходити температурні поля із більшою точністю.

4 ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Результати досліджень мають бути подані у вигляді, що дозволяє побачити та оцінити безпосереднє використання запропонованих методів. В результаті роботи повинне бути розроблене програмне забезпечення для розрахунку розподілу температурних полів в двовимірних тілах.

5 ЕТАПИ ВИКОНАННЯ РОБІТ

Найменування етапів робіт	**Строки виконання робіт (початок – кінець**
Аналіз теми та постановка задачі	12.09.2022-30.09.2022
Розробка моделі, метода та програмного забезпечення розрахунку температурних полів в двовимірних тілах	01.10.2022-31.10.2022
Використання програми та аналіз отриманих результатів	01.11.2022-12.12.2022

6 РЕАЛІЗАЦІЯ РЕЗУЛЬТАТІВ ТА ЕФЕКТИВНІСТЬ

Економічний ефект від реалізації результатів роботи очікується позитивним завдяки розробці алгоритму та програмного забезпечення для розрахунку розподілу температурних полів у двовимірних тілах, що дозволяє зменшити витрати при проектуванні і розробці таких тіл для їх використання у механізмах та пристроях.

Соціальний ефект від реалізації результатів роботи очікується позитивним, завдяки полегшенню роботи науковців і інженерів, що дозволить прискорити та здешевити розробку механізмів та пристроїв.

7 ДОДАТКОВІ ВИМОГИ

Завдання видав _____ *Бердник М.Г.*.
(підпис) (прізвище, ініціали)

Завдання прийняв до виконання _____ *Титаренко М.О.*.
(підпис) (прізвище, ініціали)

Дата видачі завдання: *12.09.2022 р.*

Термін подання кваліфікаційної роботи до ЕК *12.12.2022*

РЕФЕРАТ

Пояснювальна записка: 88 стор., 37 рис., 4 додатка, 27 джерел.

Об’єкт дослідження: теплові процеси, які відбуваються в двовимірних тілах.

Предмет дослідження: математичні моделі та методи розрахунку теплових процесів, які відбуваються в двовимірних тілах.

Мета магістерської роботи: підвищення ефективності обчислення та аналізу температурних полів в двовимірних тілах.

Методи дослідження. Для вирішення поставлених задач використані методи: чисельні методи в інформатиці, метод скінченних елементів.

Новизна отриманих результатів полягає в обґрунтуванні та розв’язанні проблеми математичного моделювання температурних полів в двовимірних тілах, що дає можливість більш точно, ефективно і економно проектувати тіла для використання у різноманітних механізмах та пристроях, які знаходяться під інтенсивним впливом температур.

Практична цінність результатів полягає в розробці чисельних методів і програмного забезпечення розрахунку полів температури в двовимірних тілах, що дозволяє знаходити температурні поля із більшою точністю.

Область застосування. Розроблене програмне забезпечення може застосовуватися для проектування двовимірних тіл, які знаходяться під інтенсивним впливом температури.

Значення роботи та висновки. Розроблене програмне забезпечення дозволяє проектувати двовимірні тіла обертання зі значним скороченням матеріальних та часових витрат, підвищити їх ефективність та безпечність.

Прогнози щодо розвитку досліджень. Розробити WEB-додаток для розрахунку температурних полів в двовимірних тілах.

Список ключових слів: математичне моделювання, температурне поле, теплопровідність, метод скінченних елементів, Maple.

ABSTRACT

Explanatory note: 88 pages, 37 figures, 4 appendices, 27 sources.

The object of research: thermal processes that occur in two-dimensional bodies.

The subject of research: mathematical models and methods of calculating thermal processes that occur in two-dimensional bodies.

The purpose of the master's work: increasing the efficiency of calculation and analysis of temperature fields in two-dimensional bodies.

Research methods. The following methods are used to solve the problems: numerical methods in informatics, finite element method.

The novelty of the obtained results lies in the substantiation and solution of the problem of mathematical modeling of temperature fields in two-dimensional bodies, which makes it possible to more accurately, efficiently and economically design bodies for use in various mechanisms and devices that are under the intense influence of temperatures.

The practical value of the results lies in the development of numerical methodsand software for calculating temperature fields in two-dimensional bodies, which allows finding temperature fields with greater accuracy.

Field of application. The developed software can be used for the design of two-dimensional bodies that are under the intense influence of temperature.

Value of work and conclusions. The developed software allows you to design two-dimensional bodies of rotation with a significant reduction in material and time costs, increase their efficiency and safety.

Forecasts regarding the development of research. Develop a WEB application for calculating temperature fields in two-dimensional bodies.

Keyword list: mathematical modeling, temperature field, thermal conductivity, finite element method, Maple.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

МСЕ – метод скінченних елементів;

ЕОМ – Електронна обчислювальна машина;

ЧМІ – чисельні методи у інформатиці;

ДРЧП –диференціальні рівняння у частинних похідних;

ПЗ – програмне забезпечення.

ЗМІСТ

ВСТУП

Процеси теплопередачі грають винятково велику роль як у природі, так і в сучасній техніці [1-3]. Як показують дослідження – теплопередача є складним процесом. При вивченні цей процес поділяють на прості явища. Теплопровідність – перенесення тепла (або внутрішньої енергії) при безпосередньому зіткненні тіл (або частин тіла) з різною температурою [4].

В даний час практика висуває нові і різноманітні задачі, що вимагають від людини вміння самостійно і творчо використовувати основні закони і методи теплопередачі [5-6]. Значно розширилася можливість прикладного використання теорії теплопровідності у зв'язку з усе більш широким впровадженням в інженерну практику швидкодіючих ЕОМ [7-8]. Багато завдань, які ще нещодавно вирішувалися тільки вузькими фахівцями в галузі теорії теплообміну, можуть бути вирішені в умовах виробництва. З одного боку ця область науки досить добре опрацьована, отримані надійні дані, які можна використовувати при вирішенні тих чи інших конкретних завдань, що виникають при проектуванні та експлуатації теплотехнічного обладнання, з іншого - проблемна, оскільки використання нових матеріалів, розширення діапазону дії теплотехнічних пристроїв вимагає створення нових, більш надійних методів розрахунку.

За останні десятиліття інтерес до математичного моделювання складних фізичних процесів і необхідність у ньому помітно зросли. Цьому значною мірою сприяє прогрес у розвитку комп'ютерної техніки, чисельних методів вирішення всіх типів задач математичної фізики та реалізованих на цій основі математичних моделей. Будь-яка сучасна наукомістка технологія так чи інакше використовує результати обчислювальних машин. У провідних наукових центрах розвинених країн інтенсивно розробляються нові чисельні методи, алгоритми та пакети прикладних програм для розв'язання відповідних класів задач.

Основною перевагою чисельних методів є можливість заміни дорогого або важкозатратного фізичного експерименту, а також можливість моделювання процесів, що не піддаються аналітичному рішенню. Необхідність в чисельному

моделюванні процесу теплопровідності виникає у багатьох галузях сучасної техніки. Одним з найбільш простих чисельних методів розв'язання рівняння теплопровідності є метод скінчених елементів(МСЕ).

Метою даної дипломної роботи є розробка програмного забезпечення (ПЗ) для розв'язання двовимірної задачі теплопровідності.

Актуальність розробленого продукту полягає в створені такого (ПЗ), що дозволяє підвищити ефективність розрахунків задач двовимірної задачі теплопровідності , а саме для таких задач, як: побудова лінійного інтерполяційного полінома, побудова інтерполяційного полінома для дискретизованної області, побудова елементів високого порядку, побудова двовимірної задачі теплопровідності, автоматична генерація скінченно-елементної сітки.

Кваліфікаційна робота складається зі вступу, трьох розділів, висновків, переліку використаних джерел і трьох додатків.

У вступі розглядається аналіз та сучасний стан проблеми, наведено опис структури роботи.

У першому розділі проведено аналіз предметної області, конкретизується мета кваліфікаційної роботи, наведено обґрунтування актуальності теми та розроблена постановка завдання, задані вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі розглядаються аналітичні та чисельні методи для розв'язання двовимірної задачі теплопровідності, опис використаних технологій та мов програмування ……...

У третьому розділі, виконано проектування і розробка ПЗ, наведено опис алгоритму функціонування програми, визначені вхідні і вихідні дані, описаний виклик та завантаження програми, описана робота програми.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАДАЧІ

1.1. Задача дослідження двовимірної задачі теплопровідності

Ціллю даної роботи є розробка програмного забезпечення для розрахунку температурних полів стаціонарних задач теплопровідності. Рівняння теплопровідності має вигляд [4]:

$$\frac{d}{dx}\left(K_{xx}\frac{d\varphi}{dx}\right)+\frac{d}{dy}\left(K_{yy}\frac{d\varphi}{dy}\right)+\frac{d}{dz}\left(K_{zz}\frac{d\varphi}{dz}\right)+Q=0$$

У двовимірному випадку, ми отримаємо таке рівняння, яке будемо розглядатися у ході всієї роботи:

$$\frac{d^2\varphi}{dx^2}+\frac{d^2\varphi}{dy^2}+2G\theta=0$$

Для рішення диференційних рівнянь є багато підходів, одним з них є МСЕ. МСЕ – це чисельний метод розв’язання диференційних рівнянь у частинних похідних. Цей метод часто застосовується у задачах механіці деформованого твердого тіла, теплообміну, гідродинаміці та електродинаміці. МСЕ потребує більших зусиль, ніж метод скінчених різниць. У МСЕ, є ряд переваг, які можна побачити на реальних задачах: довільна форма області, яка задається, сітку можна зробити більш рідкісною, де особлива точність не потрібна. Довгий час широкому використанню МСЕ заважала відсутність алгоритмів автоматичного розбиття області на «майже рівносторонні» трикутники (похибка, у залежності від варіації метода, обернено пропорційна синусу чи самого гострого, чи самого тупого кута при розбитті. Цю задачу вдалося успішно вирішити (алгоритми основані на тріангуляції Делоне), що зробило можливість повністю автоматично рішати проблеми с побудовою сітки для метода МСЕ.

МСЕ можна розподілити на декілька етапів:

1. Вибір кінцевого елемента.

У одновимірній задачі це буде звичайний відрізок. У двовимірній задачі – трикутний (симплекс) елемент, чотирьохкутний елемент, у загальному

випадку довільна фігура, за допомогою якої можна розбити досліджувану область на неперетинні підобласті, але також слід звернути увагу, що чим складніше елемент, тім з більшими складностями доведеться зіткнутися при рахуванні інтегралів, тому найбільш поширеними є трикутник та чотирикутних зі сторонами, які паралельні вісям координат. Для трьохвимірної області – тетраедр та паралелепіпед.

2. Розбиття області на кінцеві елементи.

Відмінність від метода скінчених різниць полягає в тому, що розбиття може бути не рівномірним и апріорно враховувати градієнт фазової змінної, це означає, що де передбачується швидке змінення фазової змінної сітка повинна бути більш густою та навпаки. Є різні способи автоматичного розбиття області на кінцеві елементи, тому що виконання цього процесу вручну довго та часто веде до помилок.

3. Отримання функцій форми.

В залежності від співвідношення вимог до точності задачі та можливостей комп'ютерної техніки вибирають лінійну, квадратичну, або більш високу ступень функції форми, при цьому кількість вузлів апроксимації повинно бути, як мінімум на одиницю більше порядку апроксимованої функції.

4. Побудова матриці жорсткості та вектора навантажень.

Назву цей етап взяв з будівельної механіки, тому що саме там вперше був застосований МСЕ. На цьому етапі використовується метод зважених нев'язок у границях одного кінцевого елемента.

5. Врахування граничних вимог.

6. Рішення системи алгебраїчних рівнянь.

При рішенні може бути врахована властивість матриці коефіцієнтів, тому що вона, як правило, має стрічкову форму. Для рішення системи буде використаний метод Гауса.

1.2. Огляд літератури та існуючих програм

В ході роботи було проведено аналіз існуючої літератури для використання та реалізації метода скінчених елементів. Для рішення проблеми побудови тріангуляції можна використати методичну роботу [9]. В ній можна вибрати, який алгоритми реалізації Делоне простіший, а який оптимальніше використовує пам'ять комп’ютера. Також представлений час виконання програм. Також можна використовувати як пряму побудову тріангуляції, так і алгоритми побудови тріангуляції методом злиття. В роботі [10] детально описаний інкрементний алгоритм побудови тріангуляції, такий алгоритм простіше за все реалізовується на комп’ютері. Для реалізації МСЕ є багато методичних рекомендацій. Одними з найкращих є [11]-[12]. Де детально описується реалізації методу в одновимірному просторі. Розповідається також про переваги використання у двомірному просторі трикутного елементу та яким чином вони відрізняються від чотирикутних. Також є тестові задачі, на одній із таких задач була протестований отриманий продукт для розподілу температурних полів.

ANSYS - універсальна програмна система кінцево-елементного МСЕ аналізу, яка існує і розвивається протягом останніх 30 років, є досить популярною у фахівців в області комп'ютерного інжинірингу (CAE, Computer-Aided Engineering), рішення лінійних і нелінійних, стаціонарних і нестаціонарних просторових задач механіки деформованого твердого тіла та механіки конструкцій (включаючи нестаціонарні геометричні і фізичні нелінійні задачі контактної взаємодії елементів конструкції), завдань механіки рідини і газу, теплопередачі і теплообміну, електродинаміки, акустики, а також механіки зв'язаних полів. Моделювання та аналіз в деяких областях промисловості дозволяє уникнути дорогих і тривалих циклів розробки типу «проектування - виготовлення - випробування».

Програмна система МСЕ аналізу ANSYS розробляється американською компанією ANSYS Inc. Компанія також випустила інші системи МСЕ

моделювання, в тому числі DesignSpace, AI Solutions (NASTRAN, ICEM CFD); призначені для використання в більш специфічних галузях виробництва.

Програмна система ANSYS є досить відомою CAE-системою, яка використовується на таких відомих підприємствах, як ABB, BMW, Boeing, Caterpillar, Daimler-Chrysler, Exxon, FIAT, Ford, БелАЗ, General Electric, LockheedMartin, MeyerWerft, Mitsubishi, Siemens, Shell, Volkswagen-Audi та інші.

Elmer FEM solver - повнофункціональний математичний пакет, орієнтований на математичне моделювання фізичних процесів і розрахунку конструкцій за допомогою методу скінчених елементів МКЕ (Сангл. FEM, finite elements method).

Пакет дозволяє будувати фізичні моделі для вирішення задач гідродинаміки, будівельної механіки, електродинаміки, теплопереносу, акустики і т. д.

Пакет Elmer складається з декількох частин: фізичні моделі, а також граничні та початкові умови задаються в модулі ElmerGUI; чисельне рішення задачі відбувається в ElmerSolver, а результати можна обробити в Elmerpost.

Універсальна система скінченно-елементного аналізу Elmer поширюється на умовах Open Source.

LS-DYNA - багатоцільова програма скінченно-елементного аналізу, що розробляється компанією LSTC (Livermore Software Technology Corporation) (США) з 1987 р. Програма призначена для вирішення трьохвимірних динамічних нелінійних задач механіки деформованого твердого тіла, механіки рідини і газу, теплопровідності. LS-DYNA знайшла широке застосування в таких галузях науки і техніки, як автобудування, військово-промисловий комплекс, авіа-та ракетобудування і т. д.

В LS-DYNA реалізовані явний і неявний МСЕ з можливістю побудови лагранжевой, ейлеровой і гібридної сітки, багатокомпонентна гідродинаміка, без сітковий метод згладжених часток, без сітковий метод, заснований на методі Гальоркіна. Програма має вбудовані процедури автоматичної перебудови і згладжування скінченно-елементної сітки при виродженні елементів,

високоефективні алгоритми розв'язання контактних задач, широкий набір моделей матеріалів, можливості користувальницького програмування.

З 1996 р. вирішувач LS-DYNA вбудований в пакет програм ANSYS, де використовується для вирішення завдань динамічного аналізу. У 2006 р. вирішувач LS-DYNA також увійшов до складу пакету програм MD NASTRAN.

Хоча усі перелічені математичні пакети мають свою сильні сторони, та можуть використовуватися для знаходження температурних полів, для розробки ПЗ у рамках даної кваліфікаційної роботи було убрано MAPLE, через його сумісність з іншими платформами, сучасну документацію та потужні вбудовані бібліотеки.

1.3. Постановка задачі

Розробити математичні моделі та програмне забезпечення для розрахунку температурних полів в двовимірних областях.

Для досягнення поставленої мети необхідно вирішити наступні основні завдання:

– дослідити методи, які дозволяють розрахувати розподіл температури;

розробити програмне забезпечення, яке розраховує розподіл температурних полів у двовимірних областях;

– зробити аналіз залежності відхилення отриманого рішення від аналітичного в залежності від щільності розбиття області.

В ПЗ необхідно реалізувати:

– простий та зрозумілій інтерфейс;
– зручний спосіб введення даних;
– візуалізацію отриманого рішення;
– можливість імпорту рішення у файл;

–.порівняння знайденого рішення з аналітичним розв’язком, якщо таке рішення існує;

– перевірку вхідних даних на їх коректність;
– обробку виняткових ситуацій.

Чисельне рішення характеризується наступними показниками:

- коефіцієнти рівняння;
- граничні умови;
- область, в якій шукається рішення;
- щільністю дискретизації області.

Вихідною інформацією програми є вектор знайдених температурних полів.

Вся вхідна інформація повинна перевірятися на коректність та відповідність очікуваному типу.

1.4 Висновок

Актуальність роботи полягає в створені такого ПЗ, що дозволяє підвищити ефективність розрахунку задач двовимірної задачі теплопровідності, а саме для таких задач, як: побудова лінійного інтерполяційного полінома, побудова інтерполяційного полінома для дискретизованної області, побудова елементів високого порядку, побудова двовимірної задачі теплопровідності, автоматична генерація скінченно-елементної сітки.

РОЗДІЛ 2. ОПИС МЕТОДІВ ВИРІШЕННЯ ЗАДАЧІ

2.1. Опис застосованих математичних методів

МСЕ є чисельним методом розв'язання диференціальних рівнянь, що зустрічаються у фізиці і техніці. Виникнення цього методу пов'язане з вирішенням завдань космічних досліджень [4]. Був опублікований ряд статей з застосуванням МСЕ до задач будівельної механіки та механіки суцільних середовищ. Важливий внесок у теоретичну розробку методу зробив у 1963 р. Мелош, який показав, що МСЕ можна розглядати як один з варіантів добре відомого методу Релея—Рітца [11]. У будівельній МСЕ мінімізацією потенційної енергії дозволяє звести задачу до системи лінійних рівнянь рівноваги[9].

Зв'язок МСЕ з процедурою мінімізації привела до широкого використання його при вирішенні задач в інших галузях техніки. Метод застосовувався до завдань, що описується рівнянням Лапласа або Пуассона. Рішення цих рівнянь також пов'язано з мінімізацією деякого функціоналу. В перших публікаціях за допомогою МСЕ вирішувалися завдання поширення тепла. Потім метод було застосовано до задач гідромеханіки, і зокрема до задачі течії рідини в пористому середовищі[6].

Область застосування методу кінцевих елементів істотно розширилася, коли було показано, що рівняння, які визначають елементи в задачах будівельної механіки, поширення тепла, гідромеханіки, можуть бути легко отримані за допомогою таких варіантів методу зважених нев'язок, як метод Гальоркіна або спосіб найменших квадратів. Встановлення цього факту зіграло важливу роль у теоретичному обґрунтуванні МСЕ, так як дозволило застосовувати його при вирішенні будь-яких диференціальних рівнянь[12].

МСЕ з чисельної процедури розв'язання задач будівельної механіки перетворився в загальний метод чисельного рішення диференціального рівняння або системи рівнянь. Цей прогрес був досягнутий за п'ятнадцятирічний період, за рахунок вдосконалення швидкодіючих ЕОМ, необхідних для

більш точного розрахунку конструкцій літальних апаратів, а також завдяки допомозі Національного комітету з дослідження космічного простору. ЕОМ дозволила прискорити проведення багатьох складних чисельних розрахунків. Вивчення космічного простору зажадало виділення коштів на проведення фундаментальних досліджень і стимулювало вдосконалення універсальних обчислювальних програм. МСЕ являє собою ефективний чисельний метод розв'язання інженерних і фізичних задач.

МСЕ заснований на ідеї наближення неперервної функції (у фізичній інтерпретації - температури, тиску, переміщення і т. д.) дискретною моделлю, яка будується на множині кусково-неперервних функцій, визначених на кінцевому числі підобластей, званих кінцевими елементами. Досліджувана геометрична область розбивається на елементи таким чином, щоб на кожному з них невідома функція апроксимувалась пробною функцією (як правило, поліномом). Причому ці пробні функції повинні задовольняти граничним умовам безперервності, що збігається з граничними умовами, що накладаються самим завданням. Вибір для кожного елемента апроксимуючої функції буде визначати відповідний тип елемента. В даній роботі розглядається обчислювальний алгоритм методу кінцевих елементів у формулюванні, що базується на процедурі мінімізації функціоналу, відповідного розв'язуваної безперервної задачі. В результаті виконання зазначеної процедури відбувається заміщення рівняння або системи рівнянь в частинних похідних системою алгебраїчних рівнянь, які мають в якості коефіцієнтів апроксимуючі функції, які фактично є значеннями шуканої функції в вершинах розбиття. При побудові алгоритму використовується математичний апарат, що надається системою Maple [13-18].

Основна ідея МСЕ полягає в тому, що будь-яку неперервну величину, таку, як температура, тиск і рух, можна апроксимувати дискретною моделлю, яка будується на множині кусково-неперервних функцій, визначених на кінцевому числі підобластей. Кусково-неперервні функції визначаються за допомогою значень неперервної величини в кінцевому числі точок розглянутій області.

У загальному випадку безперервна величина заздалегідь невідома і потрібно визначити значення цієї величини в деяких внутрішніх точках області. Дискретну модель, проте, дуже легко побудувати, якщо спочатку припустити, що числові значення цієї величини в кожній внутрішній точці області відомі. Після цього можна перейти до загального випадку. Отже, при побудові дискретної моделі безперервної величини надходять наступним чином:

1. У розглянутій області фіксується кінцеве число точок. Ці точки називаються вузловими точками або просто вузлами.

2. Значення безперервної величини в кожній вузловій точці вважається змінною, яка повинна бути визначена.

3. Область визначення неперервної величини розбивається на кінцеве число підобластей, званих елементами. Ці елементи мають загальні вузлові точки і в сукупності апроксимують форму області.

4. Безперервна величина апроксимується на кожному елементі поліномом, який визначається за допомогою вузлових значень цієї величини. Для кожного елемента визначається свій поліном, але поліноми підбираються таким чином, щоб зберігалася безперервність величини вздовж кордонів елемента.

Основна концепція МСЕ може бути наочно проілюстрована на одновимірному прикладі заданого розподілу температури в стержні, показаному на рис. 2.1. Розглядається безперервна величина T(x), область визначення відрізок OL вздовж осі x. Фіксовані і пронумеровані п'ять точок на осі x (рис. 2.2,а). Це вузлові точки. зовсім не обов'язково розташовувати їх на однаковій відстані один від одного[1]. Очевидно, можна ввести в розгляд і більше п'яти точок, але цих п'яти цілком достатньо, щоб проілюструвати основну ідею методу. Значення T(x) в даному випадку відомі в кожній вузловій точці. Ці фіксовані значення представлені графічно на рис. 2.2,б і позначені відповідно до номерів вузлових точок через T1, T2,..., T5.

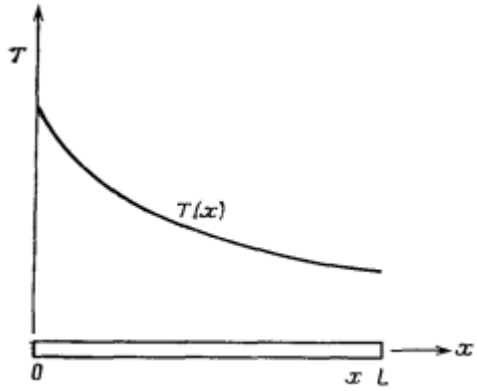


Рис. 2.1. Розподіл температури в стержні одновимірному

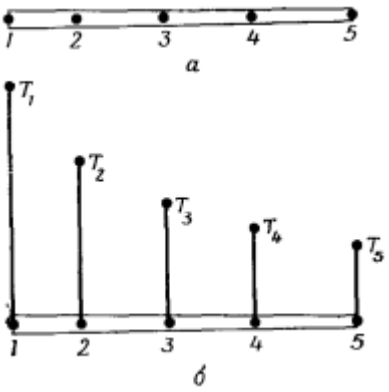


Рис. 2.2. Вузлові точки і передбачувані значення T(x)

Розбиття області на елементи може бути здійснено двома різними способами. Можна, наприклад, обмежити кожен елемент двома сусідніми вузловими точками, утворивши чотири елемента (рис. 2.3, а), або розбити область на два елементи, кожен з яких містить три вузла (рис. 2.3,б). Відповідний елементу поліном визначається за значеннями T(x) у вузлових точках елемента. У разі розбиття області на чотири елемента, коли на кожен елемент припадає по два вузли, функція елементу буде лінійна по x (дві точки однозначно визначають пряму лінію). Остаточна апроксимація T(x) буде складатися з чотирьох кусково-лінійних функцій, кожна з яких визначена на окремому елементі (рис. 2.4).

Інший спосіб розбиття області на два елементи з трьома опорними точками призводить до представлення функції елемента у вигляді полінома другого ступеня. У цьому разі остаточної апроксимацією T(x) буде сукупність двох кусково-неперервних квадратичних функцій. Це наближення буде саме кусково безперервним, так як кути нахилу обох графіків цих функцій можуть мати різні значення в третьому вузлі. В загальному випадку розподіл температури невідомий і потрібно визначити значення цієї величини в деяких точках. Методика побудови дискретної моделі залишається точно такою ж, як описано вище, але з додаванням одного додаткового кроку. Знову визначаються безліч вузлів і значення температури в цих вузлах T1, T2, T3,..., які тепер є змінними,

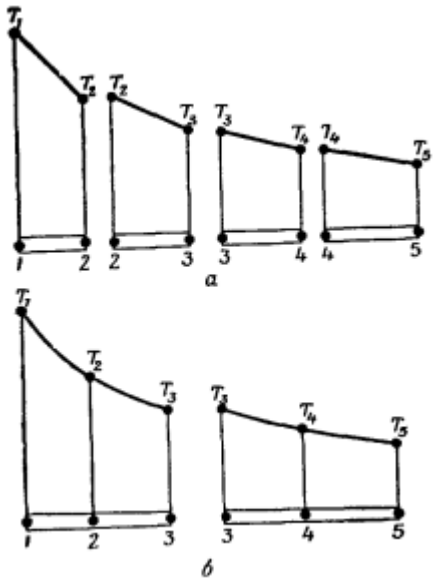


Рис. 2.3. Поділ області на елементи

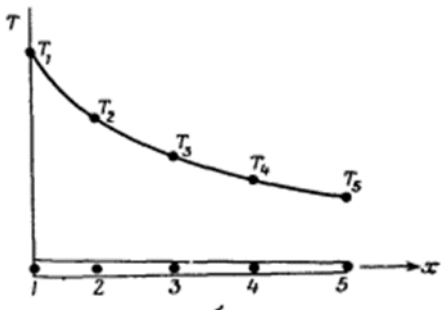


Рис. 2.4. Дискретні моделі для одновимірного температурного

так як вони заздалегідь невідомі. Область розбивається на елементи, на кожному з яких визначається відповідна функція елемента. Вузлові значення T(x) мають бути тепер «відрегульовані таким чином, щоб забезпечувалося «найкраще» наближення до істинного розподілу температури. Це регулювання здійснюється шляхом мінімізації деякої величини, пов'язаної з фізичною сутністю завдання. Якщо розглядається задача поширення тепла, то мінімізується функціонал, пов'язаний з відповідним диференціальним рівнянням. Процес мінімізації зводиться до розв'язування систем лінійних алгебраїчних рівнянь відносно вузлових значень T(x).

При побудові дискретної моделі безперервної величини, визначеної в двовимірній або тривимірній області, основна концепція методу кінцевих елементів використовується аналогічно[19-21]. У двовимірному випадку елементи описуються функціями від x, y, при цьому найчастіше розглядаються елементи у формі трикутника або чотирикутника. Функції елементів зображуються тепер плоскими (рис. 2.5) або криволінійними (рис. 2.6) поверхнями. Функція елемента буде представлятися площиною, якщо для даного

елемента взято мінімальне число вузлових точок, яке для трикутного елемента дорівнює трьом, а для чотирикутного — чотирьом[1].

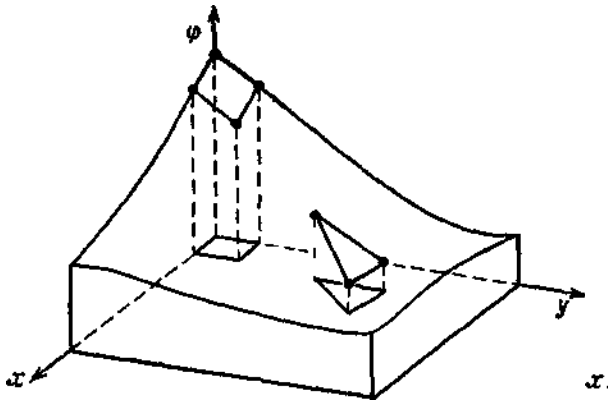


Рис. 2.5. Моделювання двовимірної скалярної функції за допомогою трикутних і чотирикутних елементів.

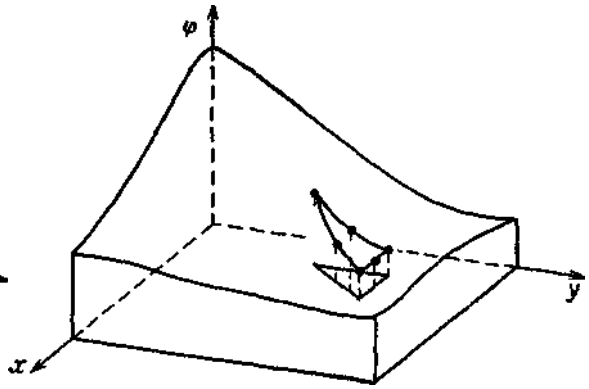


Рис. 2.6. Моделювання двовимірної скалярної функції за допомогою квадратичного трикутного елемента.

Якщо використовується число вузлів більше мінімального, то функціям елемента буде відповідати криволінійна поверхня. Крім того, надлишкове число вузлів дозволяє розглядати елементи з криволінійними границями. Остаточної апроксимацією двовимірної неперервної величини φ(x,y) буде служити сукупність кусково-неперервних поверхонь, кожна з яких визначається на окремому елементі за допомогою значень φ(x,y) у відповідних вузлових точках.

Важливим аспектом МСЕ є можливість виділити з набору елементів типовий елемент при визначенні функції елемента [22]. Це дозволяє визначати функцію елемента незалежно від відносного положення елемента в загальній зв'язковий моделі і від інших функцій елементів. Завдання функції елемента через довільне безліч вузлових значень координат дозволяє використовувати функції елемента для апроксимації геометрії області.

В даний час область застосування МСЕ дуже обширна і охоплює всі фізичні задачі, які можуть бути описані диференціальними рівняннями[23-24]. Найбільш важливими перевагами методу кінцевих елементів, завдяки яким він широко використовується, є наступні:

1. Властивості матеріалів суміжних елементів не обов'язково повинні бути однаковими. Це дозволяє застосовувати метод до тіл, складених з декількох матеріалів.

2. Криволінійна область може бути апроксимована за допомогою прямолінійних елементів або точно описана з допомогою криволінійних елементів. Таким чином, методом можна користуватися не лише для областей з «гарною» формою кордону.

3. Розміри елементів можуть бути змінними. Це дозволяє укрупнити або подрібнити мережі розбиття області на елементи, якщо в цьому є необхідність.

4. За допомогою методу кінцевих елементів не представляє праці розгляд граничних умов з розривною поверхневим навантаженням, а також змішаних граничних умов.

Зазначені вище переваги МСЕ можуть бути використані при складанні досить загальної програми для рішення задач певного класу. Наприклад, за допомогою програми для осиметричної задачі про поширення тепла можна вирішувати будь-яку задачу цього типу. Чинниками, що перешкоджають розширенню кола завдань, що вирішуються МСЕ, є обмеженість машинної пам'яті і висока вартість обчислювальних робіт.

Головний недолік МСЕ полягає в необхідності складання обчислювальних програм і застосування обчислювальної техніки. Обчислення, які потрібно проводити при використанні МСЕ, занадто громіздкі для ручного рахунку навіть у разі рішення дуже простих завдань. Для вирішення складних завдань необхідно використовувати швидкодіючу ЕОМ, що володіє великою пам'яттю.

Область застосування методу МСЕ простирається від аналізу напружень в конструкціях літаків або автомобілів до розрахунку таких складних систем, як атомна електростанція. З його допомогою розглядається рух рідини по трубах, через греблі, в пористих середовищах, вирішуються задачі електростатики і мастила, аналізуються коливання систем і багато іншого[25-26].

МСЕ, зазвичай на стадії дизайну та розробки продуктів, використовує багато дисциплін здебільшого з сім'ї механічної інженерії (таких як аеро,

морська, біометрична та автомобільна індустрії). Декілька сучасних МСЕ-пакетів включають спеціальні елементи, такі як термальні, електромагнітні, рідинні та структурні робочі середовища. В структурному моделюванні МСЕ дуже допомагає у генерації жорсткісних і силових візуалізацій у місцях зсувів та згинів, та відображання розповсюдження сил та зміщень.

МСЕ-програми забезпечують широкий спектр моделювальних можливостей контролю складності і модельовальної і аналітичної систем. За потреби в більшості інженерних програм можна змінювати бажаний рівень точності, час, потрібний для необхідних та асоційованих обчислень.

МСЕ дозволяє проектувати, відлагоджувати та оптимізовувати продукцію перед її випуском. Цей могутній засіб проектування відчутно покращив стандарти інженерних проектів та методологію цього процесу у багатьох сферах. Використання МСЕ зменшило час, за який продукт проходив від концепції до конвеєра. Його головною ідеєю було покращення початкових прототипів використовуючи МСЕ, що сприяло прискоренню їхнього тестування та розробки. В цілому, перевагами МСЕ є збільшення точності, покращення дизайну і краще бачення його критичних параметрів, створення віртуальних прототипів, зменшення кількості реальних прототипів, пришвидшення та здешевлення проектування, збільшення продуктивності та прибутковості[5].

МСЕ — числова техніка знаходження розв'язків інтегральних та диференціальних рівнянь у частинних похідних (ДРЧП). Процес розв'язання побудований або на повному усуненні диференціального рівняння для стаціонарних задач, або на розкладі ДРЧП в апроксимуючу систему звичайних диференціальних рівнянь, які потім розв'язуються використанням якої-небудь стандартної техніки, такої як метод Ейлера, Рунге-Кутти тощо.

При розв'язанні диференціальних рівнянь в частинних похідних головною метою є створення рівності, що апроксимує досліджувану рівність, і є стабільною, тобто помилки у вхідних даних і проміжних обчисленнях не акумулюються і не спричиняють беззмістовних результатів. Для реалізації цього є багато способів, кожен зі своїми плюсами і мінусами. МСЕ є добрим вибором

при розв'язуванні ДРЧП, які описують складні середовища; при змінності цих середовищ, коли бажана точність змінюється у різних ділянках середовища чи коли розв'язку не вистачає гладкості. Наприклад, при моделювання фронтального розбиття машини є можливість збільшити точність моделювання у важливіших зонах, таких, як передня частина машини, і зменшити її при обрахунку того, що відбудеться із задньою частиною машини (тим самим зменшивши ресурсоємність моделювання). Іншим прикладом може служити моделювання погоди на Землі, при якому важливішою є погода над сушею, ніж над безкраїми морськими просторами.

Інтерполяція — спосіб знаходження проміжних значень величини за наявним дискретним набором відомих значень. Багатьом із тих, хто стикається з науковими та інженерними розрахунками часто доводиться оперувати наборами значень, отриманих експериментальним шляхом чи методом випадкової вибірки.

Як правило, на підставі цих наборів потрібно побудувати функцію, зі значеннями якої могли б з високою точністю збігатися інші отримувані значення. Така задача називається апроксимацією кривої. Інтерполяцією називають такий різновид апроксимації, при якій крива побудованої функції проходить точно через наявні точки даних. Існує також близька до інтерполяції задача, що полягає в апроксимації якої-небудь складної функції іншою, простішою функцією. Якщо деяка функція занадто складна для продуктивних обчислень, можна спробувати обчислити її значення в декількох точках, а за ними побудувати, тобто інтерполювати, простішу функцію. Зрозуміло, використання спрощеної функції не дозволяє одержати такі ж точні результати, які давала б початкова функція.

Особливістю МСЕ є те, що розміри елемента і його орієнтація можуть бути вибрані так, як це необхідно, що дозволяє скласти загальні обчислювальні підпрограми, що включають різні елементи.

Для дослідження стаціонарної одновимірної задачі теплопровідності припустимо, що температура T залежить тільки від координати x. Основне диференціальне рівняння може бути записане у вигляді:

$$\frac{d}{dx}\left(k\frac{dT}{dx}\right)+S=0, \qquad (2.1)$$

де k – коефіцієнт теплопровідності;

S - джерельний член, який описує потужність тепловиділення в одиниці об'єму середовища.

Для початку припустимо, що k і S - сталі. З метою отримання чисельного розв'язку рівняння вибираємо множину точок уздовж осі x і шукаємо значення температури в них. Цю множину назвемо «розрахунковою сіткою», а точки - «розрахунковими точками». На рис. 2.7 показаний набір розрахункових точок, розміщених на однаковій відстані одна від одної і позначених як *i-1*, *i*, *i+1* і т.д. Відстань між сусідніми точками дорівнює dx. Задача чисельного методу полягає у визначенні температур T_{i-1} , T_i, T_{i+1} для будь-якого i[23].

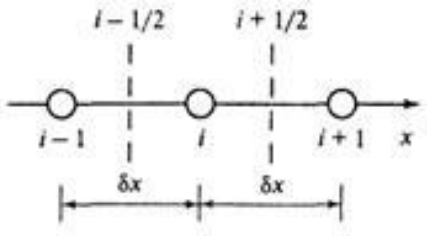


Рис. 2.7. Рівномірна розрахункова сітка для одновимірної задачі

Аналітичним розв'язком рівняння (2.1) є вираз для температури T, який залежить від x[13]. Чисельний розв'язок, навпаки, виходить у формі чисельних значень T в кінцевому числі розрахункових точок. Дискретні T_{i-1} , T_i, T_{i+1} для будь-якого i. знаходяться із системи алгебраїчних рівнянь, які називаються дискретними аналогами диференціального рівняння (2.1).

Коли розрахункова область містить невелике число розрахункових точок, дискретні аналоги є грубою апроксимацією диференціального рівняння. При цьому одержаний чисельний розв'язок звичайно не збігається з точним розв'язком диференціального рівняння. При збільшенні числа розрахункових точок чисельний розв'язок стає коректнішим і наближається до точного. Для

багатьох задач використання навіть невеликого числа розрахункових точок приводить до розв’язків, які достатньо точні для практичних цілей.

2.2. Опис використаної архітектури та шаблонів проектування

Для розрахунків в дипломній роботі використовується система Maple.

Maple здатна вирішити величезну кількість завдань взагалі без будь-якого програмування в загальноприйнятому розумінні цього поняття [13]. Достатньо лише описати алгоритм розв'язання задачі і розбити його на окремі запитання, на які система Maple здатна дати відповіді. Більше того, є тисячі задач, алгоритми рішення яких вже реалізовані у вигляді функцій і команд системи.

Ця система має вхідну мову надвисокого рівня, орієнтовану на вирішення математичних завдань практично будь-якої складності. Вона служить для завдання системи запитань або, інакше кажучи, завдання вхідних даних для подальшої їх обробки. Вхідна мова має велику кількість вбудованих математичних і графічних функцій, а також велику бібліотеку, що підключається по мірі необхідності [15].

Має Maple і свою мову процедурного програмування — Maple-мову. Ця мова має цілком традиційні засоби структурування програм: оператори циклів, оператори умовних і безумовних переходів, оператори порівняння, логічні оператори, команди керування зовнішніми пристроями, функції користувача, процедури і т.д. [15]. Ця мова також включає в себе всі команди і функції вхідної мови, йому доступні всі спеціальні оператори та функції. Багато з них є досить серйозними програмами, наприклад символьне диференціювання, інтегрування, розкладання в ряд Тейлора, побудова складних тривимірних графіків і т. д.

Синтаксис структурних операторів мови Maple нагадує суміш Бейсика і Паскаля. Це полегшує знайомство з ним тим, хто має хоча б початковий досвід програмування на цих мовах. За близьким до Бейсику правилами (і за допомогою загальноприйнятих математичних скорочень) виконується і введення математичних виразів у діалоговому режимі роботи з системою [17].

2.3. Опис використаних технологій та мов програмування

Maple — система комп'ютерної математики, розрахована на широке коло користувачів. До недавнього часу її називали системою комп'ютерної алгебри, це вказувало на особливу роль символьних обчислень і перетворень, які здатна здійснювати ця система. Але така назва звужує сферу застосування системи. Насправді вона вже здатна виконувати швидко і ефективно не тільки символьні, але і чисельні розрахунки, причому поєднує це з чудовими засобами графічної візуалізації та підготовки електронних документів[15].

Здавалося б, безглуздо називати таку потужну систему, як Maple 2017 математичної системою «для всіх». Однак у мірі її поширення вона стає корисною для багатьох користувачів ПК, які змушені в силу обставин (робота, навчання, хобі) займатися математичними обчисленнями і всім, що з ними пов'язано[18]. А все це тягнеться від вирішення навчальних завдань у вузах до моделювання складних фізичних об'єктів, систем і пристроїв, і навіть створення художньої графіки (наприклад, фракталів).

Maple — типова інтегрована система. Вона об'єднує в собі:

– потужну мову програмування (вона ж мова для інтерактивного спілкування з системою);

– редактор для підготовки та редагування документів і програм;

– сучасний багатовіконний інтерфейс з можливістю роботи в діалоговому режимі;

– потужну довідкову систему з багатьма тисячами прикладів;

– ядро алгоритмів і правил перетворення математичних виразів;

– чисельний і символьний процесори;

– систему діагностики;

– бібліотеки вбудованих і додаткових функцій;

– пакети функцій сторонніх виробників і підтримку деяких інших мов програмування та програм.

До всіх цих засобів є повний доступ прямо з програми. Maple — одна з найбільш потужних і «розумних» інтегрованих систем символьної математики, створена фірмою Waterloo Maple Inc.

Система Maple пройшла довгий шлях розвитку і апробації. Вона реалізована на великих ЕОМ, робочих станціях Sun, ПК, що працюють з операційною системою Unix, ПК класу IBM PC, Macintosh та ін. Все це самим позитивним чином вплинуло на її працю і надійність (в сенсі високу ймовірність правильності рішень і відсутності збоїв в роботі). Не випадково ядро системи Maple використовується цілим рядом інших потужних систем комп'ютерної математики, наприклад системами класу Mathcad і MATLAB.

Maple здатна вирішити величезну кількість завдань взагалі без будь-якого програмування в загальноприйнятому розумінні цього поняття. Достатньо лише описати алгоритм розв'язання задачі і розбити його на окремі запитання, на які система Maple здатна дати відповіді. Більше того, є тисячі задач, алгоритми рішення яких вже реалізовані у вигляді функцій і команд системи. Тим не менш, це зовсім не означає, що в Maple можна програмувати. Насправді Maple підтримує три власних мови: вхідний, реалізації та програмування.

Maple має вхідну мову надвисокого рівня, орієнтовану на вирішення математичних завдань практично будь-якої складності. Ця мова служить для завдання системи запитань або, інакше кажучи, завдання вхідних даних для подальшої їх обробки. Це мова інтерпретуючого типу і по своїй ідеології нагадує старий добрий Бейсік. І така подібність зовсім не недолік, а величезна перевага, адже саме з Бейсика почався справжній діалог користувача безпосередньо з комп'ютером! Вхідна мова має велику кількість вбудованих математичних і графічних функцій, а також велику бібліотеку, що підключається по мірі необхідності.

Має Maple і свою мову процедурного програмування, Maple-мову. Ця мова має цілком традиційні засоби структурування програм: оператори циклів, оператори умовних і безумовних переходів, оператори порівняння, логічні оператори, команди керування зовнішніми пристроями, функції користувача,

процедури і т. д. Вона також включає в себе всі команди і функції вхідної мови, їй доступні всі спеціальні оператори та функції. Багато з них є досить серйозними програмами, наприклад символьне диференціювання, інтегрування, розкладання в ряд Тейлора, побудова складних тривимірних графіків і т. д.

Не слід плутати вхідний мову і мову програмування системи (Maple-мова) з мовою її реалізації. Ним є один з найкращих і потужних універсальних мов програмування – Сі. На ньому написано ядро системи, що містить ретельно оптимізовані процедури. Більшість функцій, які містяться в пакетах, написані на Maple-мові, завдяки чому їх можна модифікувати і навіть писати свої власні бібліотеки. За різними оцінками, лише від 5 до 10 % коштів Maple створено на мові реалізації — все інше написано на Maple-мовою. Таким чином, система має розвинуті можливості для розширення та адаптації до завдань користувача. Для підготовки програм на мові Maple можуть використовуватися зовнішні редактори, але система має і свій вбудований редактор, що цілком задовольняє вимогам більшості користувачів. Цей редактор можна використовувати для редагування файлів програм або математичних виразів.

Maple — комерційна система комп'ютерної алгебри від компанії Waterloo Maple. Перша обмежена версія була розроблена та оприлюднено в грудні 1980-го року групою Symbolic Computation Group під керівництвом Кіта Ґеддеса з університету Во́терлу, місто Ватерлоо (Во́терлу), Онтаріо, Канада. Перший раз була продемонструвана на конференціях в 1982-ому році. До кінця 1983 року понад 50 університетів мали копії Maple, встановлені на їх машинах. Остання версія містить понад 5000 функцій для більшості розділів сучасної математики, моделювання та інтерактивної візуалізації, підтримує мову програмування Maple, і дозволяє комбінувати алгоритми, результати обчислення, математичні формули, текст, графіку, діаграми та анімацію зі звуком в електронному документі.

Основні можливості:

- символьні обчислення і чисельні методи;
- математичні функції та методи;

- розв'язування рівнянь;
- диференціальні рівняння;
- лінійна алгебра;
- програмування;
- операції з розмірностями та одиницями вимірювання величин;
- редактор математичних формул;
- візуалізація, графіки, інтерактивні меню та асистенти;
- шаблони-приклади для стандартних проблем;
- елементи для розробки графічних інтерфейсів;
- понад 30 палітр відсортованих для створення та редагування математичних виразів;
- інструментарій для фінансового моделювання;
- статистичне моделювання;
- фізичні моделі;
- високопродуктивні обчислення;
- інтерфейс для Matlab;
- експорт в інші мови програмування;
- системи доступу до баз даних.

Даний дипломний проект було реалізовано за допомогою операційної системи Windows 10 та системи комп’ютерної математики Maple.

Windows 10 — операційна система від компанії Microsoft для персональних комп'ютерів, ноутбуків, планшетів, лептопів-трансформерів і смартфонів. Оформлення Windows 10 засновано на принципах дизайну під назвою Metro, для якого притаманні прямокутні одноколірні форми, крупні шрифти, схематичні іконки та плавні ефекти переходів. У Windows 10 повернуто меню «Пуск», за структурою подібне до меню «Пуск» з версій до Windows 8, але з представленням додатків та інформації в плитках. Сповіщення від системи і додатків (поради, помилки, події календаря тощо) збираються в Центрі сповіщень, що являє собою панель з правого боку екрана, яка розгортається значком на панелі завдань. Додатково в Центрі сповіщень представлені елементи

управління режимами роботи комп'ютера, підключеннями до бездротових пристроїв та іншими параметрами системи.

Windows 10 має вдосконалену функцію мультивіконності Snap Assist, яка допомагає розподіляти простір екрана між вікнами. Вона дозволяє розташувати на робочому столі до чотирьох вікон одночасно. При цьому Windows 10 підказує, які ще додатки запущені в системі і як їх можна розмістити. В цій ОС додано функцію створення кількох робочих столів (подібну функцію можна побачити в Apple OS X та Ubuntu). Служба для входу до системи за допомогою біометричних даних Windows Hello дозволяє входити в систему за допомогою свого обличчя або в тих застосунках і сайтах, котрі її підтримують. Паралельно із Windows Hello Microsoft запускає систему, котра називається Microsoft Passport, що призначається для заміни пароля за допомогою особистих пристроїв, таких як смартфони, щоб можна було пройти аутентифікацію в корпоративних системах і онлайн-контенті.

Вимоги до системи:

- Процесор або система на чипі з мінімальною тактовою частотою 1 ГГц;
- Оперативна пам’ять: 1 ГБ (для 32-розрядної версії) або 2 ГБ (для 64-розрядної версії);
- Дисковий простір: Потрібні 16 ГБ (для 32-розрядної версії) або 20 ГБ (для 64-розрядної версії);
- Екран: 800x600.

2.4. Висновки

В даному розділі були проаналізовані методи для розрахунку розподілу температури в двувимірних тілах. Серед наведених методів найбільшу цінність та практичність має МСЕ. Розрахунок, який використовує механізм тріангуляції, дає можливість знайти розподіл температури, але результат може бути більш точним, якщо щільність сітки для тріангуляції буде більшою. Більша щільність

може привести до більш великого навантаження при розрахунку розподілу, тому треба розуміти, наскільки велика точність розподілу потрібна. Також було наведено рівняння теплопровідності, яке і задає параметри для розрахунку розподілу температури та може бути використане при розробці програмного забезпечення.

РОЗДІЛ 3.
РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ОБЧИСЛЕННЯ ТЕМПЕРАТУРНИХ ПОЛІВ В ДВУВИМІРНИХ ТІЛАХ

3.1. Інтерполяційні поліноми для дискретизованной області

Розглянемо особливості побудови лінійних інтерполяційних співвідношень, відповідних симплекс-елементів в системі Maple.

На рис. 3.1, рис. 3.2, рис.3.3 зображені:

- одновимірний симплекс-елемент, що представляє собою прямолінійний відрізок довжини L з двома вузлами, по одному на кожному кінці;

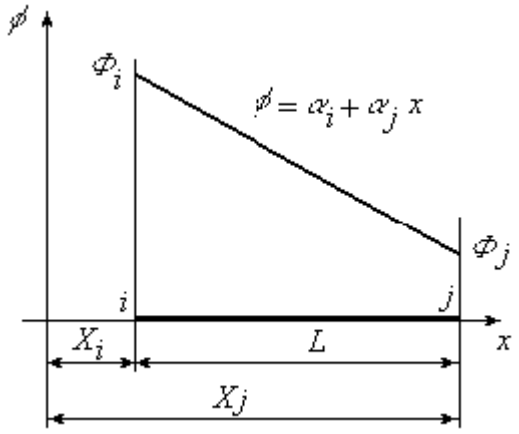


Рис. 3.1. Одновимірний симплекс-елемент

- двовимірний симплекс-елемент. Це трикутник з прямолінійними сторонами і трьома вузлами, по одному в кожній вершині;

- тривимірний симплекс-елемент. Він являє собою тетраедр. Чотири його вузла позначені індексами i,j,k,l, причому обхід вузлів i,j,k здійснюється в тому порядку, як вони записані, проти годинникової стрілки. Вузол l розташований у вершині, що знаходиться поза площиною вузлів i,j,k.

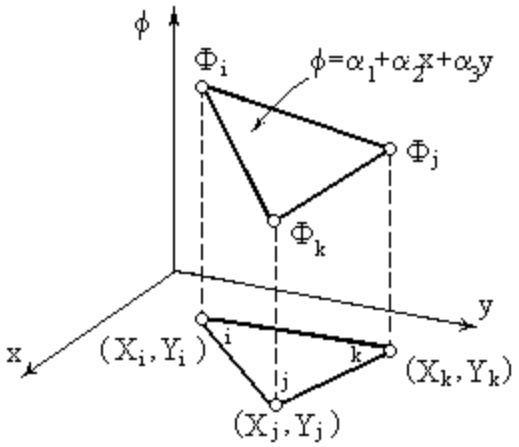


Рис. 3.2. Двовимірний симплекс-елемент

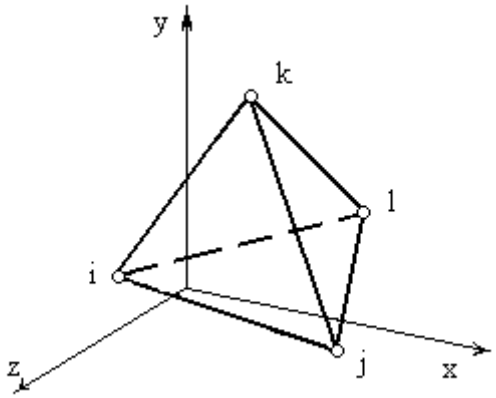


Рис. 3.3. Тривимірний симплекс-елемент

Загалом (у тривимірному випадку) лінійний інтерполяційний поліном для симплекс-елемента має вигляд:

> restart:with(linalg)

>phi = alpha[1]+alpha[2]*x+alpha[3]*y+alpha[4]*z

(У двовимірному випадку z = 0, а в одновимірному - z = 0 і y = 0).

Визначимо коефіцієнти:

>alpha[1], alpha[2], alpha[3], alpha[4]

з допомогою відомих значень шуканої функції у вузлових точках.

> P := [Phi[1], Phi[2], Phi[3], Phi[4]]

з рішення матричного рівняння:

>alpha= C^(-1)*Phi

де C- наступна матриця:

> C := matrix([[1, X[1], Y[1], Z[1]], [1, X[2], Y[2],...).

Об'єднавши в матрицю B множники при alpha в інтерполяційну формулу:

> B := [1, x, y, z],

отримаємо матрицю так званих функцій форми (базисних функцій):

> N := multiply(B,inverse(C)).

Тоді інтерполяційний поліном для шуканої функції може бути отриманий наступною операцією:

> phi := multiply(N,P).

Розглянемо одновимірний симплекс-елемент:

> restart:with(linalg):

> P := [Phi[1], Phi[2]]:

> C:=matrix([[1,X[1]],

[1,X[2]]]):

> B:=([1,x]):

> N:=multiply(B,inverse(C)):

> phi:=multiply(N,P):.

Масиви числових даних:

координати вузлів -

> X:=[1,0]:

значення функції у відповідних вузлових точках:

> Phi:=[40,34.]:.

Шуканий інтерполяційний поліном:

> phi;.

Зображуємо його на графіку (рис. 3.4):

> plot(phi,x=0..1).

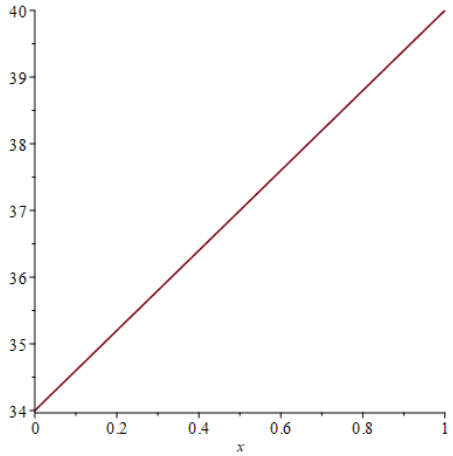


Рис. 3.4. Шуканий інтерполяційний поліном

В точці з координатою x = 0.5 маємо значення шуканої функції:

```
> subs(x=0.5,phi);
```

37.0

Отримаємо поліном, що відповідає двовимірному симплекс-елементу:

```
> P := [Phi[1], Phi[2], Phi[3]]:
> C:=matrix([[1,X[1],Y[1]],
[1,X[2],Y[2]],
[1,X[3],Y[3]] ]):
> B:=([1,x,y]):
> N:=multiply(B,inverse(C)):
> phi:=multiply(N,P):.
```

Визначаємо масиви числових даних:

координати вузлів -

```
> X:=[1,0,2]:Y:=[2,0,0]:.
```

Значення функції у відповідних вузлових точках:

```
> Phi:=[40,34.,26]:.
```

Шуканий інтерполяційний поліном буде мати вигляд:

```
> phi;
```

5.000000000*y+34.-4.00000000*x.

На відміну від тривимірного випадку, отриману апроксимуючу функцію можна представити наочно. Результат на рисунку 2.7.

```
> with(plots):
> polygonplot3d([
[X[1],Y[1],Phi[1]],
[X[2],Y[2],Phi[2]],
[X[3],Y[3],Phi[3]]],axes=boxed,style=PATCHCONTOUR);
```

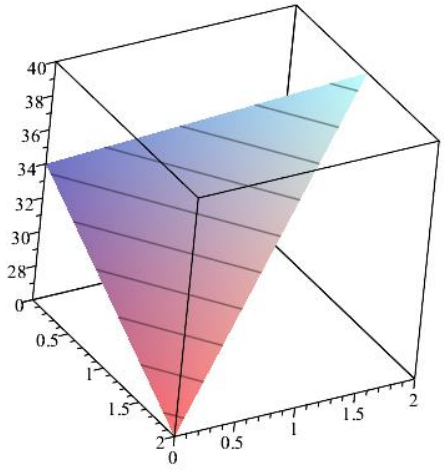


Рис. 3.5. Апроксимуюча функція

В точці з координатами (1, 0.5) маємо значення:

```
> subs(x=1,y=0.5,phi);
```

32.5.

Розглянемо тривимірний симплекс-елемент, що використовується для апроксимації розподілу неперервної функції всередині тетраедра.

Спочатку визначаємо масиви числових даних з відомими координатами чотирьох вузлів.

```
> X:=[1,0,2,1]:Y:=[2,0,0,0]: Z:=[1,0,0,3]:
```

Зобразимо цей тетраедр на екрані дисплея (рис. 3.6).

```
> with(plots):
> polygonplot3d([[X[1],Y[1],Z[1]],
[X[2],Y[2],Z[2]],
```

```
[X[3],Y[3],Z[3]],
[X[4],Y[4],Z[4]],
[X[2],Y[2],Z[2]],
[X[1],Y[1],Z[1]]],
axes=boxed,orientation=[-160,-100],
shading=XY,style=PATCHCONTOUR,light=[20,10,5,3,1],lightmodel='light3');
```

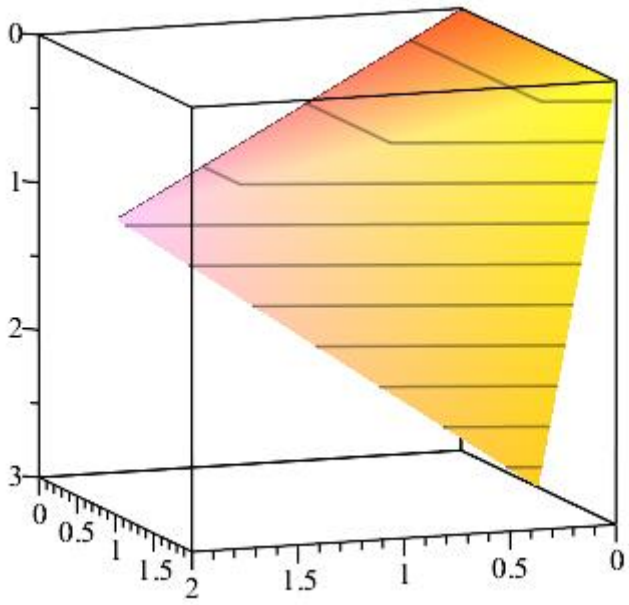


Рис. 3.6. Отриманий тетраедр

Нехай значеннями функції у відповідних вузлових точках будуть

```
> Phi := [40, 34, 26, 15].
```

Тоді шуканий інтерполяційний поліном прийме вигляд:

```
> phi;
7.500000003*y+34.-4.00000000*x-5.000000000*z.
```

У внутрішній точці з координатами (1, 0.5, 1) маємо значення функції:

```
> subs(x=1,y=0.5,z=1,phi);
28.75.
```

В даній роботі інтерполяційні співвідношення використані при розгляді скалярної величини. При інтерполяції векторних величин, наприклад переміщень у кожному вузлі, необхідно визначити більш однієї невідомої

(ступеня свободи). У цьому випадку векторну величину можна представити її компонентами, які розглядаються як невідомі скалярні величини. Кожен вузол буде містити одну, дві або три невідомі в залежності від того, яка задача розглядається - одновимірна, двомірна чи тривимірна. Причому в одновимірної задачі подання векторної та скалярної величин всередині елемента збігаються, так як в обох випадках у кожному вузлі знаходиться тільки одна невідома:

> phi := multiply(N,P).

У двомірній, тривимірної, або в загальному випадку n-мірної задачі необхідно апроксимувати кожну компоненту невідомою, наприклад:

> for i to n do phi[i] := multiply(N,P[i]) end do.

У загальному вигляді інтерполяційний поліном буде мати вигляд:

> restart:with(linalg):

> phi[e] = N[e]*Phi[e],

де e - індекс, який вказує на окремий елемент.

Техніку включення елемента двовимірного в область проілюструємо на прикладі простої пятиэлементной конфігурації, показаної на рисунку 3.7.

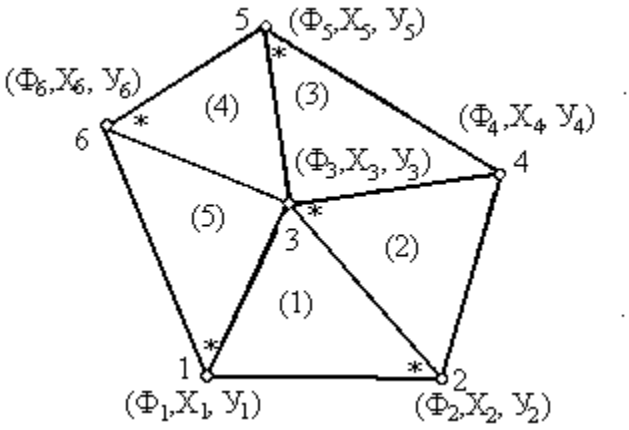


Рис. 3.7. Проста п’ятиелементна конфігурація

Визначимо числові масиви координат п'яти вузлів і значень шуканої функції у вказаних точках:

> X:=[1,3,2,4,2,0]: Y:=[0,0,1,1,2,1]: Phi:=[2.5,3,3,2,3.5,3]:

і складемо масив, що характеризує всю п'ятикутну область:

```
> G1:=array([
[X[1],Y[1],X[2],Y[2],X[3],Y[3],Phi[1],Phi[2],Phi[3]],
[X[3],Y[3],X[2],Y[2],X[4],Y[4],Phi[3],Phi[2],Phi[4]],
[X[5],Y[5],X[3],Y[3],X[4],Y[4],Phi[5],Phi[3],Phi[4]],
[X[6],Y[6],X[3],Y[3],X[5],Y[5],Phi[6],Phi[3],Phi[5]],
[X[1],Y[1],X[3],Y[3],X[6],Y[6],Phi[1],Phi[3],Phi[6]]
]);
```

$$G1 := \begin{bmatrix} 1 & 0 & 3 & 0 & 2 & 1 & 2.5 & 3 & 3 \\ 2 & 1 & 3 & 0 & 4 & 1 & 3 & 3 & 2 \\ 2 & 2 & 2 & 1 & 4 & 1 & 3.5 & 3 & 2 \\ 0 & 1 & 2 & 1 & 2 & 2 & 3 & 3 & 3.5 \\ 1 & 0 & 2 & 1 & 0 & 1 & 2.5 & 3 & 3 \end{bmatrix}$$

Задамо вектор коефіцієнтів (для обчислення базисних функцій) і вектор phi, в якому будуть накопичуватися поліноми, апроксимуючі безперервну функцію всередині кожного елемента:

```
> B:=vector([1,x,y]):phi:=vector(5):.
```

Оформимо цикл матричних обчислень:

```
> for e to 5 do
C:=matrix([[1,G1[e,1],G1[e,2]],
[1,G1[e,3],G1[e,4]],
[1,G1[e,5],G1[e,6]] ]):
N:=multiply(B,inverse(C)):
P:=([G1[e,7],G1[e,8],G1[e,9]]):
phi[e]:=multiply(N,P):
od:.
```

Вектор отриманих поліномів має вигляд:

```
> evalm(phi);
```

$$\left[2.250000000 + .250000000\,x + .250000000\,y,\ \frac{9}{2} - \frac{1}{2}x - \frac{1}{2}y,\ 3.5 + .5\,y - \frac{1}{2}x,\ 2.5 + .5\,y,\right.$$

$$\left.2.500000000 + .500000000\,y\right] .$$

Розподілу Ф представлений на рис. 3.8.

```
> with(plots):
list_point := [seq([
[G1[S,1],G1[S,2],G1[S,7]],
[G1[S,3],G1[S,4],G1[S,8]],
[G1[S,5],G1[S,6],G1[S,9]]
],S=1..5)]:
polygonplot3d(list_point, scaling=UNCONSTRAINED, axes=FRAMED, titlefont=[TIMES ROMAN,12], shading=Z,labels=[x,y,phi], style=hidden, orientation=[-50,40],lightmodel=light3).
```

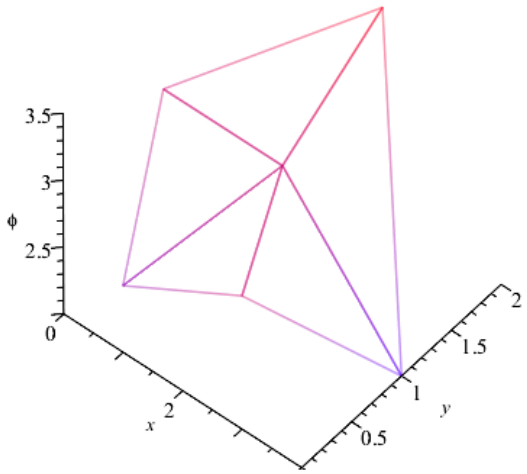


Рис. 3.8. Розподіл Ф

3.2. Одновимірна задача теплопровідності

Розглянемо одновимірний потік тепла в стержні з теплоізольованою бічною поверхнею (рис. 3.9).

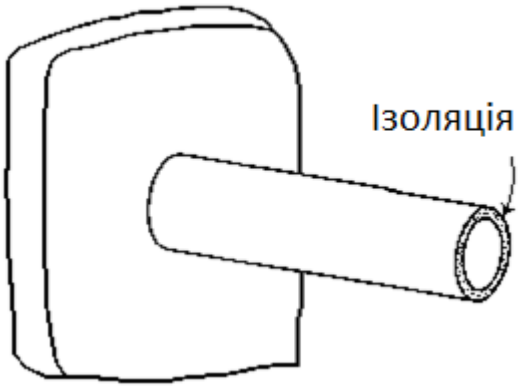


Рис. 3.9. Стержень з теплоізольованою бічною поверхнею

У варіаційному численні встановлюється, що для мінімізації функціоналу:

$$\chi = \int \frac{K\left(\frac{\partial}{\partial x}\mathrm{T}(x)\right)^2}{2} dV + \int q\,T + \frac{1\,\alpha\,(T - T_w)2}{2} dS \qquad (3.1)$$

Необхідно, щоб задовольнялося диференціальне рівняння:

$$K\left(\frac{\partial^2}{\partial x^2}\mathrm{T}(x)\right) = 0 \quad . \qquad (3.2)$$

З граничними умовами:

$$K\left(\frac{\partial}{\partial x}\mathrm{T}(x)\right) + q + \alpha\,(T - T_w) = 0 \qquad (3.3)$$

де T – температура;

K - коэффіцієнт теплопровідності;

q -тепловий потік заданної інтенсивності;

α - коэффіцієнт конвективного теплообміну;

T_w - температура навколишнього середовища;

V - об `єм стержня;

S - площа поперечного перерізу стержня;

x - координата вздовж осі стержня.

Таким чином, будь розподіл $T(x)$, при якому функціонал X стає мінімальним, є рішенням задачі теплопровідності (3.1)-(3.3):

```
> restart:
> with(linalg):.
```

Нехай стержень є дискретною моделлю, що містить шість вузлових точок:

```
> m:=6:
Phi:=vector(m):
```

з координатами:

```
> X:=[0,1,2,3,4,5]:
```

Нехай площа поперечного перерізу стержня буде:

```
> A:=Pi:
```

В граничних точках відомі значення температури:

```
> Phi[1]:=100:Phi[6]:=0:
```

Наступна частина програмного коду формує робочий масив даних G1:

```
> for i to m do
q[i]:=0: alpha[i]:=1: Tw[i]:=0:
od:
n:=m-1:
G1:=array(1..n,1..11):
for i to n do
K[i]:=1:
G1[i,1]:=X[i]: G1[i,2]:=X[i+1]:
G1[i,3]:=Phi[i]: G1[i,4]:=Phi[i+1]:
G1[i,5]:=q[i]: G1[i,6]:=q[i+1]:
G1[i,7]:=alpha[i]:G1[i,8]:=alpha[i+1]:
G1[i,9]:=Tw[i]: G1[i,10]:=Tw[i+1]:G1[i,11]:=K[i]:
od:
print (`число элементов -` ,n);
```

число элементов -, 5

```
> evalm(G1);
```

$$\begin{bmatrix} 0 & 1 & 100 & \Phi_2 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 2 & \Phi_2 & \Phi_3 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 2 & 3 & \Phi_3 & \Phi_4 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 3 & 4 & \Phi_4 & \Phi_5 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 4 & 5 & \Phi_5 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

В окремих вузлах можна визначити тепловий потік (за замовчуванням для інших і вузлів q[i]=0). У вузлах контактують з навколишнім середовищем можна задати температуру останньої Tw (за замовчуванням Tw[i]=0) і відомі коефіцієнти тепловіддачі навколишнього середовища (за замовчуванням alpha[i]=1). Також можна ввести коефіцієнти теплопровідності в окремих елементах(за замовчуванням K[j]=1).

В наступній комірці автоматично обчислюються інтерполяційні поліноми і функціонали для кожного з n окремих елементів, в результаті виходить сумарний функціонал.

```
> B:=vector([1,x]):
eqns:=NULL:ch:=0:

for e to n do

C:=matrix([[1,G1[e,1]],
[1,G1[e,2]] ]):
N:=multiply(B,inverse(C)):
P:=([G1[e,3],G1[e,4]]):
phi:=multiply(N,P):

chi:=int((1/2)*K[e]*(diff(phi,x))^2*A,x=G1[e,1]..G1[e,2]):

if q[e]<>0 or q[e+1]<>0 then
```

```
chi:=chi+sum( 'int(q[k]*Phi[k],s=0..A)','k'=e..e+1):
fi:
if Tw[e]<>0 or Tw[e+1]<>0 then
chi:=chi+sum('int((alpha[k]/2)*(Phi[k]-Tw[k])^2,s=0..A)',
'k'=e..e+1):
fi:

ch:=ch+chi:

od:
ch;
```

$$\frac{1}{2}(-100+\Phi_2)^2\pi+\frac{1}{2}(-\Phi_2+\Phi_3)^2\pi+\frac{1}{2}(-\Phi_3+\Phi_4)^2\pi+\frac{1}{2}(-\Phi_4+\Phi_5)^2\pi+\frac{1}{2}\Phi_5{}^2\pi$$

Мінімізація функціоналу:

```
> var1:=NULL:
for e to m do
if type(Phi[e],integer) then
eq1:=0=0
else
eq1:=diff(ch,Phi[e])=0:
var1:=var1,Phi[e]:
fi:
eqns:=eqns,eq1:
od:.
```

Система лінійних алгебраїчних рівнянь:

```
> eqns:={eqns}:var:={var1}:
```

рішення якої дає невідомі значення температури у вузлах:

```
> sols := solve( eqns,var);
```

$$sols := \{\Phi_3 = 60, \Phi_2 = 80, \Phi_4 = 40, \Phi_5 = 20\}$$

Процедура сортування обчислених значень має вигляд:

```
> sorty:=proc(Phi,sols,m)
local i,j:
for i to nops(sols) do
for j to m do
if Phi[j]<>op(1,sols[i]) then next
else Phi[j]:=op(2,sols[i])
fi:
od:
od:
end:
sorty(Phi,sols,m):
evalm(Phi);
evalm(G1):
```

[100, 80, 60, 40, 20, 0] .

Графік розподілу температури прдставлений на рис. 2.12:

```
> with(plots):
f_1:=plot([seq([X[i], Phi[i]], i = 1 .. nops(X))], style = point, symbol = circle, color =
red):display(f_1);
```

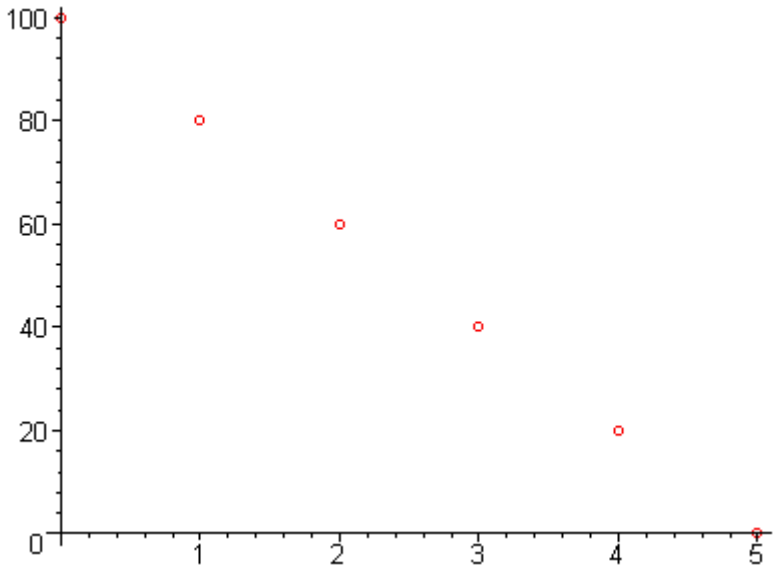


Рис. 3.10. Графік значень температури у вузлах

Розглянуте рішення простого прикладу може бути легко перевірена аналітично за допомогою засобів Maple. Задаємо диференціальне рівняння і граничні умови:

```
> with(DEtools):
du1:=diff(T_0(x),x,x)=0;
```

$$du1 := \frac{\partial^2}{\partial x^2} T\_0(x) = 0$$

```
in1:=T_0(0)=100,T_0(5)=0;
```

$$in1 := T\_0(0) = 100, T\_0(5) = 0$$,

отримуємо рішення

```
> qq:=dsolve({du1,in1},{T_0(x)}):
> T_a:=subs(qq,T_0(x));
```

$$T\_a := -20\,x + 100$$

Тепер визначаємо параметри графіка:

```
> f_2:=plot(T_a,x=0..5,title=`Проверка`):
```

Тоді чисельне і аналітичне рішення представлено на рис. 2.18:

```
> display({f_1,f_2})
```

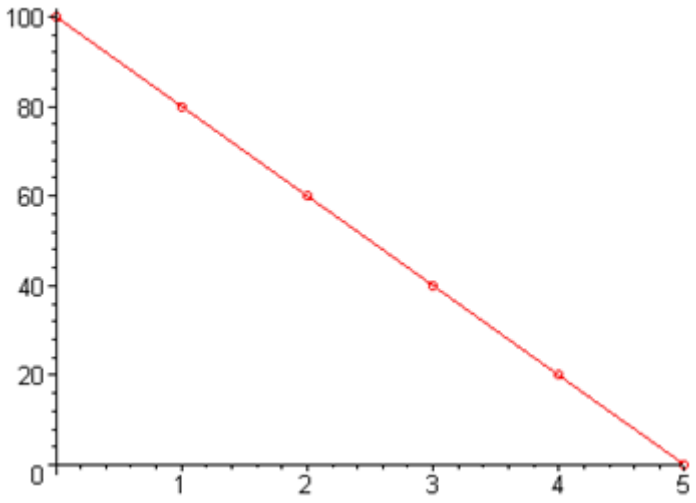


Рис. 3.11. Температури у вузлах, які отримані чисельним і аналітичним методами.

Аналітичне та чисельне рішення повністю збігаються.

3.3. Двовимірна задача теплопровідності

Розглянемо задачу теплопровідності у двовимірній постановці. З варіаційної точки зору пошук мінімуму функціонала:

$$\chi = \int \frac{1\left(K_x\left(\frac{\partial}{\partial x}\mathrm{T}(x,y)\right)^2 + K_y\left(\frac{\partial}{\partial y}\mathrm{T}(x,y)\right)^2 - 2\,Q\,\mathrm{T}(x,y)\right)}{2} dV + \int q\,\mathrm{T}(x,y) + \frac{1\,\alpha\,(\mathrm{T}(x,y) - T_w)2}{2} dS \qquad (3.4)$$

еквівалентно рішенням диференціального рівняння теплопровідності

$$\left(\frac{\partial}{\partial x}K_x\left(\frac{\partial}{\partial x}\mathrm{T}(x,y)\right)\right) + \left(\frac{\partial}{\partial y}K_y\left(\frac{\partial}{\partial y}\mathrm{T}(x,y)\right)\right) + Q = 0 \qquad (3.5)$$

з граничними умовами

$$K_x\left(\frac{\partial}{\partial x}\mathrm{T}(x,y)\right)l_x + K_y\left(\frac{\partial}{\partial y}\mathrm{T}(x,y)\right)l_y + q + \alpha\,(T_{x,y} - T_w) = 0 \qquad (3.6)$$

де T – температура;

k - коефіцієнт теплопровідності;

q - тепловий потік заданої інтенсивності;

Q - внутрішній тепловий джерело або стік;

a - коефіцієнт конвективного теплообміну;

$\mathrm{T_w}$ - температура навколишнього середовища;

l - направляючі косинуси вектора нормалі до поверхні;

V – обсяг;

S - площа поверхні;

x,y – координати.

Таким чином, будь-який розподіл $T(x,y)$, при якому функціонал X стає мінімальним, являється рішенням задачі теплопровідності (3.4)- (3.6).

Нехай область має вигдяд (рис. 2.19):

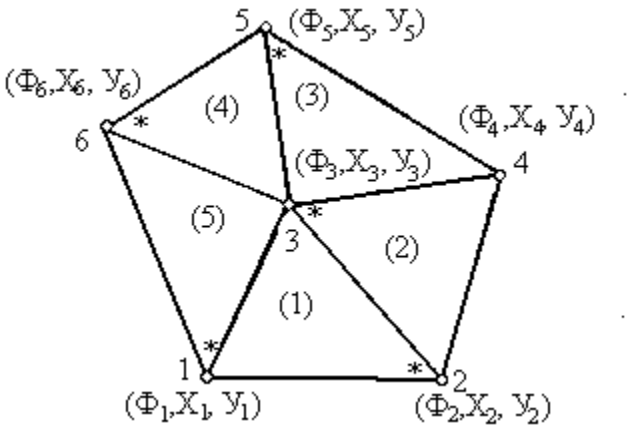


Рис. 3.12. П’ятиелементна конфігурація

Задаємо число вузлових точок:

>restart; with(linalg)

>m:=6:

і число елементів:

>n:=5:.

Вводимо координати вузлових точок:

> X:=[1,3,2,4,2,0]: Y:=[0,0,1,1,2,1]:.

Дані про структуру області можуть бути представлені номерами вузлів кожного з елементів, які вводяться в обході по годинникової стрілки, починаючи з вузла, позначеної на малюнку зірочкою.

> G1:=array(1..n,1..25,

[

[2,3,1],

[3,2,4],

[5,3,4],

[6,3,5],

[1,3,6]

]):

Тоді розрахункова сітка має вигляд (рис. 3.13).

>PLOT(POLYGONS([[X[G1[e, 1]], Y[G1[e, 1]]], [X[G1[e, 2]], Y[G1[e, 2]]], [X[G1[e, 3]], Y[G1[e, 3]]]]$e = 1 .. n)).

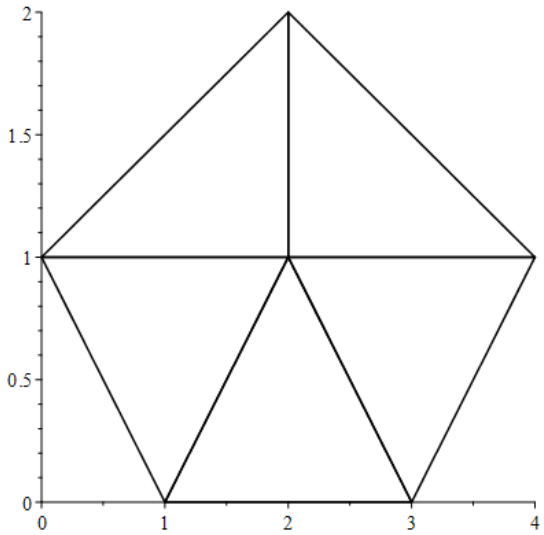


Рис. 3.13. Розрахункова сітка

Обчислимо площі кожного з елементів:

> for e to n do A[e] := X[G1[e, 1]]*Y[G1[e, 2]]-X[G1[e, 1]]*Y[G1[e, 3]]-X[G1[e, 2]]*Y[G1[e, 1]]+X[G1[e, 2]]*Y[G1[e, 3]]+X[G1[e, 3]]*Y[G1[e, 1]]-X[G1[e, 3]]*Y[G1[e, 2]] end do.

Визначимо параметри фізичних властивостей розрахункової області за замовчуванням:

> Phi:=vector(m);

for i to m do

q[i]:=0: Q[i]:=0: alpha[i]:=1: Tw[i]:=0:

od:

$\Phi := \mathrm{array}(1 .. 6, [\ ])$

Вузлові температури відомі:

> Phi[1]:=100.:Phi[4]:=.0:

Теплові потоки відомі у вузлах розрахункової сітки:

> Q[3]:=-10:.

Можна задати відомі температури навколишнього середовища для елементів, з нею контактують (наприклад, Tw[2]=20) і відповідні коефіцієнти

тепловіддачі навколишньому середовищі (наприклад alpha[2]=5). Коефіцієнти теплопровідності окремих елементів можуть бути відмінними від прийнятих за замовчуванням (наприклад, K[3]=1.3).

Робочий масив даних:

```
for i to n do K[i] := 1;
G1[i, 4] := X[G1[i, 1]];G1[i, 5] := Y[G1[i, 1]];
G1[i, 6] := X[G1[i, 2]]; G1[i, 7] := Y[G1[i, 2]];
G1[i, 8] := X[G1[i, 3]]; G1[i, 9] := Y[G1[i, 3]];
G1[i, 10] := Phi[G1[i, 1]]; G1[i, 11] := Phi[G1[i, 2]];
G1[i, 12] := Phi[G1[i, 3]]; G1[i, 13] := q[G1[i, 1]];
G1[i, 14] := q[G1[i, 2]]; G1[i, 15] := q[G1[i, 3]];
G1[i, 16] := alpha[G1[i, 1]]; G1[i, 17] := alpha[G1[i, 2]];
G1[i, 18] := alpha[G1[i, 3]]; G1[i, 19] := Tw[G1[i, 1]];
G1[i, 20] := Tw[G1[i, 2]]; G1[i, 21] := Tw[G1[i, 3]];
G1[i, 22] := K[i]; G1[i, 23] := Q[G1[i, 1]]; G1[i, 24] := Q[G1[i, 2]];
G1[i, 25] := Q[G1[i, 3]] end do;
evalm(G1).
```

$$\begin{bmatrix}
2 & 3 & 1 & 3 & 0 & 2 & 1 & 1 & 0 & \Phi_2 & \Phi_3 & 100. & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & -10 & 0 \\
3 & 2 & 4 & 2 & 1 & 3 & 0 & 4 & 1 & \Phi_3 & \Phi_2 & 0. & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & -10 & 0 & 0 \\
5 & 3 & 4 & 2 & 2 & 2 & 1 & 4 & 1 & \Phi_5 & \Phi_3 & 0. & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & -10 & 0 \\
6 & 3 & 5 & 0 & 1 & 2 & 1 & 2 & 2 & \Phi_6 & \Phi_3 & \Phi_5 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & -10 & 0 \\
1 & 3 & 6 & 1 & 0 & 2 & 1 & 0 & 1 & 100. & \Phi_3 & \Phi_6 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & -10 & 0
\end{bmatrix}$$

Наступний фрагмент програмного коду виробляє автоматичне обчислення інтерполяційних поліномів і функціоналів для кожного з n окремих елементів. У підсумку виходить сумарний функціонал.

```
> B:=vector([1,x,y]):
eqns:=NULL:ch:=0:
for e to n do
```

```
C:=matrix([[1,G1[e,4],G1[e,5]],
[1,G1[e,6],G1[e,7]],
[1,G1[e,8],G1[e,9]] ]):
N:=multiply(B,inverse(C)):
P:=([G1[e,10],G1[e,11],G1[e,12]]):
phi:=multiply(N,P):
chi:=int((1/2)*(K[e]*((diff(phi,x))^2+K[e]*(diff(phi,y))^2)*A[e]),z=0..1:
if G1[e,23]<>0 or G1[e,24]<>0 or G1[e,25]<> 0 then
chi:=chi-sum( 'int((1/2)*G1[e,k]*G1[e,k-13],z=0..1)','k'=23..25):
fi:
if G1[e,13]<>0 or G1[e,14]<>0 or G1[e,15]<> 0 then
chi:=chi+sum( 'int(G1[e,k]*G1[e,k-3],s=0..A[e])','k'=13..15):
fi:
if G1[e,19]<>0 or G1[e,20]<>0 or G1[e,21]<>0 then
chi:=chi+sum('int((G1[e,k-3]/2)*(G1[e,k-9]-G1[e,k])^2,s=0..A[e])',
'k'=19..21):
fi:
ch:=ch+chi:
od:
ch;
```

$$(.5000000000\,\Phi_2 - 50.)^2 + (-.5000000000\,\Phi_2 + \Phi_3 - 50.)^2 + 25\,\Phi_3 + \frac{1}{2}\Phi_3{}^2 + \left(\frac{1}{2}\Phi_3 - \Phi_2\right)^2 + 2\,(\Phi_5 - \Phi_3)^2$$

$$+\left(-\frac{1}{2}\Phi_6 + \frac{1}{2}\Phi_3\right)^2 + (-.5000000000\,\Phi_6 + .5000000000\,\Phi_3)^2$$

$$+ (-100. + .5000000000\,\Phi_3 + .5000000000\,\Phi_6)^2$$

.

Здійснюємо мінімізацію функціонала:

```
var1 := NULL;
for i to m do if type(Phi[i], integer) or type(Phi[i], float) then eql := 0 = 0 else eql :=
d*ch/dPhi[i] = 0; var1 := var1, Phi[i] end if; eqns := eqns, eql end do.
```

Формуємо підсумкову систему лінійних алгебраїчних рівнянь (СЛАР):

```
> eqns := {eqns}; var := {var1}
```

$$eqns := \{-2.000000000\,\Phi_2 + 9.000000000\,\Phi_3 - 175.0000000 - 4\,\Phi_5 - .5000000000\,\Phi_6 = 0,\ 4\,\Phi_5 - 4\,\Phi_3 = 0,$$

$$1.500000000\,\Phi_6 - .5000000000\,\Phi_3 - 100.0000000 = 0,\ 0 = 0,\ 3.000000000\,\Phi_2 - 2.000000000\,\Phi_3 = 0\}$$

і вирішуємо її:

```
> sols := solve*{eqns, var};
```

$$sols := \{\Phi_6 = 86.50793651,\ \Phi_5 = 59.52380952,\ \Phi_2 = 39.68253968,\ \Phi_3 = 59.52380952\}$$

Таким чином отримані невідомі значення температури у відповідних вузлах. Процедура сортування та виведення підсумкового вектора вузлових значень:

```
> sorty := proc (Phi, sols, m) local i, j;
for i to nops(sols) do for j to m do if Phi*{j} <> op(1, sols[i]) then next else Phi[j] :=
op(2, sols[i]) end if end do end do end proc; sorty(Phi, sols, m);
evalm(Phi);
evalm(G1);
```

[100., 39.68253968, 59.52380952, 0., 59.52380952, 86.50793651]

Значення нев'язок:

```
> op(eqns);
```

$$-.6\ 10^{-7} = 0,\ 0 = 0,\ 0. = 0\ .$$

Тоді графік розподілу температури має вигляд (рис. 2.16):

```
> with(plots):
list_poy := [seq([
[G1[S,4],G1[S,5],G1[S,10]],
[G1[S,6],G1[S,7],G1[S,11]],
[G1[S,8],G1[S,9],G1[S,12]]
```

],S=1..n)]:

polygonplot3d(list_poy, scaling=UNCONSTRAINED, axes=FRAMED, titlefont=[TIMES,ROMAN,12], shading=ZGREYSCALE,labels=[x,y,T], style=PATCHCONTOUR, orientation=[-20,70]);

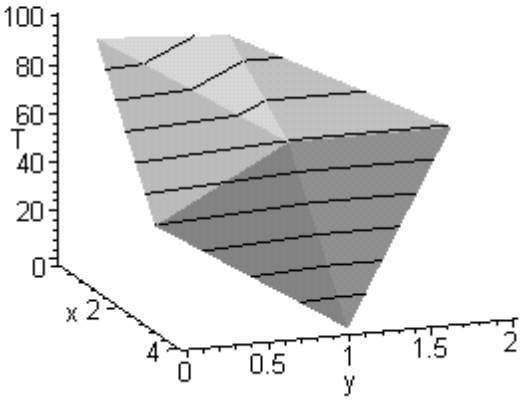


Рис. 3.14. Графік розподілу температури

3.4. Порівняння з аналітичним рішенням

До звичайних помилок округлення та апроксимації, властивим майже будь-обчислювальної процедури в методі кінцевих елементів мають місце:

а) помилки, що є результатом відмінностей геометричних межі області визначення і її звичайно-елементним наближенням;

б) помилки, обумовлені різницею між точним розв'язком задачі в частинних похідних і за його поданням пробної функцією.

Розглянемо рішення рівняння Лапласа на одиничному квадраті з граничними умовами, зазначеними на рис. 3.15.

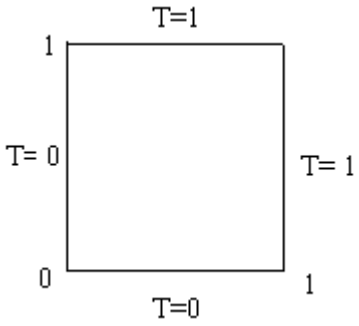


Рис. 3.15. Одиничний квадрат з граничними умовами

При вирішенні задачі методом кінцевих елементів число вузлових точок - 25, число елементів -32:

> m:=25:

> n:=32:.

Результат на рисунку 3.16.

Координати вузлових точок:

> X:=[0,0.25,0.5,0.75,1,

0,0.25,0.5,0.75,1,

0,0.25,0.5,0.75,1,

0,0.25,0.5,0.75,1,

0,0.25,0.5,0.75,1]:

Y:=[0,0,0,0,0,

0.25,0.25,0.25,0.25,0.25,

0.5,0.5,0.5,0.5,0.5,

0.75,0.75,0.75,0.75,0.75,

1,1,1,1,1]:

Структура області:

> G1:=array(1..n,1..25,

[

[1,2,6], [2,7,6], [2,3,7],[3,8,7],[3,4,8],[4,9,8],[4,5,9],[5,10,9],

[6,7,11],[7,12,11],[7,8,12],[8,13,12],[8,9,13],[9,14,13],[9,10,14],

[10,15,14],[11,12,16],[12,17,16],[12,13,17],[13,18,17],[13,14,18],

[14,19,18],[14,15,19],[15,20,19],[16,17,21],[17,22,21],[17,18,22],
[18,23,22],[18,19,23],[19,24,23],[19,20,24],[20,25,24]
]):

> PLOT(POLYGONS([[X[G1[e,1]],Y[G1[e,1]]], [X[G1[e,2]],Y[G1[e,2]]],
[X[G1[e,3]],Y[G1[e,3]]]] $e=1..n)).

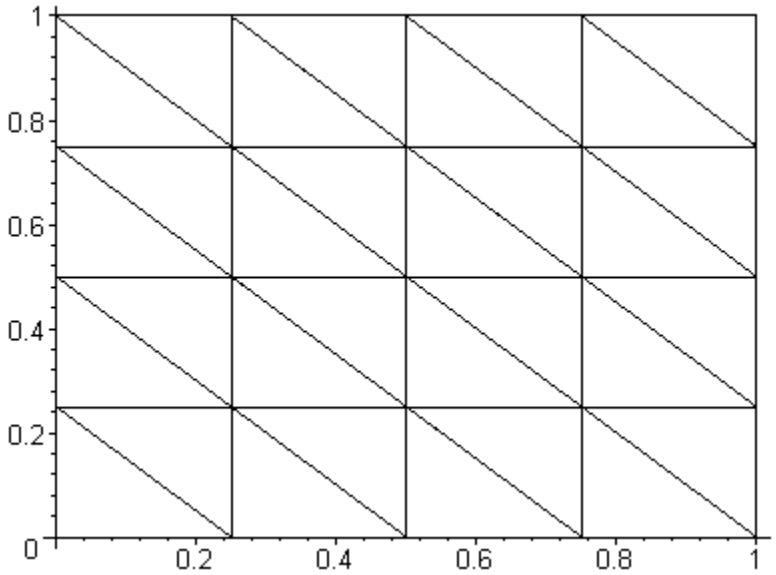


Рис. 3.16. Кінцево-елементна розрахункова сітка

Параметри за замовчуванням:

> Phi:=vector(m):

for i to m do

q[i]:=0: Q[i]:=0: alpha[i]:=1: Tw[i]:=0:

od:

Введення відомих значень функції в точках на межі:

> Phi[1]:=0.:Phi[2]:=.0:Phi[3]:=0:Phi[4]:=0: Phi[5]:=0.5:

Phi[6]:=0.:Phi[15]:=1.:Phi[11]:=0.:Phi[20]:=1.:

Phi[16]:=0.:Phi[10]:=1.:

Phi[21]:=0.5:Phi[22]:=1:Phi[23]:=1:Phi[24]:=1:Phi[25]:=1.:.

Розрахунковий блок кінцево-елементного методу:

> for e to n do

A[e]:=(X[G1[e,2]]*Y[G1[e,3]]+X[G1[e,3]]*Y[G1[e,1]]+X[G1[e,1]]*Y[G1[e,2]]-

```
X[G1[e,2]]*Y[G1[e,1]]-X[G1[e,3]]*Y[G1[e,2]]-X[G1[e,1]]*Y[G1[e,3]]):
od:
for i to n do
K[i]:=1:
G1[i,4]:=X[G1[i,1]]: G1[i,5]:=Y[G1[i,1]]:
G1[i,6]:=X[G1[i,2]]: G1[i,7]:=Y[G1[i,2]]:
G1[i,8]:=X[G1[i,3]]: G1[i,9]:=Y[G1[i,3]]:
G1[i,10]:=Phi[G1[i,1]]:G1[i,11]:=Phi[G1[i,2]]:G1[i,12]:=Phi[G1[i,3]]:
G1[i,13]:=q[G1[i,1]]:G1[i,14]:=q[G1[i,2]]:G1[i,15]:=q[G1[i,3]]:
G1[i,16]:=alpha[G1[i,1]]:G1[i,17]:=alpha[G1[i,2]]:
G1[i,18]:=alpha[G1[i,3]]:
G1[i,19]:=Tw[G1[i,1]]:G1[i,20]:=Tw[G1[i,2]]:G1[i,21]:=Tw[G1[i,3]]:
G1[i,22]:=K[i]:
G1[i,23]:=Q[G1[i,1]]:G1[i,24]:=Q[G1[i,2]]:G1[i,25]:=Q[G1[i,3]]:
od:
evalm(G1):
B:=vector([1,x,y]):
eqns:=NULL:ch:=0:
for e to n do
C:=matrix([[1,G1[e,4],G1[e,5]],
[1,G1[e,6],G1[e,7]],
[1,G1[e,8],G1[e,9]] ]):
N:=multiply(B,inverse(C)):
P:=([G1[e,10],G1[e,11],G1[e,12]]):
phi:=multiply(N,P):
chi:=int((1/2)*(K[e]*((diff(phi,x))^2+K[e]*(diff(phi,y))^2)*A[e]),z=0..1):
if G1[e,23]<>0 or G1[e,24]<>0 or G1[e,25]<> 0 then
chi:=chi-A[e]*sum( 'int((1/2)*G1[e,k]*G1[e,k-13],z=0..1)','k'=23..25):
fi:
if G1[e,13]<>0 or G1[e,14]<>0 or G1[e,15]<> 0 then
```

```
chi:=chi+sum( 'int(G1[e,k]*G1[e,k-3],s=0..A[e])','k'=13..15):
fi:
if G1[e,19]<>0 or G1[e,20]<>0 or G1[e,21]<>0 then
chi:=chi+sum('int((G1[e,k-3]/2)*(G1[e,k-9]-G1[e,k])^2,s=0..A[e])',
'k'=19..21):
fi:
ch:=ch+chi:
od:
ch:
var1:=NULL:
for i to m do
if type(Phi[i],integer) or type(Phi[i],float) then
eq1:=0=0
else
eq1:=diff(ch,Phi[i])=0:
var1:=var1,Phi[i]:
fi:
eqns:=eqns,eq1:
od:
eqns:={eqns}:var:={var1}:
sols := solve( eqns,var):
sorty:=proc(Phi,sols,m)
local i,j:
for i to nops(sols) do
for j to m do
if Phi[j]<>op(1,sols[i]) then next
else Phi[j]:=op(2,sols[i])
fi:
od:
od:
```

```
end:
sorty(Phi,sols,m):
evalm(Phi):
evalm(G1):
```

Матриця знайдених значень невідомих вузлів:

```
> T_num:=matrix([[Phi[7],Phi[8],Phi[9]],
[Phi[12],Phi[13],Phi[14]],
[Phi[17],Phi[18],Phi[19]]]);
```

$$T\_num := \begin{bmatrix} .1428571429 & .2857142857 & .5000000000 \\ .2857142857 & .5000000000 & .7142857143 \\ .5000000000 & .7142857143 & .8571428571 \end{bmatrix}$$

Опис графічного представлення матриці рішень:

```
> FIG[1]:=matrixplot(T_num,axes=FRAMED,style=PATCHCONTOUR,title=`МКЭ
`,labels=[x,y,T], orientation=[-10,90]):
```

Аналітичне рішення знаходиться за методом Фур'є, на основі розділення змінних і застосування принципу суперпозиції. При цьому знаходиться формальне рішення, тобто рішення у вигляді нескінченного ряду, у якому кожен член є рішення рівняння Лапласа і задовольняє поставленим граничним умовам. Кількість внутрішніх вузлів по кожній з осей координат:

```
> Nn:=3:
```

Число членів ряду, що використовуються для побудови рішення:

```
> Ns:=40:
```

Блок обчислень за методом Фур'є:

```
> tt:=1/(Nn+1):p:=1: q:=1:
a:=array(1..Ns):b:=array(1..Ns):
for n from 1 to Ns do
a[n]:=sqrt(2/p)*int(0*sin(Pi*n*x),x=0..1):
b[n]:=sqrt(2/p)*int(1*sin(Pi*n*x),x=0..1):
```

```
od:

u:=array(1..Nn,1..Nn):
for i from 1 to Nn do
for j from 1 to Nn do trew:=sum('sin(Pi*k*i*tt)/sinh(Pi*k*q)*(a[k]*sinh(Pi*k*(1-
j*tt))+b[k]*sinh(Pi*k*j*tt))', 'k=1..Ns'):
u[i,j]:=sqrt(2)*trew:
od:
od:

for n from 1 to Ns do
a[n]:=sqrt(2/p)*int(0*sin(Pi*n*y),y=0..1):
b[n]:=sqrt(2/p)*int(1*sin(Pi*n*y),y=0..1):
od:
u1:=array(1..Nn,1..Nn):
for i from 1 to Nn do
for j from 1 to Nn do trew:=sum('sin(Pi*k*j*tt)/sinh(Pi*k*q)*(a[k]*sinh(Pi*k*(1-
i*tt))+b[k]*sinh(Pi*k*i*tt))', 'k=1..Ns'):
u1[i,j]:=sqrt(2)*trew:
od:od:.
```

Матриця значень, знайдених за методом Фур'є, в точках області, відповідних скінченно-елементного розбиття:

```
> T_anal:=evalf(evalm(u+u1)).
```

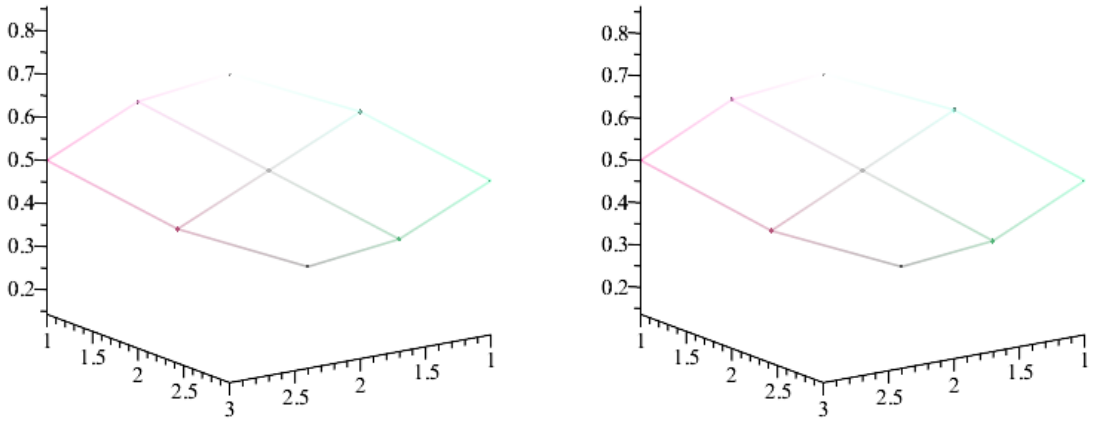
$$T\_anal := \begin{bmatrix} .1359433362 & .2774424497 & .4999999995 \\ .2774424497 & .4999999998 & .7225575495 \\ .4999999995 & .7225575495 & .8640566628 \end{bmatrix}.$$

Опис графічного представлення матриці рішень методом Фур'є:

> FIG[2]:=matrixplot(T_anal,axes=FRAMED,style=PATCHCONTOUR,title=`Мето
д Фурье`,labels=[x,y,T], orientation=[-10,90]).

Розподіл температури за двома методами представлений на рис. 3.17.

> display(FIG);



а б

Рис. 3.17. Розподіл температури за двома методами (а-МСЕ,б-аналітичним методом)

Матриця відносних помилок для знайдених значень неперервної функції у вузлах розбиття (відсотки):

> fd:=evalm(abs(T_num-T_anal)):mer:=matrix(3,3):

for i to 3 do

for j to 3 do

mer[i,j]:=100*fd[i,j]/T_anal[i,j]

od:

od:

evalm(mer);

$$\begin{bmatrix} 5.085800373 & 2.981460122 & .1000000001\ 10^{-6} \\ 2.981460122 & .4000000002\ 10^{-7} & 1.144799498 \\ .1000000001\ 10^{-6} & 1.144799498 & .8001565172 \end{bmatrix}$$

Таким чином, максимальна відносна похибка алгоритму методу кінцевих елементів у порівнянні з аналітичним рішенням за методом Фур'є не перевищила 5.1 %. При подрібненні розрахункової сітки та наближення трикутного скінченного елемента за формою до равностороннему, чисельне рішення, як правило, прагне до точного.

3.5. Розрахунок поля температур в області зі складною геометрією

Процедури автоматичного отримання розрахункової сітки можуть бути засновані на різних геометричних поняттях і принципах, як правило, не маючи при цьому глибокого теоретичного обґрунтування. Головна їх мета - виробити структуру вихідних даних для подальшого розрахунку.

Всі процедури, задіяні в цьому робочому документі, об'єднані в пакет h_FEM і зчитуються з жорсткого диска в наступній командного осередку:

```
> restart:
> read"d:/МКЭ/h_FEM.m":
```

Завантажимо пакети, необхідні в подальшому

```
> with(plots):with(h_FEM):
```

У розглянутому прикладі вихідна область визначення розбивається попередньо (в ручну) на чотири чотирикутних зони (підобласті), що мають спільні сторони, як показано на рис. 3.18.

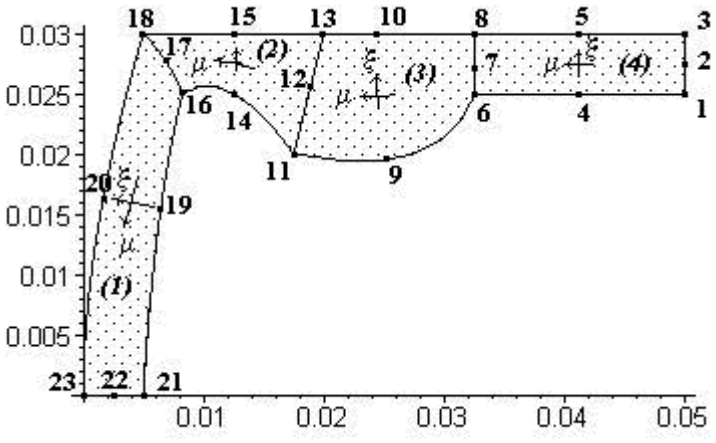


Рис. 3.18. Приклад вихідної області

Нумерація вузлів, що використовуються для завдання чотирикутних зон, також як і самих зон довільна.

Число зон задає змінна INRG, а число граничних точок - INBP.

```
> INRG:=4: INBP:=23:.
```

Координати граничних точок в порядку їх нумерації заносяться в масив XYP:

```
> XYP:=array(1..INBP,1..2,                                    [
[50,25],[50,27.5],[50,30],[41.25,25],[41.25,30],
[32.5,25],[32.5,27.5],[32.5,30],[27.5,20],[26.25,30],
[17.5,20],[18.75,25],[20,30],[12.5,25],[12.5,30],
[8.25,25],[7.,27.5],[5,30],[6.25,14],[1.25,14],
[5,0],[2.5,0],[0,0]
]):.
```

У кожній підобласті, для того щоб була можливість приписувати вузлам, розташованих уздовж спільного кордону одні й ті ж номери, введена місцева система координат ξ μ, орієнтація якої впливає на структуру підсумкової матриці системи лінійних алгебраїчних рівнянь (її ширину смуги).

Чим менша ширина смуги буде отримана, тим ефективніше буде подальший процес обчислень.

Дані про з'єднання зон вводяться в масив JT і складаються з чисел, що представляють собою номери кожної із сторін окремої зони. При цьому сторони нумеруються наступним чином: (- μ) - перша сторона, (ξ) - друга, (μ) - третя, (- ξ) - четверта.

Перший рядок масиву JT (зона 1) показує, що сторона 1 межує із зоною 2.

Другий рядок (зона 2) - сторона 1 межує з зоною 3, а сторона 3 межує з зоною 1.

Третій рядок (зона 3) - сторона 1 межує з зоною 4, а сторона 3 межує із зоною 2.

Четвертий рядок (зона 4) - сторона 3 межує з зоною 3.

```
> JT:=array(1..INRG,1..4,[
[2,0,0,0],
[3,0,1,0],
[4,0,2,0],
[0,0,3,0]]):
```

Дані зон вводяться в масив region. Структура даних в рядку (на прикладі першого рядка масиву):

1 - номер зони, 14 - число рядків вузлів в зоні, 5 - число стовпців вузлів у зоні;

16,17,18,20,23,22,21,19 - номери вузлів зони обходу проти годинникової стрілки,

тобто починаючи з вузла з координатами - μ, - ξ і рухаючись зліва направо від - ξ до + та ξ зверху вниз від + μ к - μ.

```
> region:=array(1..INRG,1..11,[
[1, 14,5, 16,17,18,20,23,22,21,19],
[2, 7, 5, 11,12,13,15,18,17,16,14],
[3, 7, 5, 6, 7, 8, 10,13,12,11,9 ],
[4, 9, 5, 1, 2, 3, 5, 8, 7, 6, 4 ]]):
```

Початкові вихідні дані передаються в процедуру сіткового розбиття grid, яка повертає результати дискретизації:

```
> grid():
`Ширина полосы матрицы системы уравнений`, 7
`Общее число элементов`, 264
`Общее число узлов`, 170
```

Сіткове розбиття можна візуалізувати (рис. 3.19). При цьому певний вибраний елемент можна помістити на сітці, для цього в якості параметра команди v_rem() вводиться номер цього елемента.

```
> v_rem(5):v_gr();
`Номер помеченного элемента`, 5
`Номера узлов помеченного элемента и их координаты:`
```

`Первый узел №`, 8, [2.615384615, 2.186390533]

`Второй узел №`, 9, [1.329696745, 2.105029586]

`Третий узел №`, 3, [2.50, 0.]

`Сторона 1 элемента образована узлами `, 8, `и`, 9

`Сторона 2 элемента образована узлами `, 9, `и`, 3

`Сторона 3 элемента образована узлами`, 3, `и`, 8

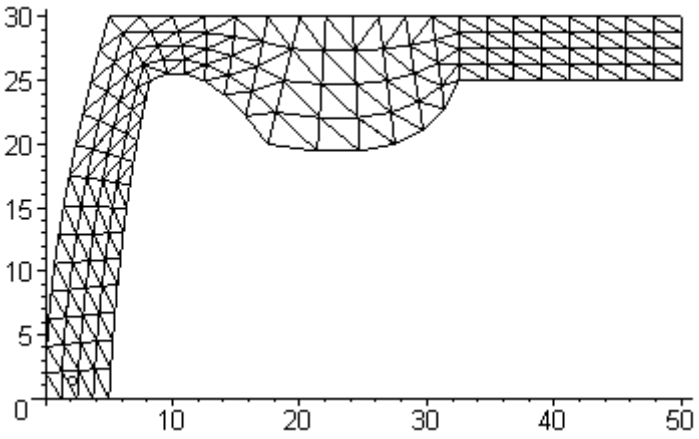


Рис. 3.19. Сітка кінцевих елементів

Далі вихідні дані модифікуються для розрахунку температурного поля, тобто задаються граничні умови і фізичні характеристики області. Та частина кордону, на якій умови теплообміну не будуть задані за замовчуванням теплоізольована. Масив conv заповнюється для елементів, сторони яких беруть участь у процесі теплообміну з навколишнім середовищем. Число таких сторін визначає nr:

> nr:=4;

$nr := 4$.

Структура двовимірного масиву conv наступна: conv [номер елемента, номер сторони].

У конвективному теплообміні може брати участь не більше двох сторін одного елемента.

Якщо теплообміну з середовищем не відбувається, всі відповідні дані залишаються рівними нулю.

> conv:=array(1..nr,1..2,[[2,2],[2,3],[6,3],[8,2]]);

$$conv := \begin{bmatrix} 2 & 2 \\ 2 & 3 \\ 6 & 3 \\ 8 & 2 \end{bmatrix}$$

Змінним H і TINF присвоюються значення відповідно коефіцієнта конвективного теплообміну [Вт/(м^2 К)] і температури [градуси Цельсія] навколишнього середовища в точках поверхні конвективного теплообміну:

> H:=5: TINF:=-5:.

Змінним KXX та KYY присвоюються значення коефіцієнтів теплопровідності вихідного матеріалу відповідно вздовж координатних осей X і Y [Вт/м].

> KXX:=10: KYY:=10:.

Число вузлів обчислювальної сітки, в яких задана потужність теплового потоку (UZq) і відомо значення температури (UZt):

> UZq:=1: UZt:=0:.

Номери вузлів, в яких задана потужність теплового потоку, заносяться у масив Numpy:

> NUMq:=array(1..UZq,[21]);

$$NUMq := [21]$$.

Задане значення потужності у вузлах [Вт] заносяться в масив QQ:

> QQ:=array(1..UZq,[100]);

$$QQ := [100]$$.

Номери вузлів, в яких відоме значення температури заносяться в масив Nums:

> NUMt:=array(1..UZt,[]);

$$NUMt := \text{array}(1 .. 0, [\ ])$$

Значення температури [градуси Цельсія] у відповідних вузлах заносяться в масив TT:

> TT:=array(1..UZt,[]);

$TT := \text{array}(1 .. 0, [\ ])$.

Дані передаються яка обчислює процедуру:

> tdheat():.

Тепер з масиву Xd можна вивести значення температури в і-му вузлі сітки, наприклад:

> Xd[15];

2.950875445

Розподіл температурного поля представлено на рис. 3.20.

> polygonplot3d(list_poy, scaling=UNCONSTRAINED, axes=FRAMED, titlefont=[TIMES,ROMAN,12], shading=Z,labels=[x,y,T], style=hidden, orientation=[-130,50],lightmodel=light3);.

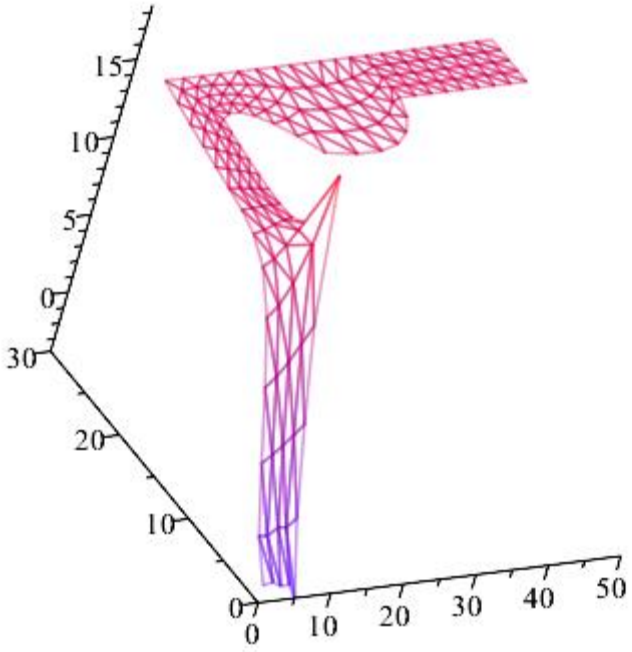


Рис. 3.20. Розподіл температурного поля

3.6. Елементи високого порядку

Застосування елементів високого порядку, як правило, призводить до

досягнення заданої точності рішення при меншій кількості вхідних даних. Проте воно не завжди веде до скорочення повного часу розрахунку, оскільки для складання матриць елемента необхідно використовувати методи чисельного інтегрування, які вимагають виконання великої кількості арифметичних операцій.

Розглянемо загальну форму одновимірного апроксимуючого полінома квадратичним елементом (з трьома вузлами) на рис. 3.21.

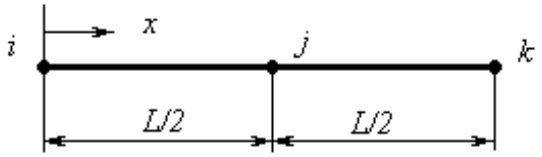


Рис. 3.21. Загальна форма одновимірного апроксимуючого полінома квадратичним елементом

В цьому випадкувідповідає апроксимуючий поліном має вигляд:

$$\phi = \alpha_1 + \alpha_2 x + \alpha_3 x^2$$

Формула кубічного полінома має вигляд (рис. 3.22):

$$\phi = \alpha_1 + \alpha_2 x + \alpha_3 x^2 + \alpha_4 x^3$$

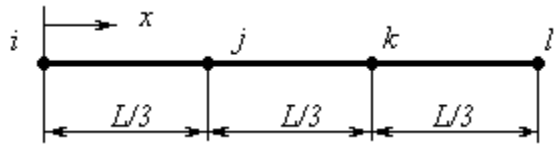


Рис. 3.22. Загальна форма одновимірного апроксимуючого полінома кубічним елементом

Коефіцієнти поліномів можна визначити аналогічно лінійним за допомогою заданих значень шуканої функції у вузлах:

> restart: with(linalg):

> P:=([Phi[1],Phi[2],Phi[3]]):

> C:=matrix([[1,X[1],X[1]^2],

[1,X[2],X[2]^2],

[1,X[3],X[3]^2]

]):

> B:=([1,x,x^2]):.

Норма функції форми прийме вигляд:

> N:=multiply(B,inverse(C));

$$N := \left[\frac{X_2 X_3}{X_2 X_3 - X_2 X_1 + X_1{}^2 - X_3 X_1} - \frac{x\,(X_2 + X_3)}{X_2 X_3 - X_2 X_1 + X_1{}^2 - X_3 X_1} + \frac{x^2}{X_2 X_3 - X_2 X_1 + X_1{}^2 - X_3 X_1}, \right.$$

$$- \frac{X_3 X_1}{-X_3 X_1 + X_2 X_1 - X_2{}^2 + X_2 X_3} + \frac{x\,(X_1 + X_3)}{-X_3 X_1 + X_2 X_1 - X_2{}^2 + X_2 X_3} - \frac{x^2}{-X_3 X_1 + X_2 X_1 - X_2{}^2 + X_2 X_3},$$

$$\left. \frac{X_2 X_1}{-X_3 X_1 + X_2 X_1 + X_3{}^2 - X_2 X_3} - \frac{x\,(X_1 + X_2)}{-X_3 X_1 + X_2 X_1 + X_3{}^2 - X_2 X_3} + \frac{x^2}{-X_3 X_1 + X_2 X_1 + X_3{}^2 - X_2 X_3} \right]$$

Квадратичний інтерполяційний поліном буде дорівнює:

> phi:=multiply(N,P);

$$\phi := \left(\frac{X_2 X_3}{X_2 X_3 - X_2 X_1 + X_1{}^2 - X_3 X_1} - \frac{x\,(X_2 + X_3)}{X_2 X_3 - X_2 X_1 + X_1{}^2 - X_3 X_1} + \frac{x^2}{X_2 X_3 - X_2 X_1 + X_1{}^2 - X_3 X_1} \right) \Phi_1$$

$$+ \left(- \frac{X_3 X_1}{-X_3 X_1 + X_2 X_1 - X_2{}^2 + X_2 X_3} + \frac{x\,(X_1 + X_3)}{-X_3 X_1 + X_2 X_1 - X_2{}^2 + X_2 X_3} - \frac{x^2}{-X_3 X_1 + X_2 X_1 - X_2{}^2 + X_2 X_3} \right) \Phi_2$$

$$+ \left(\frac{X_2 X_1}{-X_3 X_1 + X_2 X_1 + X_3{}^2 - X_2 X_3} - \frac{x\,(X_1 + X_2)}{-X_3 X_1 + X_2 X_1 + X_3{}^2 - X_2 X_3} + \frac{x^2}{-X_3 X_1 + X_2 X_1 + X_3{}^2 - X_2 X_3} \right) \Phi_3 .$$

Нехай, наприклад, дано:

координати трьох вузлів

> X:=[0,1,2]:

і значення функції в них:

> Phi:=[40,25.,20]:

Тоді шуканий інтерполяційний поліном:

> phi;

$$40. - 20.\, x + 5.\, x^2$$

Зобразимо його на графіку (рис. 3.23).

> plot(phi,x=0..2);

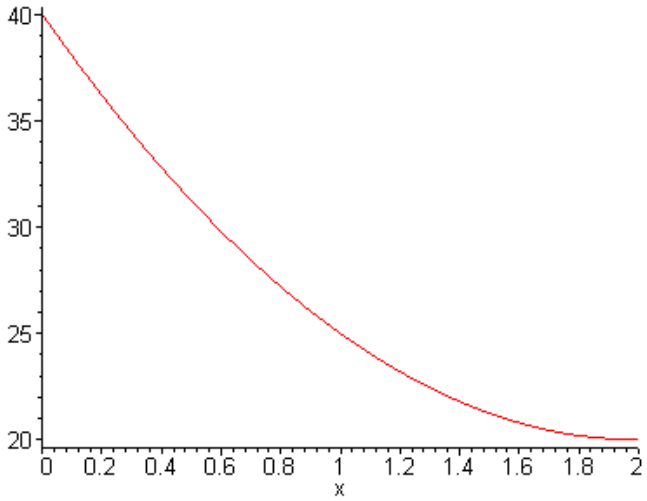


Рис. 3.23. Шуканий інтерполяційний поліном

В точці з координатою x = 0.5 маємо значення шуканої функції:

> subs(x=0.5,phi);

31.25.

Показаний спосіб отримання інтерполяційних поліномів неважко поширити на інші елементи різних типів і розмірностей.

Використання при вирішенні завдань елементів високого порядку аналогічно використанню симплекс-елементів, оскільки вибір інтерполяційного полінома не пов'язаний з вихідними диференціальними рівняннями.

3.7. Опис роботи розробленого програмного додатку

Вхідними даними є змінні, які вводяться користувачем. А саме крок, та такі значення: T – температура; k - коефіцієнт теплопровідності; q - тепловий потік

заданої інтенсивності; Q - внутрішне теплове джерело або стік; *a* - коефіцієнт конвективного теплообміну; T_w - температура навколишнього середовища; *l* - направляючі косинуси вектора нормалі до поверхні; *V* – обсяг; *S* - площа поверхні; *x,y* - координати.

Вихідними даними для програми є текст, математична формула, графік, матриця, масиви даних, рисунок.

Запуск програми здійснюється одним з наведених нижче способів:

1. Вибрати файл проекту з розширенням .mws та запустити його.

2. Відкрити середовище Maple, вибрати пункт меню File→Open або натиснути комбінацію клавіш Ctrl+O та в діалоговому вікні вибрати файл з розширенням .mws. Після відкриття файлу необхідно натиснути на піктограму запуску «!!!» або «!» .

Запуск програми виконується через систему комп'ютерної математики Maple, здійснюється стандартним для Windows способом. Для запуску програми потрібно запустити ярлик на робочому столі «Maple» після інсталяції, та відкрити файл проекту за допомогою «File – Open» або ж запустити файл з розширенням .mws у папці з проектом. Після запуску головне вікно головної програми має вигляд, який приведено на рис. 3.24-3.25.

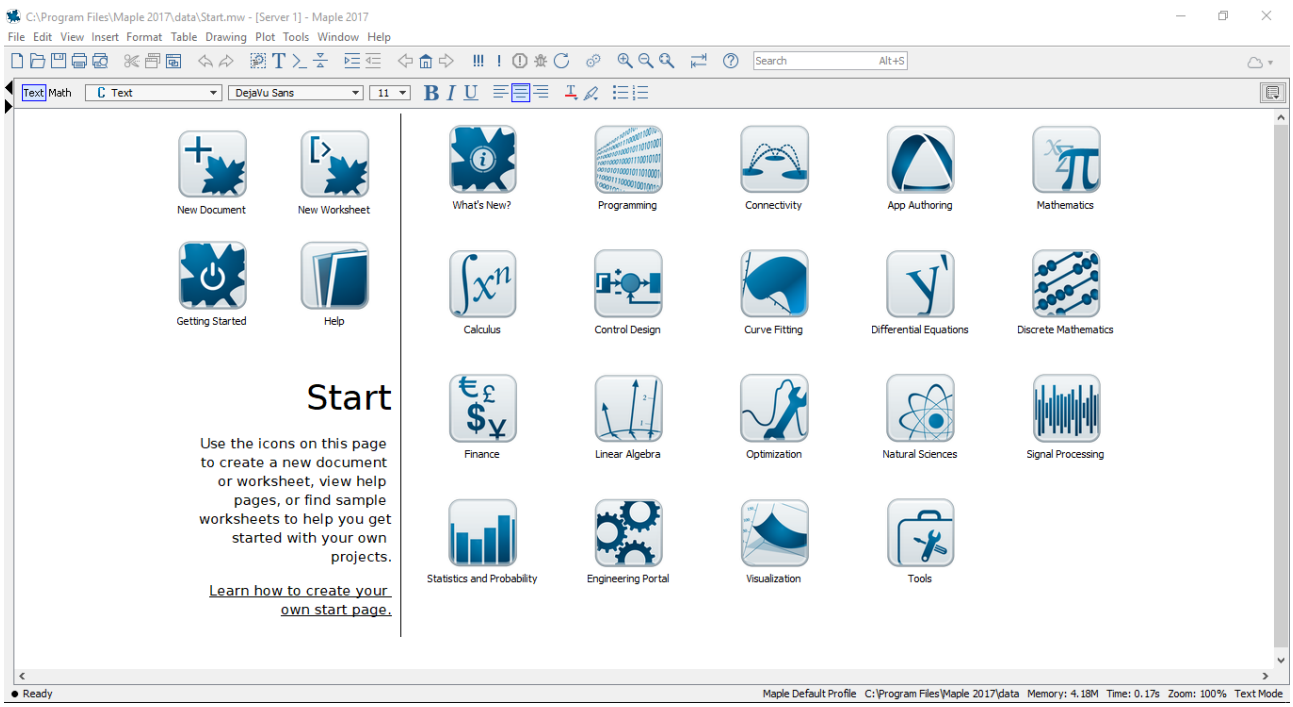


Рис. 3.24. Головне вікно програми

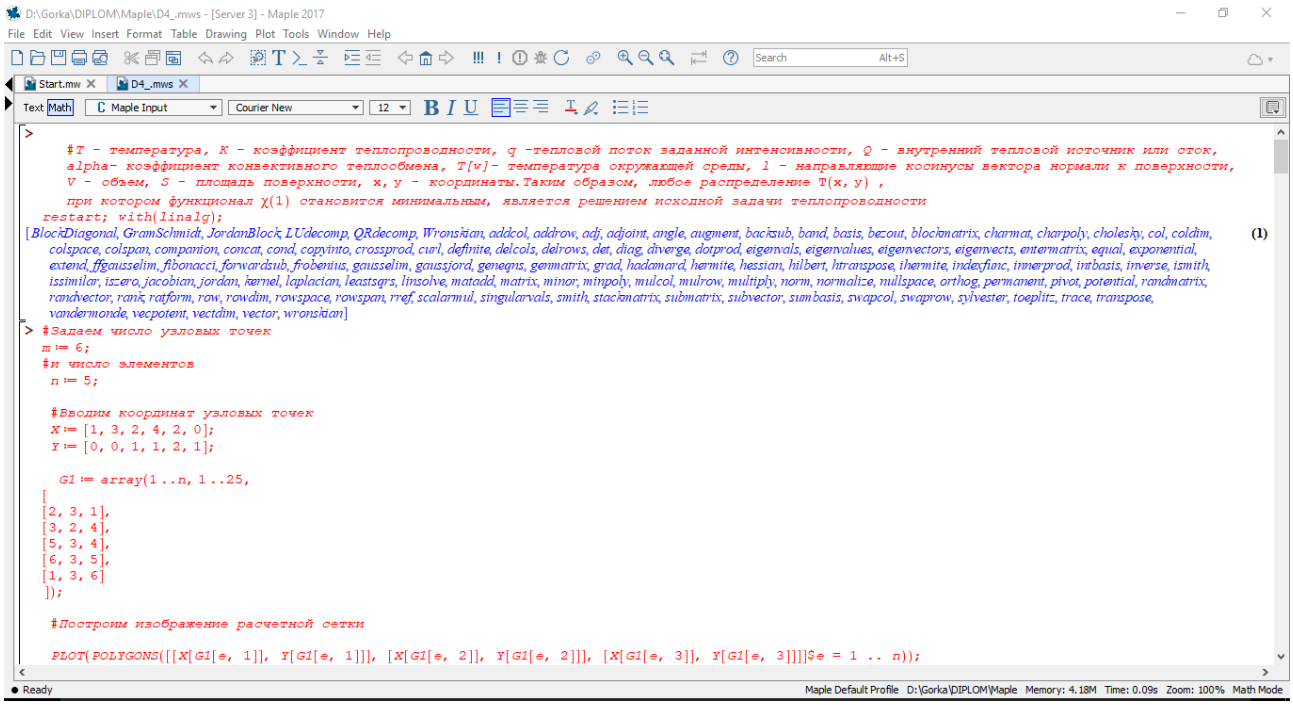


Рис. 3.25. Вікно програми після відкриття файлу проекту

Для того щоб запустити програму потрібно натиснути на піктограму «!!!», щоб запустити всю програму. Або ж піктограму «!», щоб запустити частину коду програми, на якій знаходиться курсор (рис. 3.26).

Після запуску програма прийме вид, як зображено на рис. 3.27. Червоним кольором та вирівнювання зліва – текст програми. Синім кольором та вирівнювання по центру – результат програми. Графіки, рисунки також є результатом праці програми.

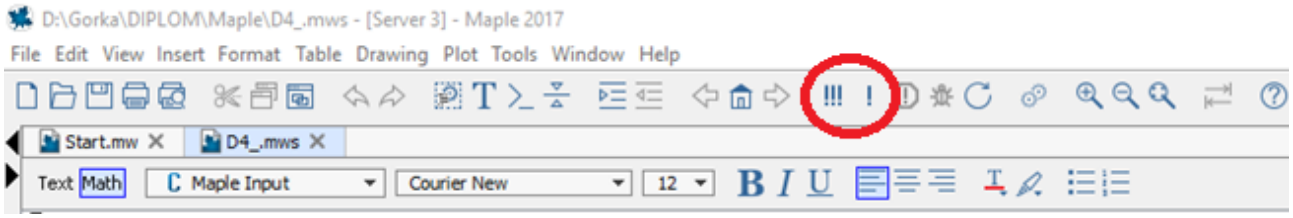


Рис. 3.26. Піктограми запуску програми

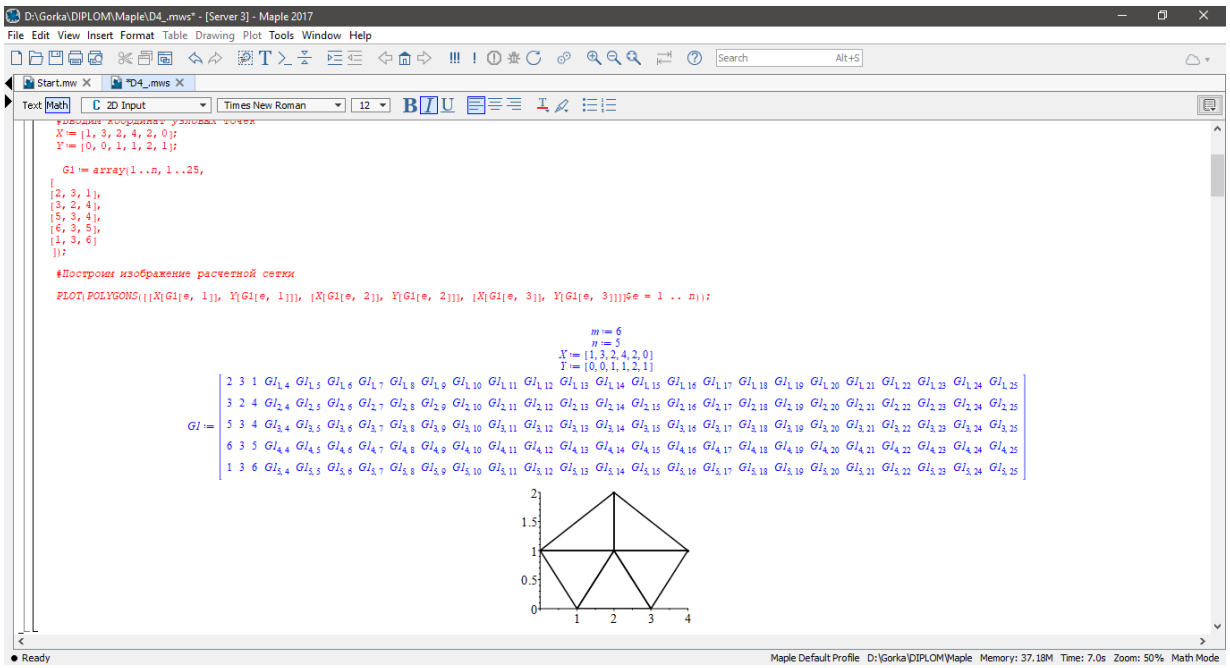


Рис. 3.27. Результат виконання програми

Аналогічні дії виконуються для інших файлів проекту.

Щоб змінити дані на нові, потрібно замінити вхідні дані на бажані користувачем (рис. 3.28). Якщо ці дані не будуть підходити то система Maple виведе в результаті про це повідомлення та подальші розрахунки будуть призупинені. Приклад невірних даних та повідомлення про це зображені на рис. 3.29.

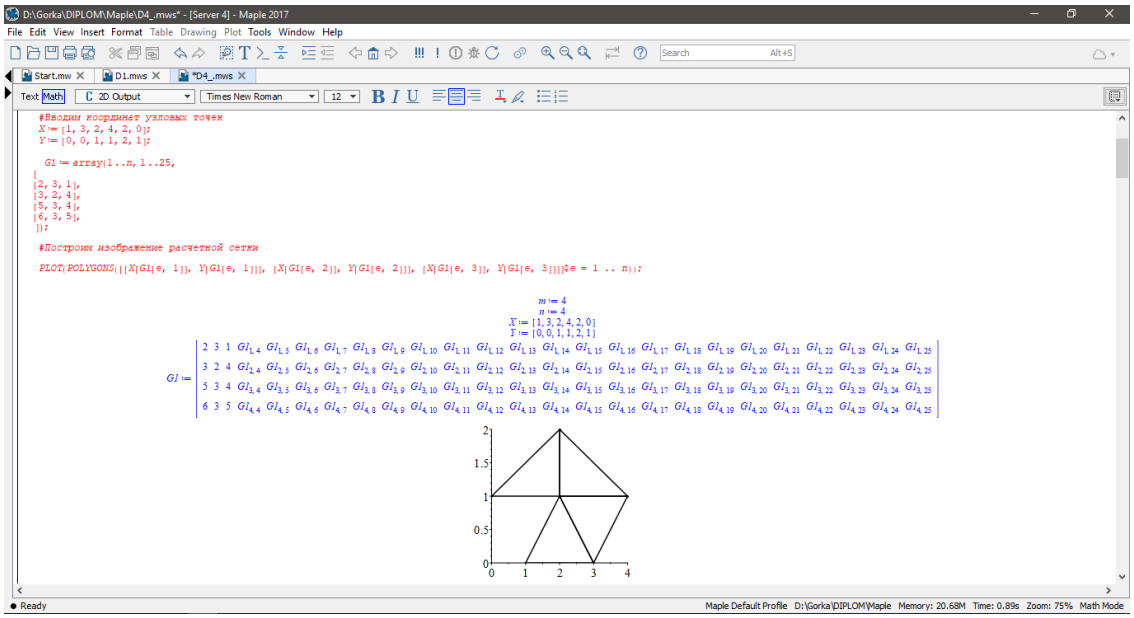


Рис. 3.28. Результат роботи програми зі зміненними даними

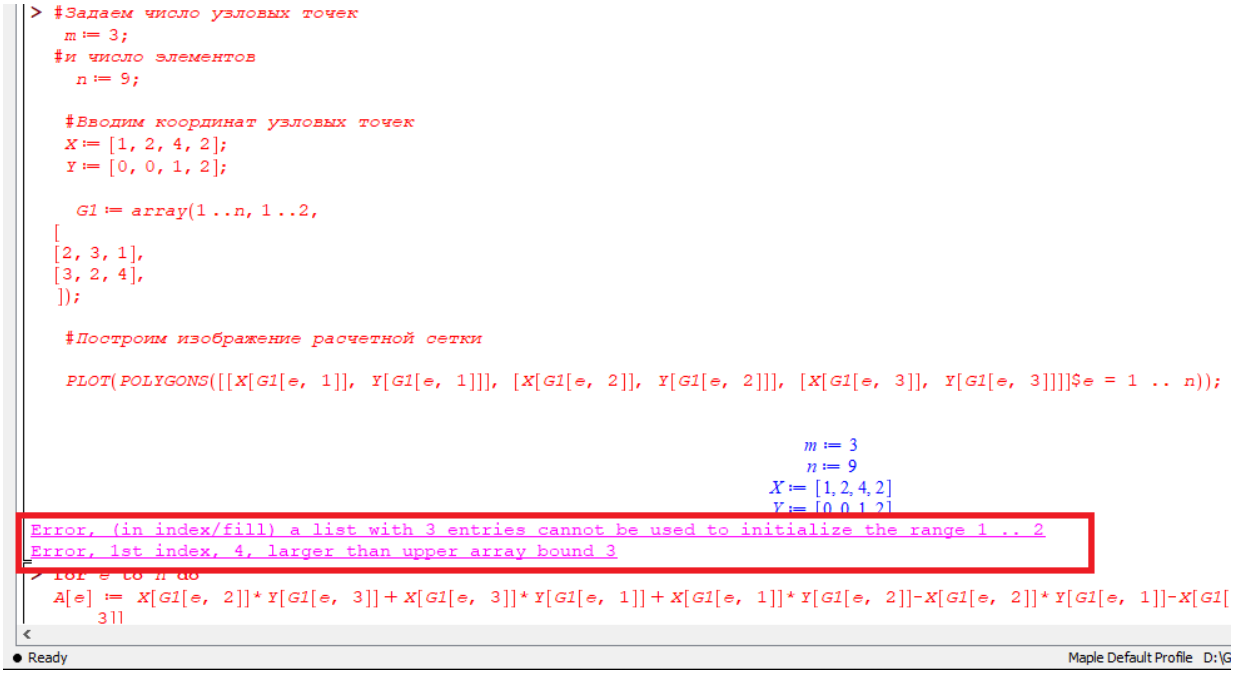


Рис. 3.29. Повідомлення про помилку

Для того, щоб зберегти файл проекту потрібно натиснути у випадаючому меню «File - Save» (рис. 3.30). Для виходу з програми потрібно натиснути у випадаючому меню «File - Exit» (рис. 3.30).

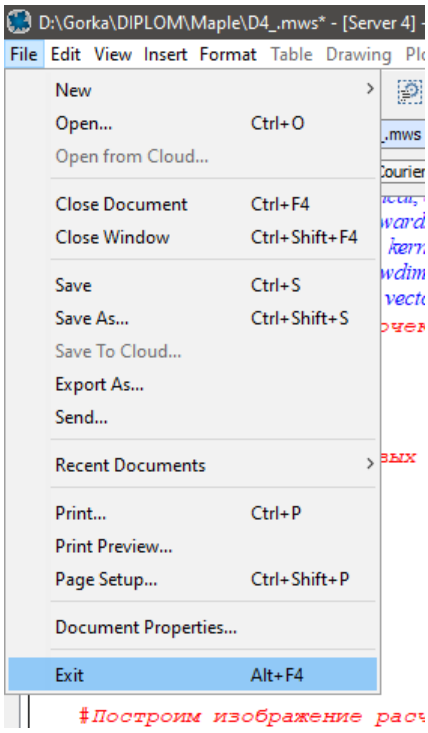


Рис. 3.30. Випадаюче меню

3.8 Висновок

Порівнявши результати, отримані аналітичним шляхом,та результати для МСЕ, можна зробити висновок, що результати отримані за допомогою програми, збігаються з результатами отриманими методом Фур’є, що робить впровадження даного програмного забезпечення для знаходження температурних полів в двовимірних тілах ефективним. Також точність отриманих використовуючи ПЗ результатів, залежить від щільності дискретизації сітки, більш щільна сітка дає точніші результати.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було розроблене програмне забезпечення, яке працює в середовищі системи комп’ютерної математики Maple, використовуючи операційну систему Windows, для розрахунку розподілу температурних полів у двовимірних тілах з використанням метода МСЕ, що дозволяє отримати найбільш точний результат розрахунку. За допомогою програмного забезпечення було проведення порівняння його з аналітичним розрахунком розподілу температурних полів. Максимальна відносна похибка алгоритму методу кінцевих елементів у порівнянні з аналітичним рішенням за методом Фур'є не перевищила 5.1 %.

Були розраховані розподіли температурних полів для області зі складною геометрією. Алгоритм реалізує метод скінченних елементів и рівняння теплопровідності для забезпечення результату.

Програмне забезпечення може використовуватись при моделюванні і проектуванні механізмів, що працюють в умовах інтенсивного нагріву, що допоможе при їх розробці, так як аналіз розподілу температур дозволяє створити більш ефективну і безпечну форму. Для досягнення поставленої мети були виконані задачі:

– досліджені фізичні закони розподілу температури у температурних полях;

– досліджені методи для розрахунку розподілу температурних полів;

– проаналізовані метод скінченних елементів і рівняння теплопровідності для реалізації їх в алгоритмі;

– створено програмне забезпечення для розрахунку розподілу температур.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Юшко С. В. Стаціонарна теплопровідність. Підручник / С. В. Юшко, О. Є. Борщ, М. А. Юшко. – Харків: НТУ «ХПІ», 2011. – 80 с.
2. Єгоров Я.О. Теоретичні основи теплотехніки у системах машинобудування /Я.О.Єгоров, С.Б.Бєліков, О.М.Улітенко. – Запоріжжя:Дике Поле, 2004. – 286 с.
3. Слинько Г.І. Теплотехнічні процеси та теплова обробка матеріалів і виробів / Г.І.Слинько, С.Б.Бєліков, О.М.Улітенко. – Мелітополь: ООО «Издательский дом Мелитопольской городской типографии», 2011. – 258 с.
4. Константінов С.М. Теплообмін: Підручник. - К.: ВПІ ВПК "Політехніка": Інрес, 2005. -304 с.
5. Капинос В.М., Кошельник В.М., Навроцкий В.В. Процессы теплообмена в примерах и задачах / В.М.Капинос, В.М.Кошельник, В.В. Навроцкий В.В. - Харьков: НТУ "ХПИ", 2007. - 192 с.
6. Ткаченко О.О. Високотемпературні процеси та установки / О.О. Ткаченко – Київ: А. С. К. 2005. – 480 с.
7. Махней О. В. Математичне забезпечення автоматизації прикладних досліджень / О. В. Махней,Т. П. Гой. – Івано-Франківськ : Сімик, 2013. – 304 с.
8. Основи роботи в пакеті Maple. Частина 2. Інструментальні засоби математичного моделювання /О.М.Гірник, М.В.Лучко – Львів : Видавництво Львівської комерційної академії, 2006. – 39 с.
9. Карвацький А.Я. Метод скінченних елементів у задачах механіки суцільних середовищ.– К.: НТУУ «КПІ», 2015. – 391 с.
10. Chen, Z. X. Finite element methods and their applications / Z. X. Chen. – Berlin : Springer, 2005. – 424 p.
11. Solin, P. Partial differential equations and the finite element method / P. Solin. – Hoboken : Wiley-Interscience, 2006. – 504 p.
12. Thomee, V. Galerkin finite element methods for parabolic problems / V. Thomee. – Berlin : Springer, 2006. – 459 p.

13. Попов Б. О. Розв'язування математичних задач у системі комп'ютерної алгебри Maple V. — Київ : ViP, 2001. — 312 с. — ISBN 966-7897-03-6.

14. Popov B., Laushnyk O. A package of function approximation. — Waterloo Maple, 2001. — 40 p.

15. Малачівський П. С. Розв’язування задач в середовищі Maple: навчальний посібник / П. С. Малачівський, Я. В. Пізюр. – Львів: Растр-7, 2016. – 282 с.

16. Білоусова Л. І. Курс вищої математики у середовищі Maple / Л. І. Білоусова, М. М. Горонескуль. – Х. : УЦЗУ, КП «Міська друкарня», 2009. – 412.

17. Кобильник Т. П. Системи комп’ютерної математики: Maple, Mathematica, Maxima / Т. П. Кобильник. – Дрогобич : Редакційно-видавничий відділ ДДПУ імені Івана Франка, 2008. – 315 с.

18. Махней О. В. Лабораторний практикум у Maple: методичні рекомендації до проведення лабораторних занять. – Івано-Франківськ: ВДВ ЦІТ ПНУ, 2010. – 32 с.

19. Maple 7. Основи практичного застосування / О.М.Гірник, А.В. Костенко, М.В.Лучко, М.І.Плеша – Львів : ВНТЛ-Класика, 2002. – 174 с.

20. Національна бібліотека України імені В. І. Вернадського. [Електронний ресурс]. – Режим доступу: http://www.nbuv.gov.ua/

21. Наукова бібліотека НУ «Запорізька політехніка». [Електронний ресурс]. –Режим доступу: http://library.zntu.edu.ua/

22. Сидоренко С.І. Теорія тепло- та масопереносу у матеріалах [Електронний підручник] / С.І.Сидоренко, С.М.Волошко, С.О.Замулко, Г.Д.Холмська. – К.: КПІ, – Режим доступу: http://kpm.kpi.ua/doc/DemoVersion090914/index.htm

23. Національна бібліотека України імені Ярослава Мудрого (м. Київ, вул. Грушевського, 1) / [Електронний ресурс]. – Режим доступу: http://elib.nplu.org/

24. Науково-технічна бібліотека ім.. Г.І. Денисенка Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» (м. Київ, Проспект Перемоги, 37) / [Електронний ресурс]. – Режим доступу: http://www.library.kpi.ua/, http://ela.kpi.ua/handle/123456789/2145

25. Державна науково-технічна бібліотека України (м. Київ, вул. Антоновича, 180) / [Електронний ресурс]. – Режим доступу: http://gnbt.gov.ua/

26. Національна парламентська бібліотека України (м. Київ, вул. М. Грушевського, 1) / [Електронний ресурс]. – Режим доступу: http://nplu.org/

27. Наукова бібліотека Кабінету Міністрів України (м. Київ, вул. Грушевського. 12/2) / [Електронний ресурс]. – Режим доступу: http://kmu.gov.ua/

ДОДАТОК А

ЛІСТИНГ ПРОГРАМИ

```
// 1
restart:with(linalg );
P:=[Phi[1],Phi[2],Phi[3],Phi[4]];
C:=matrix(4,4,[
[1,X[1],Y[1],Z[1]],
[1,X[2],Y[2],Z[2]],
[1,X[3],Y[3],Z[3]],
[1,X[4],Y[4],Z[4]]
]);
B:=[1,x,y,z];
N:=multiply(B,inverse(C));
phi:=multiply(N,P);
X:=[1,0,2,1]; Y:=[2,0,0,0]; Z:=[1,0,0,3];
with(plots):
polygonplot3d([[X[1],Y[1],Z[1]],
[X[2],Y[2],Z[2]],
[X[3],Y[3],Z[3]],
[X[4],Y[4],Z[4]],
[X[2],Y[2],Z[2]],
[X[1],Y[1],Z[1]]],
axes=boxed,orientation =[-160,-100],
shading=XY,style=PATCHCONTOUR,light=[20,10,5,3,1],lightmodel='light3');
Phi:=[40,34.,26,15];
phi;
subs(x=1,y=0.5,z=1,phi);
P := [Phi[1], Phi[2], Phi[3]];
C:=matrix([
[1,X[1],Y[1]],
[1,X[2],Y[2]],

[1,X[3],Y[3]]
]);
B:=([1,x,y]);
N:=multiply(B,inverse(C));
phi:=multiply(N,P);
X:=[1,0,2];Y:=[2,0,0];
Phi:=[40,34.,26];
phi;
with(plots):
polygonplot3d([
[X[1],Y[1],Phi[1]],
[X[2],Y[2],Phi[2]],
[X[3],Y[3],Phi[3]]],axes=boxed,style=PATCHCONTOUR);
subs(x=1,y=0.5,phi);
restart:with(linalg);
P := [Phi[1], Phi[2]];
C:=matrix([
[1,X[1]],
[1,X[2]]
```

```
]);
B:=([1,x]);
N:=multiply(B,inverse(C));
phi:=multiply(N,P);
X:=[1,0];
Phi:=[40,34.];
phi;
plot(phi,x=0..1);
subs(x=0.5,phi);

//2
restart; with(linalg);
X := [1, 3, 2, 4, 2, 0]; Y := [0, 0, 1, 1, 2, 1]; Phi := [2.5, 3, 3, 2, 3.5, 3];
G1 := array([[X[1], Y[1], X[2], Y[2], X[3], Y[3], Phi[1], Phi[2], Phi[3]], [X[3], Y[3], X[2], Y[2],
X[4], Y[4], Phi[3], Phi[2], Phi[4]], [X[5], Y[5], X[3], Y[3], X[4], Y[4], Phi[5], Phi[3], Phi[4]],
[X[6], Y[6], X[3], Y[3], X[5], Y[5], Phi[6], Phi[3], Phi[5]], [X[1], Y[1], X[3], Y[3], X[6], Y[6],
Phi[1], Phi[3], Phi[6]]]);
B := vector([1, x, y]); phi := vector(5);
for e to 5 do C := matrix([[1, G1[e, 1], G1[e, 2]], [1, G1[e, 3], G1[e, 4]], [1, G1[e, 5], G1[e,
6]]]); N := multiply(B, inverse(C)); P := [G1[e, 7], G1[e, 8], G1[e, 9]]; phi[e] := multiply(N, P)
end do;
evalm(phi);
with(plots); list_point := [seq([[G1[S, 1], G1[S, 2], G1[S, 7]], [G1[S, 3], G1[S, 4], G1[S, 8]],
[G1[S, 5], G1[S, 6], G1[S, 9]]], S = 1 .. 5)]; polygonplot3d(list_point, scaling = UNCONSTRAINED,
axes = FRAMED, titlefont = [TIMES, ROMAN, 12], shading = Z, labels = [x, y, phi], style = hidden,
orientation = [-50, 40], lightmodel = light3);




//3
restart;
with(linalg);
m := 6; Phi := vector(m);
X := [0, 1, 2, 3, 4, 5];
A := Pi;
Phi[1] := 100; Phi[6] := 0;
for i to m do q[i] := 0; alpha[i] := 1; Tw[i] := 0 end do;
n := m-1; G1 := array(1 .. n, 1 .. 11);
for i to n do K[i] := 1; G1[i, 1] := X[i]; G1[i, 2] := X[i+1]; G1[i, 3] := Phi[i]; G1[i, 4] :=
Phi[i+1]; G1[i, 5] := q[i]; G1[i, 6] := q[i+1]; G1[i, 7] := alpha[i]; G1[i, 8] := alpha[i+1]; G1[i,
9] := Tw[i]; G1[i, 10] := Tw[i+1]; G1[i, 11] := K[i] end do; print*(`÷èñëî ýëåìåíòîâ -`, n);
evalm(G1);
B := vector([1, x]); eqns := NULL; ch := 0;
for e to n do C := matrix([[1, G1[e, 1]], [1, G1[e, 2]]]); N := multiply(B, inverse(C)); P := [G1[e,
3], G1[e, 4]]; phi := multiply(N, P); chi := int((1/2)*K[e]*(diff(phi, x))^2*A, x = G1[e, 1] ..
G1[e, 2]); if q[e] <> 0 or q[e+1] <> 0 then chi := chi+sum('int(q[k]*Phi[k], s = 0 .. A)', 'k' = e
.. e+1) end if; if Tw[e] <> 0 or Tw[e+1] <> 0 then chi := chi+sum('int((1/2)*alpha[k]*(Phi[k]-
Tw[k])^2, s = 0 .. A)', 'k' = e .. e+1) end if; ch := ch+chi end do; ch;
var1 := NULL; for e to m do if type(Phi[e], integer) then eq1 := 0 = 0 else eq1 := diff(ch, Phi[e])
= 0; var1 := var1, Phi[e] end if; eqns := eqns, eq1 end do;
eqns := {eqns}; var := {var1};
sols := solve(eqns, var);
```

```
sorty := proc (Phi, sols, m) local i, j; for i to nops(sols) do for j to m do if Phi[j] <> op(1,
sols[i]) then next else Phi[j] := op(2, sols[i]) end if end do end do end proc; sorty(Phi, sols, m);
evalm(Phi); evalm(G1);
with(plots); f_1 := plot([seq([X[i], Phi[i]], i = 1 .. nops(X))], style = point, symbol = circle,
color = red); display(f_1);
with(DEtools); du1 := diff(T_0(x), x, x) = 0; in1 := T_0(0) = 100, T_0(5) = 0;
qq := dsolve({du1, in1}, {T_0(x)}); T_a := subs(qq, T_0(x));
f_2 := plot(T_a, x = 0 .. 5, title = Ïðîâåðêà);
display({f_1, f_2});

//4
restart; with(linalg);
m := 6; n := 5; X := [1, 3, 2, 4, 2, 0]; Y := [0, 0, 1, 1, 2, 1]; G1 := array(1 .. n, 1 .. 25, [[2,
3, 1], [3, 2, 4], [5, 3, 4], [6, 3, 5], [1, 3, 6]]); PLOT(POLYGONS([[X[G1[e, 1]], Y[G1[e, 1]]],
[X[G1[e, 2]], Y[G1[e, 2]]], [X[G1[e, 3]], Y[G1[e, 3]]]]$e = 1 .. n));
for e to n do A[e] := X[G1[e, 1]]*Y[G1[e, 2]]-X[G1[e, 1]]*Y[G1[e, 3]]-X[G1[e, 2]]*Y[G1[e,
1]]+X[G1[e, 2]]*Y[G1[e, 3]]+X[G1[e, 3]]*Y[G1[e, 1]]-X[G1[e, 3]]*Y[G1[e, 2]] end do;
Phi := vector(m); for i to m do q[i] := 0; Q[i] := 0; alpha[i] := 1; Tw[i] := 0 end do;
Phi[1] := 100.; Phi[4] := 0.;
Q[3] := -10;
for i to n do K[i] := 1; G1[i, 4] := X[G1[i, 1]]; G1[i, 5] := Y[G1[i, 1]]; G1[i, 6] := X[G1[i, 2]];
G1[i, 7] := Y[G1[i, 2]]; G1[i, 8] := X[G1[i, 3]]; G1[i, 9] := Y[G1[i, 3]]; G1[i, 10] := Phi[G1[i,
1]]; G1[i, 11] := Phi[G1[i, 2]]; G1[i, 12] := Phi[G1[i, 3]]; G1[i, 13] := q[G1[i, 1]]; G1[i, 14] :=
q[G1[i, 2]]; G1[i, 15] := q[G1[i, 3]]; G1[i, 16] := alpha[G1[i, 1]]; G1[i, 17] := alpha[G1[i, 2]];
G1[i, 18] := alpha[G1[i, 3]]; G1[i, 19] := Tw[G1[i, 1]]; G1[i, 20] := Tw[G1[i, 2]]; G1[i, 21] :=
Tw[G1[i, 3]]; G1[i, 22] := K[i]; G1[i, 23] := Q[G1[i, 1]]; G1[i, 24] := Q[G1[i, 2]]; G1[i, 25] :=
Q[G1[i, 3]] end do; evalm(G1);
B := vector([1, x, y]); eqns := NULL; ch := 0; for e to n do C := matrix([[1, G1[e, 4], G1[e, 5]],
[1, G1[e, 6], G1[e, 7]], [1, G1[e, 8], G1[e, 9]]]); N := multiply(B, inverse(C)); P := [G1[e, 10],
G1[e, 11], G1[e, 12]]; phi := multiply(N, P); chi := int((1/2)*K[e]*((diff(phi,
x))^2+K[e]*(diff(phi, y))^2)*A[e], z = 0 .. 1); if G1[e, 23] <> 0 or G1[e, 24] <> 0 or G1[e, 25] <>
0 then chi := chi-sum('int((1/2)*G1[e, k]*G1[e, k-13], z = 0 .. 1)', 'k' = 23 .. 25) end if; if
G1[e, 13] <> 0 or G1[e, 14] <> 0 or G1[e, 15] <> 0 then chi := chi+sum('int(G1[e, k]*G1[e, k-3], s =
0 .. A[e])', 'k' = 13 .. 15) end if; if G1[e, 19] <> 0 or G1[e, 20] <> 0 or G1[e, 21] <> 0 then chi
:= chi+sum('int((1/2)*G1[e, k-3]*(G1[e, k-9]-G1[e, k])^2, s = 0 .. A[e])', 'k' = 19 .. 21) end if;
ch := ch+chi end do; ch;
var1 := NULL; for i to m do if type(Phi[i], integer) or type(Phi[i], float) then eql := 0 = 0 else
eql := d*ch/dPhi[i] = 0; var1 := var1, Phi[i] end if; eqns := eqns, eql end do;
eqns := {eqns}; var := {var1};
sorty := proc (Phi, sols, m) local i, j; for i to nops(sols) do for j to m do if Phi*{j} <> op(1,
sols[i]) then next else Phi[j] := op(2, sols[i]) end if end do end do end proc; sorty(Phi, sols, m);
evalm(Phi); evalm(G1);
op(eqns);
with(plots); list_poy := [seq([[G1[S, 4], G1[S, 5], G1[S, 10]], [G1[S, 6], G1[S, 7], G1[S, 11]],
[G1[S, 8], G1[S, 9], G1[S, 12]]], S = 1 .. n)]; polygonplot3d(list_poy, scaling = UNCONSTRAINED,
axes = FRAMED, titlefont = [TIMES, ROMAN, 12], shading = ZGREYSCALE, labels = [x, y, T], style =
PATCHCONTOUR, orientation = [-20, 70]);

//5
restart; with(linalg); with(plots); FIG := array(1 .. 2);
m := 25; n := 32;
```

```
X := [0, .25, .5, .75, 1, 0, .25, .5, .75, 1, 0, .25, .5, .75, 1, 0, .25, .5, .75, 1, 0, .25, .5,
.75, 1]; Y := [0, 0, 0, 0, 0, .25, .25, .25, .25, .25, .5, .5, .5, .5, .5, .75, .75, .75, .75, .75,
1, 1, 1, 1, 1];
G1 := array(1 .. n, 1 .. 25, [[1, 2, 6], [2, 7, 6], [2, 3, 7], [3, 8, 7], [3, 4, 8], [4, 9, 8], [4,
5, 9], [5, 10, 9], [6, 7, 11], [7, 12, 11], [7, 8, 12], [8, 13, 12], [8, 9, 13], [9, 14, 13], [9,
10, 14], [10, 15, 14], [11, 12, 16], [12, 17, 16], [12, 13, 17], [13, 18, 17], [13, 14, 18], [14,
19, 18], [14, 15, 19], [15, 20, 19], [16, 17, 21], [17, 22, 21], [17, 18, 22], [18, 23, 22], [18,
19, 23], [19, 24, 23], [19, 20, 24], [20, 25, 24]]);
PLOT(POLYGONS([[X[G1[e, 1]], Y[G1[e, 1]]], [X[G1[e, 2]], Y[G1[e, 2]]], [X[G1[e, 3]], Y[G1[e, 3]]]]$e
= 1 .. n));
Phi := vector(m); for i to m do q[i] := 0; Q[i] := 0; alpha[i] := 1; Tw[i] := 0 end do;
Phi[1] := 0.; Phi[2] := 0.; Phi[3] := 0; Phi[4] := 0; Phi[5] := .5; Phi[6] := 0.; Phi[15] := 1.;
Phi[11] := 0.; Phi[20] := 1.; Phi[16] := 0.; Phi[10] := 1.; Phi[21] := .5; Phi[22] := 1; Phi[23] :=
1; Phi[24] := 1; Phi[25] := 1.;
for e to n do A[e] := X[G1[e, 1]]*Y[G1[e, 2]]-X[G1[e, 1]]*Y[G1[e, 3]]-X[G1[e, 2]]*Y[G1[e,
1]]+X[G1[e, 2]]*Y[G1[e, 3]]+X[G1[e, 3]]*Y[G1[e, 1]]-X[G1[e, 3]]*Y[G1[e, 2]] end do;
for i to n do K[i] := 1; G1[i, 4] := X[G1[i, 1]]; G1[i, 5] := Y[G1[i, 1]]; G1[i, 6] := X[G1[i, 2]];
G1[i, 7] := Y[G1[i, 2]]; G1[i, 8] := X[G1[i, 3]]; G1[i, 9] := Y[G1[i, 3]]; G1[i, 10] := Phi[G1[i,
1]]; G1[i, 11] := Phi[G1[i, 2]]; G1[i, 12] := Phi[G1[i, 3]]; G1[i, 13] := q[G1[i, 1]]; G1[i, 14] :=
q[G1[i, 2]]; G1[i, 15] := q[G1[i, 3]]; G1[i, 16] := alpha[G1[i, 1]]; G1[i, 17] := alpha[G1[i, 2]];
G1[i, 18] := alpha[G1[i, 3]]; G1[i, 19] := Tw[G1[i, 1]]; G1[i, 20] := Tw[G1[i, 2]]; G1[i, 21] :=
Tw[G1[i, 3]]; G1[i, 22] := K[i]; G1[i, 23] := Q[G1[i, 1]]; G1[i, 24] := Q[G1[i, 2]]; G1[i, 25] :=
Q[G1[i, 3]] end do;
evalm(G1);
B := vector([1, x, y]); eqns := NULL; ch := 0; for e to n do C := matrix([[1, G1[e, 4], G1[e, 5]],
[1, G1[e, 6], G1[e, 7]], [1, G1[e, 8], G1[e, 9]]]); N := multiply(B, inverse(C)); P := [G1[e, 10],
G1[e, 11], G1[e, 12]]; phi := multiply(N, P); chi := int((1/2)*K[e]*((diff(phi,
x))^2+K[e]*(diff(phi, y))^2)*A[e], z = 0 .. 1); if G1[e, 23] <> 0 or G1[e, 24] <> 0 or G1[e, 25] <>
0 then chi := chi-A[e]*(sum('int((1/2)*G1[e, k]*G1[e, -13+k], z = 0 .. 1)', 'k' = 23 .. 25)) end if;
if G1[e, 13] <> 0 or G1[e, 14] <> 0 or G1[e, 15] <> 0 then chi := chi+sum('int(G1[e, k]*G1[e, -3+k],
s = 0 .. A[e])', 'k' = 13 .. 15) end if; if G1[e, 19] <> 0 or G1[e, 20] <> 0 or G1[e, 21] <> 0 then
chi := chi+sum('int((1/2)*G1[e, -3+k]*(G1[e, -9+k]-G1[e, k])^2, s = 0 .. A[e])', 'k' = 19 .. 21) end
if; ch := ch+chi end do;
ch; var1 := NULL; for i to m do if type(Phi[i], integer) or type(Phi[i], float) then eq1 := 0 = 0
else eq1 := diff(ch, Phi[i]) = 0; var1 := var1, Phi[i] end if; eqns := eqns, eq1 end do;
eqns := {eqns}; var := {var1}; sols := solve(eqns, var);
sorty := proc (Phi, sols, m) local i, j; for i to nops(sols) do for j to m do if Phi[j] <> op(1,
sols[i]) then next else Phi[j] := op(2, sols[i]) end if end do end do end proc; sorty(Phi, sols, m);
evalm(Phi); evalm(G1);
T_num := matrix([[Phi[7], Phi[8], Phi[9]], [Phi[12], Phi[13], Phi[14]], [Phi[17], Phi[18],
Phi[19]]]);
FIG[1] := matrixplot(T_num, axes = FRAMED, style = PATCHCONTOUR, title = `Ìåòîä Êîíå÷íûõ Ýëåìåíòîâ`,
labels = [x, y, T], orientation = [-10, 90]);
Nn := 3; Ns := 40;
tt := 1/(Nn+1); p := 1; q := 1; a := array(1 .. Ns); b := array(1 .. Ns); for n to Ns do a[n] :=
sqrt(2/p)*(int(0*sin(Pi*n*x), x = 0 .. 1)); b[n] := sqrt(2/p)*(int(sin(Pi*n*x), x = 0 .. 1)) end do;
u := array(1 .. Nn, 1 .. Nn); for i to Nn do for j to Nn do trew :=
sum('sin(Pi*k*i*tt)*(a[k]*sinh(Pi*k*(-j*tt+1))+b[k]*sinh(Pi*k*j*tt))/sinh(Pi*k*q)', 'k = 1 .. Ns');
u[i, j] := sqrt(2)*trew end do end do;
for n to Ns do a[n] := sqrt(2/p)*(int(0*sin(Pi*n*y), y = 0 .. 1)); b[n] :=
sqrt(2/p)*(int(sin(Pi*n*y), y = 0 .. 1)) end do; u1 := array(1 .. Nn, 1 .. Nn); for i to Nn do for j
to Nn do trew := sum('sin(Pi*k*j*tt)*(a[k]*sinh(Pi*k*(-i*tt+1))+b[k]*sinh(Pi*k*i*tt))/sinh(Pi*k*q)',
'k = 1 .. Ns'); u1[i, j] := sqrt(2)*trew end do end do;
```

```
T_anal := evalf(evalm(u+u1));
FIG[2] := matrixplot(T_anal, axes = FRAMED, style = PATCHCONTOUR, title = `Ìåòîä Ôóðüå`, labels =
[x, y, T], orientation = [-10, 90]);
display(FIG);
fd := evalm(abs(T_num-T_anal)); mer := matrix(3, 3); for i to 3 do for j to 3 do mer[i, j] :=
100*fd[i, j]/T_anal[i, j] end do end do; evalm(mer);

//6
restart;
read "h_FEM.m";
INRG := 4; INBP := 23;
XYP := array(1 .. INBP, 1 .. 2, [[50, 25], [50, 27.5], [50, 30], [41.25, 25], [41.25, 30], [32.5,
25], [32.5, 27.5], [32.5, 30], [27.5, 20], [26.25, 30], [17.5, 20], [18.75, 25], [20, 30], [12.5,
25], [12.5, 30], [8.25, 25], [7., 27.5], [5, 30], [6.25, 14], [1.25, 14], [5, 0], [2.5, 0], [0,
0]]);
JT := array(1 .. INRG, 1 .. 4, [[2, 0, 0, 0], [3, 0, 1, 0], [4, 0, 2, 0], [0, 0, 3, 0]]);
region := array(1 .. INRG, 1 .. 11, [[1, 14, 5, 16, 17, 18, 20, 23, 22, 21, 19], [2, 7, 5, 11, 12,
13, 15, 18, 17, 16, 14], [3, 7, 5, 6, 7, 8, 10, 13, 12, 11, 9], [4, 9, 5, 1, 2, 3, 5, 8, 7, 6, 4]]);
grid();

v_rem(5); v_gr();

nr := 4;
conv := array(1 .. nr, 1 .. 2, [[2, 2], [2, 3], [6, 3], [8, 2]]);
H := 5; TINF := -5;
KXX := 10; KYY := 10;
UZq := 1; UZt := 0;
NUMq := array(1 .. UZq, [21]);
QQ := array(1 .. UZq, [100]);
NUMt := array(1 .. UZt, []);
TT := array(1 .. UZt, []);
tdheat();
Xd[15];
polygonplot3d(list_poy, scaling = UNCONSTRAINED, axes = FRAMED, titlefont = [TIMES, ROMAN, 12],
shading = Z, labels = [x, y, T], style = hidden, orientation = [-130, 50], lightmodel = light3);

//7
restart; with(linalg);
P := [Phi[1], Phi[2], Phi[3]]; C := matrix([[1, X[1], X[1]^2], [1, X[2], X[2]^2], [1, X[3],
X[3]^2]]); B := [1, x, x^2];
N := multiply(B, inverse(C));
phi := multiply(N, P);
X := [0, 1, 2];
Phi := [40, 25., 20];
phi;
plot(phi, x = 0 .. 2);
subs(x = .5, phi);
```

Додаток Б

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«ДНІПРОВСЬКА ПОЛІТЕХНІКА»

Факультет інформаційних технологій
Кафедра програмного забезпечення комп'ютерних систем

ВІДГУК

Наукового керівника Бердник Михайла Генадійовича, д.т.н., доцент, професор каф. ПЗКС
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання, посада, місце роботи)

На кваліфікаційну роботу
студента Титаренко Марії Олександрівні
(прізвище, ім'я, по батькові)

курсу II групи 121м-20-1
спеціальності 121 Інженерія програмного забезпечення
на тему Розробка та дослідження ефективності впровадження програмного програмного забезпечення двовимірної задачі теплопровідності

Актуальність теми обумовлена потребою в оптимізації розрахунків розподілу температури при моделюванні механізмів, які працюють в умовах інтенсивного нагріву.

Мета досліджень – підвищення ефективності обчислення та аналізу температурних полів у двовимірних тілах.

Коротка характеристика розділів роботи У першому розділі проаналізована предметна область і наведені дані про фізичні процеси, що використовуються для вирішення проблеми. У другому розділі проаналізовані методи, які можуть бути використані для вирішення проблеми. Третій розділ описує способи рішення розподілу температури, наведений приклад роботи програмного забезпечення і виконане порівняння отриманого результату роботи програми з аналітичним рішенням.

Практичне значення роботи полягає в вирішенні проблеми математичного

моделювання розрахунку температурних полів в двовимірних тілах, що дозволяє більш ефективно і економно проектувати пристрої, які працюють при інтенсивних температурних умовах.

Зауваження та недоліки _____

Висновки та оцінка Кваліфікаційна робота заслуговує оцінки «відмінно», виконавець заслуговує на присвоєння відповідної кваліфікації.

Науковий керівник Бердник Михайло Генадійович, професор, каф. ПЗКС
(прізвище, ім'я, по батькові, посада, місце роботи)

«_13_»___грудня___2022 р. _____
(підпис)

Додаток В

РЕЦЕНЗІЯ
на кваліфікаційну роботу

студента Титаренко Марії Олександрівні
(прізвище, ім'я, по батькові)
курсу II групи 121м-20-1
кафедри програмного забезпечення комп'ютерних систем
спеціальності 121 Інженерія програмного забезпечення
Тема роботи Розробка та дослідження ефективності впровадження програмного забезпечення двовимірної задачі теплопровідності

Стисла характеристика розділів роботи Перший розділ надає інформацію про предметну область і про теплові процеси. Другий розділ інформує про методи, що розглядаються для використання у програмному забезпеченні. Третій розділ містить інформацію про розрахунки розподілу температурних полів, а також про роботу програмного забезпечення і його порівняння з аналітичним рішенням.

Пропозиції, внесені студентом, рівень їх наукового обґрунтування робота, яка рецензується, написана сучасною науковою мовою, висновки обґрунтовані і випливають із суті зробленої роботи.

Практичне значення роботи полягає в аналізі та використанні методів для розрахунку температурних полів для їх більш ефективного використання у тілах обертання для розробки пристроїв, працюючих в умовах інтенсивного нагріву.

Якість оформлення роботи робота виконана у відповідності до вимог оформлення дипломних робіт і повністю відповідає поставленій задачі.
Недоліки в роботі у роботі не досліджено роботи закордонних вчених з тематики роботи.
Загальний висновок отримані результати є закінченою науково-дослідною роботою і викликають науковий інтерес і демонструють здатність Титаренко Марії Олександрівні до самостійного аналізу і проведення наукової роботи та вміння розробки комп'ютерних програм. Кваліфікаційна робота заслуговує оцінки "відмінно", а Титаренко М. О. – присвоєння відповідної кваліфікації.
(підготовленість студента до самостійної роботи як спеціаліста)

Оцінка магістерської роботи "відмінно"

Рецензент Шедловський І.А., к.т.н., доцент, доцент каф. ІТКІ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання, посада, місце роботи)

«___»_____20__ р. _____
(підпис)

ДОДАТОК Г

ПЕРЕЛІК ДОКУМЕНТІВ НА ОПТИЧНОМУ НОСІЇ

Ім’я файла	**Опис**
Пояснювальні документи	
Диплом_ Титаренко.doc	Пояснювальна записка роботи. Документ Word.
Диплом_ Титаренко.pdf	Пояснювальна записка роботи в форматі PDF
Програма	
Program.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
Презентація_ Титаренко.ppt	Презентація роботи