Л. П. ФЕЛЬДМАН, А. І. ПЕТРЕНКО, О. А. ДМИТРІЄВА

ЧИСЕЛЬНІ МЕТОДИ В ІНФОРМАТИЦІ

Затверджено Міністерством освіти і науки України як підручник для студентів вищих навчальних закладів, які навчаються за напрямами «Комп'ютерні науки», «Комп'ютеризовані системи, автоматика і управління», «Комп'ютерна інженерія», «Прикладна математика»

Серія «ІНФОРМАТИКА»
За загальною редакцією академіка НАН України
М. З. Згуровського

Київ
Видавнича група BHV
2006

ББК 22.161.1я73
Ф39
УДК 517.9(075.8)

Рецензенти: *Д. В. Федасюк*, доктор технічних наук, професор, директор Інституту комп'ютерних наук та інформаційних технологій Національного університету «Львівська політехніка»;
С. С. Забара, доктор технічних наук, професор, декан факультету інформаційних техиологій Відкритого міжнародного університету розвитку людини «Україна»;
О. І. Харламова, доктор фізико-математичних наук, провідний науковий співробітник Інституту прикладної математики і механіки НАН України.

Гриф надано Міністерством освіти і науки України,
лист № 14/18.2-331 від 13.02.2006 р.

Фельдман Л. П., Петренко А. І., Дмитрієва О. А.
Ф39 Чисельні методи в інформатиці. — К.: Видавнича група BHV, 2006. — 480 с.: іл.
ISBN 966-552-155-1

Докладно і послідовно викладено основні методи розв'язання задач лінійної та нелінійної алгебри, наближення функцій, звичайних диференціальних рівнянь, рівнянь із частинними похідними та інтегральних рівнянь, застосовуваних для моделювання та оптимізації об'єктів і систем найрізноманітнішої природи. Поряд із класичними розглядаються сучасні чисельні методи розв'язання погано обумовлених і жорстких систем рівнянь. Значну увагу приділено практичній реалізації методів у середовищі комп'ютерної системи математичних розрахунків Mathematica.

Для студентів, які навчаються за напрямами «Комп'ютерні науки», «Комп'ютерна інженерія», «Комп'ютеризовані системи, автоматика і управління», може бути рекомендований студентам усіх спеціальностей, які вивчають сучасні інформаційні технології в межах дисциплін «Чисельні методи» і «Основи програмування та алгоритмічні мови». Підручник може бути корисний також викладачам зазначених дисциплін.

ББК 22.161.1я73

Серія підручників «Інформатика» виходить у світ за підтримки видавництв «Издательский дом "Питер"», м. Санкт-Петербург, та «Освітня книга», м. Київ.

ISBN 966-552-155-1

Стислий зміст

Зміст

Обчислення — не просто засіб отримання числових результатів,
це також знаряддя розуму для дослідження всесвіту.

Мудрість нашого часу

Передмова

Сама назва підручника «Чисельні методи в інформатиці» вказує на те, що з одного боку він призначений для підготовки фахівців у галузі комп'ютерних технологій, а з іншого — спрямований на прикладні аспекти застосування обчислювальних процедур в інженерній практиці. Цим він відрізняється від широко відомих підручників та учбових посібників для студентів математичних спеціальностей, написаних знаними професорами Московського, Санкт-Петербурзького, Новосибірського та інших державних університетів і виданих переважно в 70-х і 80-х роках минулого століття.

Математика, і зокрема обчислювальна математика, як наука має свою методологію та інструментарій, свою термінологію і мову запису теорем і лем, доказів і припущень, абстракцій і закономірностей. Вони ніби утворюють «внутрішній світ» науки, який детально вивчають майбутні математики.

Підготовка інженерів, і в тому числі фахівців у галузі комп'ютерних технологій, базується, навпаки, на освоєнні «зовнішніх» прикладних застосувань математики в інженерній практиці, на освоєнні методики і засобів розв'язання обчислювальних задач певної предметної галузі переважно за допомогою комп'ютерів. Тож майбутнім спеціалістам не слід без потреби занурюватися у «внутрішній світ» математики. Математичні дисципліни для майбутніх математиків і майбутніх інженерів мають викладатися за різними програмами, різними методиками та за допомогою різних підручників — хоча б тому, що неможливо виховати із студента кваліфікованого математика і водночас творчого інженера (хоча, звичайно, є і винятки).

Отже матеріал, поданий у підручнику, і методика його викладу відповідають певним вимогам до підготовки інженерів.

1. У підручнику надані рекомендації з використання окремих чисельних процедур в інженерній практиці і наведено багато типових прикладів їх застосування.
2. Підручник містить багато вправ і задач, призначених для закріплення навичок розв'язання обчислювальних задач як за допомогою калькулятора, так і автоматизованим способом, за допомогою одного з поширених математичних пакетів індивідуального користування (Maple, Matlab, Mathcad і Mathematica), освоєння якого передбачене як самостійна робота студентів.
3. У підручнику розглянуті відсутні в наявній учбовій літературі нові чисельні методи, призначені для розв'язання реальних погано обумовлених і жорстких інженерних задач, а також обчислювальних задач, що мають дуже велику розмірність.

У результаті вивчення даного курсу майбутні фахівці в галузі комп'ютерних технологій повинні:

- засвоїти основні особливості чисельних методів, умови їх правильного використання, можливості адаптації до конкретних інженерних задач;
- набути вміння вибирати чисельні методи для розв'язання конкретних прикладних задач, забезпечувати необхідні умови їх застосування з погляду збіжності та стійкості, оцінювати похибки обчислень і розробляти, або адаптувати вже існуючі, алгоритми для подальшої програмної реалізації обраних методів;
- мати навички розв'язання обчислювальних задач своєї предметної галузі як за допомогою математичних пакетів (Maple, Matlab, Mathcad і Mathematica), так і власних програм, а також пристосування складних обчислювальних процедур систем математичного моделювання до конкретних особливостей розв'язуваних задач.

Матеріал підручника написано на основі відповідних курсів лекцій, які читають автори в Національному технічному університеті України «Київський політехнічний інститут» і Донецькому національному технічному університеті, а також результатів багаторічних науково-дослідних робіт, що проводяться авторами.

Автори вдячні Навчально-методичній Комісії Міністерства освіти і науки України з підготовки фахівців у галузі комп'ютерних наук за включення цього видання до серії підручників з інформатики для вищих навчальних закладів України та шановним рецензентам і редакторам, зауваження і поради яких сприяли поліпшенню змісту підручника.

Від видавництва

Свої зауваження, пропозиції та запитання надсилайте за адресою електронної пошти pg@bhv.kiev.ua, а також залишайте на сайті http://www.osvita.info. На цьому ж сайті можна отримати детальну інформацію про видання серії «Інформатика» для вищих навчальних закладів.

Інформацію про всі книжки Видавничої групи BHV ви знайдете на сайті http://www.bhv.kiev.ua.

Розділ 1
Вступ до чисельних методів

- Властивості чисельних методів
- Оцінювання складності алгоритмів
- Похибки обчислень
- Операції над матрицями і векторами

Оцінюючи ефективність чисельного методу, слід враховувати такі його властивості, як збіжність, стійкість та простоту реалізації. Застосовуючи метод для розв'язання складних практичних задач, необхідно також завжди мати на увазі, що чисельний розв'язок, як правило, містить деяку похибку, величина якої залежить від багатьох факторів.

1.1. Обчислювальна задача. Чисельні методи та їх особливості

Серед інформаційних технологій, які лежать в основі всіх напрямів підготовки спеціалістів з комп'ютерних технологій, особливе місце займає математичне моделювання. При цьому під *математичною моделлю* фізичної системи, об'єкта або процесу звичайно розуміють сукупність математичних співвідношень (формул, рівнянь, логічних виразів), які визначають характеристики стану і властивості системи, об'єкта і процесу та їх функціонування залежно від параметрів їх компонентів, початкових умов, вхідних збуджень і часу. Загалом математична модель описує функціональну залежність між вихідними залежними змінними, через які відображається функціонування системи, незалежними (такими, як час) і змінюваними змінними (такими, як параметри компонентів, геометричні розміри та ін.), а також вхідними збудженнями, прикладеними до системи.

Згадана функціональна залежність, що відображається математичною моделлю, може бути явною чи неявною, тобто може бути зображена або як просте алгебраїчне співвідношення, або ж як велика за розміром сумісна система диференціально-алгебраїчних рівнянь. До того, як обчислювальна техніка набула широкого розповсюдження, переважали явні функціональні моделі низьких порядків, пристосовані до можливостей розрахунків ручним способом або розрахунків з малим ступенем механізації (логарифмічна лінійка, арифмометр та ін.).

Саме вони і є сьогодні теоретичною основою багатьох інженерних та природничих дисциплін, яка дозволяє під час проектування проводити наближені розрахунки з точністю до кількох десятих відсотка з подальшим обов'язковим макетуванням проектованого об'єкта та його експериментальним доведенням до потрібних параметрів, внаслідок чого розробка нового виробу розтягується на багато років.

Сучасні комп'ютери дозволяють у багатьох випадках відмовитися від натурного макетування проектованих виробів, замінивши його математичним моделюванням (обчислювальним експериментом), що дуже важливо, коли натурне макетування складне або практично неможливе (наприклад, моделювання прориву дамби, переміщення всюдиходу поверхнею Марса та ін.). Але при цьому повинна бути істотно підвищена точність математичних моделей об'єктів та систем, що враховують багато фізичних ефектів та дестабілізуючих чинників, якими раніше нехтували. В результаті розмірність і складність математичних моделей істотно зростають, а їх розв'язання в аналітичному вигляді стає неможливим. Це звичайний для сучасної науки і техніки компроміс, що полягає в отриманні нової якості одного параметра (висока точність обчислювального експерименту і відмова від натурного макетування) за рахунок зменшення чи ускладнення іншого параметра (відмова від звичних для вищої математики аналітичних рішень).

Для кожної математичної моделі звичайно формулюється математична задача. У загальному випадку, коли функціональна залежності для множини вхідних даних (значення незалежних та змінюваних змінних і вхідних збуджень), що виступають як множина аргументів, задана неявно, за допомогою математичної моделі необхідно визначити множину вихідних залежних змінних, що виступають як множина значень функцій. При цьому відповідно до виду математичної моделі розрізняють такі базові типи математичних задач:

- розв'язання системи лінійних (в загальному випадку лінеаризованих) рівнянь;
- розв'язання нелінійних алгебраїчних рівнянь;
- апроксимація масиву даних або складної функції набором стандартних, більш простих функцій;
- чисельне інтегрування і диференціювання;
- розв'язання систем звичайних диференціальних рівнянь;
- розв'язання диференціальних рівнянь в частинних похідних;
- розв'язання інтегральних рівнянь.

Прості математичні задачі малої розмірності, що вивчаються в курсі вищої математики, допускають можливість отримання аналітичних рішень. Складні математичні моделі великої розмірності вимагають застосування чисельних методів, що вивчаються в даному курсі.

Чисельні методи — це математичний інструментарій, за допомогою якого математична задача формулюється у вигляді, зручному для розв'язання на комп'ютері. У такому разі говорять про перетворення математичної задачі в *обчислювальну* задачу. При цьому послідовність виконання необхідних арифметичних і логічних операцій визначається *алгоритмом* її розв'язання. Алгоритм повинен бути

рекурсивним і складатися з відносно невеликих блоків, які багаторазово виконуються для різних вхідних даних.

Слід зазначити, що з появою швидких та потужних цифрових комп'ютерів роль чисельних методів для розв'язання наукових та інженерних задач значно зросла. І хоча аналітичні методи розв'язання математичних задач, як і раніше, дуже важливі, чисельні методи істотно розширюють можливості розв'язання наукових та інженерних задач, не дивлячись на те, що самі рівняння математичних моделей з ускладненням структури сучасних виробів стають погано обумовленими та жорсткими, що істотно ускладнює їх розв'язування. Узявши виконання рутинних обчислень на себе, комп'ютери звільняють час вченого або інженера для творчості: формулювання задач і генерування гіпотез, аналізу та інтерпретації результатів розрахунку тощо.

Чисельні методи забезпечують системний формалізований підхід до розв'язання математичних задач. Проте за умов їх ефективного використання окрім уміння присутня і деяка частка мистецтва, що залежить від здібностей користувача, оскільки для розв'язання кожної математичної задачі існує декілька можливих чисельних методів і їх програмних реалізацій для різних типів комп'ютерів. На жаль, для обрання ефективного способу розв'язання поставленої задачі лише інтуїції замало, потрібні глибокі знання і певні навички. Існує декілька переконливих причин, що мотивують необхідність глибокого вивчення чисельних методів майбутніми фахівцями у галузі комп'ютерно-системної інженерії та прикладної математики.

Чисельні методи є надзвичайно потужним інструментарієм для розв'язання проблемних задач, що описуються довільними нелінійними диференціально-алгебраїчними рівняннями великої розмірності, для яких в даний час не існує аналітичних рішень. Освоївши такі методи, майбутній фахівець набуває здібностей до системного аналізу через математичне моделювання найскладніших задач сучасної науки і техніки.

У своїй майбутній професійній діяльності такий фахівець у першу чергу орієнтуватиметься на використання пакетів сучасних обчислювальних програм, причому те, наскільки правильно він буде їх застосовувати, безпосередньо залежатиме від знання і розуміння ним особливостей і обмежень, властивих чисельним методам, що реалізовані в пакеті. Може трапитися, що одна й та сама математична задача за допомогою певного програмно-технічного комплексу буде одним фахівцем успішно розв'язана, а іншим — ні, оскільки в сучасних пакетах передбачено їх налагоджування для конкретної задачі.

Може з'ясуватися, що низку задач неможливо розв'язати з використанням наявних пакетів програм. Якщо майбутній фахівець знає чисельні методи і володіє навичками програмування, він буде в змозі самостійно провести розробку необхідного алгоритму і програмно його реалізувати, вбудувавши в обчислювальний комплекс.

Вивчення чисельних методів стимулює освоєння самих комп'ютерів, оскільки найкращим способом навчитися програмувати є написання комп'ютерних програм власноруч. Правильно застосувавши чисельні методи, майбутній фахівець зможе пересвідчитися у тому, що комп'ютери успішно розв'язують його

професійні задачі. При цьому він сам відчує вплив похибок обчислень на результат і навчиться контролювати ці похибки.

Вивчення чисельних методів сприяє також переосмисленню і більш глибокому розумінню математики в цілому, оскільки однією із задач чисельних методів є зведення методів вищої математики до виконання простих арифметичних операцій.

Хоча існує безліч чисельних методів, усі вони (як і алгоритми, що їм відповідають) мають багато спільних властивостей і характеристик. Чисельні методи:

- передбачають проведення великої кількості рутинних арифметичних обчислень за допомогою рекурсивних співвідношень, що використовуються для організації *ітерацій*, тобто повторюваних циклів обчислень зі зміненими початковими умовами для поліпшення результату;
- направлені на локальне спрощення задачі, коли, наприклад, використовувані нелінійні залежності лінеаризуються за допомогою своїх обчислених похідних або похідні замінюються різницевими апроксимаціями;
- значно залежать від близкості початкового наближення (або декількох наближень), необхідного для початку обчислень до розв'язку, від властивостей нелінійних функцій, які використовуються в математичних моделях, що накладає обмеження (для забезпечення єдиного розв'язку) на їх диференційованість, на швидкість зміни функцій та ін.;

Чисельні методи характеризуються:

- різною *швидкістю збіжності*, тобто числом ітерацій, виконання яких необхідне для отримання заданої точності розв'язку;
- різною *стійкістю*, тобто збереженням достовірності розв'язку під час подальших ітерацій;
- різною *точністю* отримуваного розв'язку в разі виконання однакового числа ітерацій або циклів обчислень.

Чисельні методи розрізняються:

- за широтою і легкістю застосування, тобто за ступенем своєї *універсальності* та *інваріантності* для розв'язання різних математичних задач;
- за *складністю* їх програмування;
- за можливостями використання у разі їх реалізації наявних бібліотек функцій і процедур, створених для підтримки різних алгоритмічних мов;
- за *ступенем чутливості* до погано обумовлених (або некоректних) математичних задач, коли малим змінам вхідних даних можуть відповідати великі зміни розв'язку.

1.2. Оцінка складності алгоритмів і обчислень

У попередньому підрозділі було введено поняття алгоритму як сукупності операторів, що визначають суть і послідовність операцій, які потрібно виконати для отримання результату під час розв'язання математичної задачі. Побудова алгоритму може бути виконана розбиттям задачі на підзадачі у такий спосіб, щоб вихідні дані однієї підзадачі були вхідними для іншої.

Найважливішою частиною підготовки задачі для комп'ютера є отримання рекурсивної формули для конкретного чисельного методу. Складність обчислень, передбачених алгоритмом, може бути оцінена за допомогою сигнальних функцій $f_A(n)$, які визначають час роботи алгоритму через кількість необхідних операцій, що використовуються алгоритмом у процесі обчислень. При цьому $f_A(n)$ — верхня межа кількості операцій; n — розмірність задачі.

Залежно від складності розрізняють два типи алгоритмів.

1. *Поліноміальний* алгоритм, сигнальна функція якого $f_A(n)$ зростає зі збільшенням n не швидше, ніж деякий поліном $g(n)$. При цьому розрізняють оцінки двох типів (*O-велике* та *o-маленьке*) залежно від співвідношень:

$$f_A(n) = O[g(n)] \Rightarrow \lim_{n\to\infty} \frac{f_A(n)}{g(n)} \neq 0$$

і

$$f_A(n) = o[g(n)] \Rightarrow \lim_{n\to\infty} \frac{f_A(n)}{g(n)} = 0.$$

Так, для алгоритму з кількістю необхідних операцій $f(n) = 2n^5 + 6n^4 + 6n^2 + 18$ сигнальними можуть бути такі функції $f_A(n)$: $O(n^5)$, $o(e^n)$, $o(n^6)$, $o(2^n)$.

2. *Комбінаторний* алгоритм, коли сигнальна функція $f_A(n)$ змінюється як експоненціальна функція або містить у собі оцінки операцій перебору, сполучень, визначення факторіалу $n!$.

Щоб оцінити, що таке факторіал із погляду об'єму обчислень, спробуємо підрахувати час, необхідний для виконання $n = 20!$ ($20! = 24{,}329 \cdot 10^{17}$) операцій, для комп'ютера з середнім часом виконання операції 10^{-7} с. Він становитиме більше 77 століть.

Прикладом комбінаторного алгоритму може бути алгоритм розрахунку визначника матриці через алгебраїчну суму $n!$ його членів, складених із усіх можливих добутків елементів матриці, взятих поодинці в кожному рядку і в кожному стовпці.

Розглянемо п'ять різних алгоритмів за умови, що для виконання окремої операції потрібно 1 мс.

Таблиця 1.1. Значення сигнальної функції для алгоритмів різної складності

Алгоритм	Значення сигнальної функції	Максимальна розмірність задачі		
	$f_A(n)$	1 с	1 хв	1 год
A_1	n	1000	$6 \cdot 10^4$	$3{,}6 \cdot 10^6$
A_2	$n\log n$	140	4893	$2 \cdot 10^6$
A_3	n^2	31	244	1897
A_4	n^3	10	39	153
A_5	2^n	9	15	21

Припустимо, що швидкодія комп'ютера виросла в 10 разів, і перерахуємо дані таблиці.

Таблиця 1.2. Зміна максимальних розмірностей задач у разі збільшення швидкості комп'ютера

Алгоритм	Значення сигнальної функції	Максимальна розмірність задачі	
	$f_A(n)$	До прискорення	Після прискорення
A_1	n	S_1	$10S_1$
A_2	$n\log n$	S_2	$10S_2$
A_3	n^2	S_3	$3{,}6S_3$
A_4	n^3	S_4	$2{,}15S_4$
A_5	2^n	S_5	$S_5 + 3{,}3$

Зміна комп'ютера майже не відобразилася на властивостях алгоритму A_5, тому слід завжди обирати для програмної реалізації найефективніший алгоритм із декількох альтернативних.

Розглянемо процедуру оцінки складності алгоритму на прикладі рекурсивного обчислення значення полінома за схемою Горна:

$$p(z) = a_0z^3 + a_1z^2 + a_2z + a_3 = ((a_0z + a_1)z + a_2)z + a_3,$$

згідно з якою множина початкових коефіцієнтів полінома a_i, $i = 0, 1, ..., n$ перераховується в множину коефіцієнтів b_i, $i = 0, 1, ..., n$ відповідно до алгоритму:

$$b_0 = a_0, \quad b_i = a_i + zb_{i-1}, \quad i = 1, 2, 3,$$

при цьому шукане значення полінома $p(z) = b_3$.

У разі обчислення значення полінома в загальному вигляді $p(z) = a_0z^n + a_1z^{n-1} + ... + a_{n-1}z + a_n$ з урахуванням локальної процедури підвищення степеня його члена $z^i = zz^{i-1}$ кількість необхідних операцій дорівнює $2(n + 1)$ множень і $(n + 1)$ додавань. Обчислюючи значення полінома за схемою Горна, необхідно виконати лише n множень і n додавань, тобто алгоритм Горна вдвічі ефективніший.

1.3. Похибки обчислень

Існують декілька джерел похибок обчислень.

- Похибки вхідних даних і спрощення моделей компонентів.
- Округлення під час обчислень, локальні відсікання.
- Похибки зображення чисел у комп'ютері.

Розрізняють також *глобальну* похибку, як різницю точного і обчисленого значень, і *локальну* похибку, як похибку методу, в основному обумовлену відсіканням частини ряду Тейлора, і тому обмежену значенням першого неврахованого члену цього ряду, пропорційного $O(\Delta x)^{p+1}$, де p — порядок методу обчислення.

Нехай a — точне значення величини; $\tilde{a}$ — наближене значення цієї величини, тоді

$$\varepsilon = |a - \tilde{a}| \quad \text{— абсолютна похибка,}$$

$$\delta(\tilde{a}) = \frac{\varepsilon}{|a|} \quad \text{— відносна похибка.}$$

Під час виконання арифметичних операцій похибки обчислень тільки накопичуються, незалежно від типу виконуваної операції:

$$\Delta(a_1 + a_2) = \Delta a_1 + \Delta a_2, \quad \delta(a_1 + a_2) = \frac{\Delta(a_1) + \Delta(a_2)}{a_1 + a_2},$$

$$\Delta(a_1 - a_2) = \Delta a_1 + \Delta a_2, \quad \delta(a_1 - a_2) = \frac{\Delta(a_1) + \Delta(a_2)}{|a_1 - a_2|},$$

$$\delta(a_1 a_2) = \delta(a_1) + \delta(a_2), \quad \delta(a_1/a_2) = \delta(a_1) + \delta(a_2).$$

Похибки в разі обчислення функції $y = f(x)$ залежно від похибки аргументу $\Delta x = x - x_0$ оцінюються відсіканням частини ряду Тейлора, внаслідок чого одержуємо оцінку абсолютної похибки як $\Delta(f(x)) \cong |f'(x_0)|\Delta x$ і відносної похибки як $\delta[f(x_0)] \cong |f'(x)/f(x_0)|\Delta x$.

Для функції багатьох аргументів $y = f(x_1, x_2, x_3, ..., x_n)$ формула максимальної абсолютної похибки набуває такого вигляду:

$$|\Delta y| = \sum_{i=1}^{n} \left| \frac{\partial y(\tilde{x})}{\partial x_i} \right| |\Delta x_i|. \tag{1.1}$$

У комп'ютері числа з плаваючою комою зображуються у вигляді $a = m \cdot 10^q$, де $0{,}1 \leqslant m \leqslant 1$ — мантиса числа, q — його порядок. Це число m розташовується у лівому краї машинного слова і містить t розрядів.

У загальному випадку

$$\Delta m = \tilde{m} - m \leq \begin{cases} \frac{1}{2} \cdot 10^{-t} \cdot 10^q & \text{(округлювання),} \\ 10^{-t} \cdot 10^q & \text{(відсікання),} \end{cases} \tag{1.2}$$

$$\delta m = \frac{\frac{1}{2} \cdot 10^{-t} \cdot 10^q}{m \cdot 10^q} \leqslant \frac{1}{2} \cdot 10^{1-t} = u.$$

Обумовленість конкретного алгоритму C_A характеризує процес накопичення похибок під час обчислень і визначається як функція від $u = 1/2 \cdot 10^{1-t}$.

Розглядають окремо задачу прямого аналізу похибок, коли відомі: A — збурені вхідні дані та вихідні дані $B = \varphi(A)$ як результат обробки за деяким точним алгоритмом $B_t = \varphi(A_t)$, тоді $M = B_t - B$ — похибка обчислення на комп'ютері даних B, яку потрібно оцінити. Існує і задача оберненого аналізу похибок, коли відомі дані $B_t = \varphi(A_t) = \varphi(A + \varepsilon)$ як результат обробки збурених даних A за точним алгоритмом і потрібно оцінити похибки вхідних даних $\varepsilon = A_t - A$.

Приклад 1.1

Оцінимо похибку обчислення функції $y = x_1^2 - x_2$ за збурених аргументів $x_1 = 1,03 \pm 0,01$, $x_2 = 0,45 \pm 0,01$.

Згідно з формулою (1.1)

$$\left|\frac{\partial y}{\partial x_1}\right| = |2x_1| \leqslant 2,1, \quad \left|\frac{\partial y}{\partial x_2}\right| = |-1| = 1,$$

$$\Delta y \leqslant 2,1 \cdot 0,01 + 1 \cdot 0,01 = 0,031,$$

$$y = 0,611 \pm 0,032.$$

Для зменшення похибок можна використовувати більшу кількість значущих розрядів або змінювати послідовність обчислень. Останнє положення ілюструється таким прикладом випадку компенсації, коли підраховується різниця двох майже рівних чисел $\sqrt[3]{a_1} - \sqrt[3]{a_2}$, $a_1 \approx a_2$, що зумовлює більше зростання відносної похибки. Цього можна уникнути, якщо замінити алгоритм обчислень на такий:

$$\sqrt[3]{a_1} - \sqrt[3]{a_2} = \frac{a_1 - a_2}{\sqrt[3]{a_1^2} + \sqrt[3]{a_1 a_2} + \sqrt[3]{a_2^2}}.$$

Як уже наголошувалося, результат обчислень залежить не тільки від обраного алгоритму розв'язання, але і від обумовленості самої обчислювальної задачі, під якою розуміють залежність максимально можливого відхилення результату від похибок вхідних даних:

$$C_p = \left|\frac{\dfrac{|f(x+\Delta x) - f(x)|}{|f(x)|}}{\dfrac{\Delta x}{x}}\right|, \quad C_p = \left|\frac{\dfrac{f'(x)}{f(x)}}{\dfrac{1}{x}}\right|. \tag{1.3}$$

Прикладом погано обумовленої задачі може бути досить поширена в інженерних розрахунках задача обчислення коренів поліноміальної функції, яка дуже чутлива до точності задавання коефіцієнтів.

Приклад 1.2

Сформуємо слідом за Д. Уілкінсоном [35] поліном 20-го порядку, задавши його корені як числа натурального ряду:

$$P(x) = (x-1)(x-2)...(x-20) = x^{20} - 210x^{19} + ... + 20!.$$

Якщо зараз внести незначну похибку в коефіцієнт другого члена

$$a_{19} = 210 \Rightarrow (210 + 2^{-23}) = 210,000000119,$$

то серед коренів, знайдених за коефіцієнтами полінома, несподівано з'являться десять комплексних коренів.

Для зменшення обумовленості задачі C_p необхідно змінювати форму зображення вхідних і вихідних величин математичної задачі.

Під час реалізації практичних обчислень звичайно загально задана похибка розподіляється по кроках обчислень, а потім на кожному кроці здійснюється контроль похибки за локальною оцінкою, оскільки точні значення глобальної похибки не відомі. Тому в процедурах чисельних методів виділяють три рівноправні частини.

1. Обчислення наближення розв'язку за рекурсивною формулою вибраного алгоритму.
2. Оцінювання похибки.
3. Управління продовженням розв'язування задачі.

Згадане управління може зводитися до зміни кроку приросту аргументів неявно заданої функціональної залежності (зокрема, часового кроку), заміни формули обчислення наближення розв'язку, зміни значень загальної заданої похибки розв'язку і допустимої кількості ітерацій. Зрозуміло, що за всіх інших умов найкращим буде розв'язок, отриманий у результаті виконання меншої кількості ітерацій. І цього можна досягти, зокрема, використовуючи так звану екстраполяцію Річардсона, яка здійснюється таким чином. Припустимо, що ми використовуємо певний алгоритм, що визначає для математичної задачі розв'язок $F(h)$, похибка якого порівняно з точним значенням a_0 пропорційна кроку обчислення h у степені p. Якщо провести розрахунки двічі з різними кроками h і qh, то за допомогою отриманих рівнянь можна знайти точний розв'язок задачі, навіть якщо кроки обчислень будуть великими:

$$\begin{cases} F(h) = a_0 + a_1 h^p + O(h^r), \\ F(qh) = a_0 + a_1 (qh)^p + O(h^r), \end{cases}$$

$$a_0 = F(h) + \frac{F(h) - F(qh)}{q^p - 1} + O(h^r). \qquad (1.4)$$

З формул (1.4) можна отримати значення оцінки глобальної похибки обчислень через різницю двох приблизних розв'язків з урахуванням порядку p використаного методу:

$$\varepsilon = a_0 - F(h) = \frac{F(h) - F(qh)}{q^p - 1}. \qquad (1.5)$$

1.4. Базові операції над матрицями і векторами

Більшість математичних моделей складних об'єктів та систем зручно записувати у вигляді векторно-матричних рівнянь. Нагадаємо основні положення векторно-матричного аналізу, які будуть потрібні нам для подальшого розгляду чисельних методів.

Система $m \times n$ чисел (дійсних чи комплексних), які розташовані у прямокутній таблиці, що складається з m рядків та n стовпчиків

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix},$$

називається матрицею. Числа a_{ij} $(1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n)$ — елементи матриці. Якщо $n = m$, то матрицю називають *квадратною* порядку n, а в загальному випадку її називають *прямокутною* розміром $m \times n$.

Квадратна матриця $\mathbf{A}$ називається *симетричною*, якщо

$$a_{ij} = a_{ji}.$$

Елементи одиничної матриці $\mathbf{E}$ визначаються як

$$a_{ij} = \delta_{ij}, \quad \text{де} \quad \delta_{ij} = \begin{cases} 1, & i = j, \\ 0, & i \neq j, \end{cases}$$

де δ_{ij} називають символом Кронекера.

Визначник (детермінант) матриці обчислюється зі співвідношення:

$$\det \mathbf{A} = |\mathbf{A}| = \sum_{(\alpha_1, \alpha_2, \dots, \alpha_n)} (-1)^{\chi} a_{1\alpha_1} a_{2\alpha_2} \dots a_{n\alpha_n}, \tag{1.6}$$

де операція підсумовування поширена на всі можливі перестановки $(\alpha_1, \alpha_2, \dots, \alpha_n)$ елементів послідовності $1, 2, \dots, n$ і містить $n!$ доданків, причому $\chi = 0$, якщо перестановка парна, і $\chi \neq 0$ $(\chi = 1)$, якщо вона непарна. Квадратну матрицю $\mathbf{A}$ називають *неособливою* (несингулярною), якщо $\det \mathbf{A} \neq 0$. У разі $\det \mathbf{A} = 0$ матриця $\mathbf{A}$ — особлива або *вироджена*.

До матриць застосовні операції транспонування і обернення. Транспонуванням матриці називається заміна її рядків стовпцями. Транспонована матриця позначається як $\mathbf{A}^{\mathrm{T}}$.

Обернена матриця $\mathbf{A}^{-1}$ визначається як матриця, множення якої зліва або справа на матрицю $\mathbf{A}$ дає одиничну матрицю: $\mathbf{A}^{-1}\mathbf{A} = \mathbf{A}\mathbf{A}^{-1} = \mathbf{E}$.

Елементи оберненої матриці $\mathbf{A}^{-1}$ підраховуються за формулою Якобі:

$$\mathbf{A}^{-1} = \frac{[\mathbf{A}_1]^{\mathrm{T}}}{\det \mathbf{A}}, \tag{1.7}$$

де $\mathbf{A}_1$ — приєднана матриця, що складається з алгебраїчних доповнень A_{ij} елементів початкової матриці a_{ij}, які дорівнюють визначнику матриці, одержуваної з початкової матриці $\mathbf{A}$ викресленням рядка i і стовпця j, причому знак A_{ij} визначається парністю суми індексів елемента i та j.

Дійсна матриця **A** називається ортогональною, якщо її транспонована матриця $\mathbf{A}^T$ збігається з оберненою матрицею $\mathbf{A}^{-1}$, тобто

$$\mathbf{A}^T = \mathbf{A}^{-1}, \quad \text{або} \quad \mathbf{A}\mathbf{A}^T = \mathbf{E}.$$

Дві матриці (**A** і **B**), які здійснюють однакові лінійні перетворення в різних базисах, називаються подібними (**A** ~ **B**). Дві матриці подібні тоді і лише тоді, коли одну можна отримати з іншої перетворенням за допомогою якоїсь ортогональної матриці **Q** тобто $\mathbf{B} = \mathbf{Q}^{-1}\mathbf{A}\mathbf{Q}$.

Теорема. Якщо дана квадратна матриця порядку n має n лінійно незалежних *власних векторів*, то, взявши останні за базисні, отримаємо діагональну матрицю, подібну до даної.

Наслідок. Будь-яку квадратну матрицю, *власні значення* якої попарно різні, перетворенням подібності можна звести до діагональної, тобто

$$\mathbf{A} = \mathbf{Q}^{-1}\mathbf{D}\mathbf{Q}, \tag{1.8}$$

де **D** — діагональна матриця власних значень матриці **A**, причому власні значення λ_i і власні вектори $\mathbf{x}_i$ матриці **A** пов'язані за допомогою відомого співвідношення:

$$\mathbf{A}\mathbf{x}_i = \lambda_i \mathbf{x}_i, \quad i = 1, \ldots, n.$$

Приклад 1.3

У математичних пакетах, зокрема і в пакеті Mathematica, передбачені спеціальні оператори `Inverse[A]`, `Transpose[A]`, `Det[A]`, `Eigenvectors[A]`, `Eigenvalues[A]`, за допомогою яких виконуються операції обернення, транспонування, знаходження визначника, власних векторів, власних чисел матриці. Більшість із них буде використана в прикладах до розділів 2–4, наведемо тут лише один із них, а саме оператор обернення матриці. Нехай задана матриця

$$\mathbf{A} = \begin{bmatrix} 1 & 4 \\ -7 & 2 \end{bmatrix}.$$

Тоді обернену матрицю знаходять так:

```
In[]:= A = {{1,4}, {-7,2}}; Inverse[A]
```

```
Out[]= {{1/15, -2/15}, {7/30, 1/30}}
```

ПРИМІТКА

Позначення `In[]` та `Out[]` генеруються самим пакетом Mathematica під час введення вхідних даних і отримання обчислених результатів.

Тобто для нашого прикладу обернена матриця дорівнює:

$$\mathbf{A}^{-1} = \begin{bmatrix} \frac{1}{15} & -\frac{2}{15} \\ \frac{7}{30} & \frac{1}{30} \end{bmatrix},$$

у чому не важко переконатися, користуючись формулою (1.7) і враховуючи, що $\det \mathbf{A} = 30$.

Оцінку і порівняння векторів і матриць проводять, звичайно, через їх норми. Норму p-порядку вектора $\mathbf{x}$ описують за допомогою виразу

$$\|\mathbf{x}\|_p = \left[|x_1|^p + |x_2|^p + \ldots + |x_n|^p\right]^{1/p}, \quad 1 \leqslant p \leqslant \infty.$$

Для випадку $p = 2$ одержуємо квадратичну норму

$$\|\mathbf{x}\|_2 = \left[|x_1|^2 + |x_2|^2 + \ldots + |x_n|^2\right]^{1/2} = (\mathbf{x}^{\mathrm{T}}\mathbf{x})^{1/2}. \tag{1.9}$$

Для випадку $p = \infty$ одержуємо максимальну норму

$$\|\mathbf{x}\|_\infty = \max|x_i|, \quad 1 \leqslant i \leqslant n. \tag{1.10}$$

Приклад 1.4

У математичному пакеті Mathematica серед векторних операцій є операція обчислення норми вектора, що здійснюється за допомогою оператора `VectorNorm[x, p]` і, в залежності від значення параметра p, дозволяє обчислити максимальну чи квадратичну норму. Якщо компоненти вектора задані числами, то норма вектора також буде виражена числом. Наприклад, для заданого вектора $\mathbf{x} = [0.6,\ -1.8,\ 9.6,\ -5.8]^{\mathrm{T}}$ квадратична норма дорівнює:

```
In[]:= <<LinearAlgebra`MatrixManipulation` (* Завантаження пакета LinearAlgebra *)
       VectorNorm[{0.6, -1.8, 9.6, -5.8}, 2]

Out[]= 11.3754
```

Аналогічно обчислюється будь-яка p — норма вектора. Знайдемо, наприклад, норму третього порядку ($p = 3$) для того ж самого вектора $\mathbf{x} = [0,6,\ -1,8,\ 9,6,\ -5,8]^{\mathrm{T}}$.

```
In[]:= VectorNorm[{0.6, -1.8, 9.6, -5.8}, 3]
Out[]= 10.2785
```

Якщо розглядається максимальна норма ($p = \infty$), відповідний оператор пакета Mathematica дещо змінюється:

```
In[]:= VectorNorm[{0.6, -1.8, 9.6, -5.8}, Infinity]
Out[]= 9.6
```

У такий же спосіб вводиться *максимальна норма матриці*, яка просто визначається максимальною сумою абсолютних значень елементів її рядка:

$$\|\mathbf{A}\|_\infty = \max_i \sum_{j=1}^{n} |a_{ij}|, \quad 1 \leqslant i \leqslant n. \tag{1.11}$$

Визначають такі головні властивості норм векторів або матриць:

$$\|\mathbf{ax}\| = |a|\|\mathbf{x}\|;$$

$$\|\mathbf{aA}\| = |a| * \|\mathbf{A}\|;$$

$$\|\mathbf{x} + \mathbf{y}\| \leqslant \|\mathbf{x}\| + \|\mathbf{y}\|;$$

$$\|\mathbf{A} + \mathbf{B}\| \leqslant \|\mathbf{A}\| + \|\mathbf{B}\|;$$

$$\|\mathbf{Ax}\| \leqslant \|\mathbf{A}\|\|\mathbf{x}\|,$$

звідки виводиться ще одна формула для оцінки норми матриці:

$$\|\mathbf{A}\| \leqslant \max \frac{\|\mathbf{Ax}\|}{\|\mathbf{x}\|}, \quad \mathbf{x} \neq 0. \tag{1.12}$$

Приклад 1.5

Максимальну норму матриці можна обчислити за допомогою оператора `MatrixNorm[A,p]` пакета Mathtmatica:

```
In[]:= A = {{1., 4.}, {-7, 2.}}; MatrixNorm[A, Infinity]
Out[]= 9.
In[]:= MatrixNorm[A, 2]
Out[]= 7.28139
```

Над матрицями можна виконувать такі дії [6]:

- додавання, віднімання, коли операції проводяться над елементами з однаковими індексами матриць одного типу;
- множення матриці на дійсне число α, коли множаться всі елементи матриці, при цьому

$$\det \alpha \mathbf{A} = \alpha^n \det \mathbf{A};$$

- множення матриць, виконуване за умови, що число стовпців матриці **A** дорівнює числу рядків матриці **B**, при цьому елементи матриці $\mathbf{C} = \mathbf{AB}$ визначаються за допомогою виразу

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \ldots + a_{in}b_{nj}, \quad i = 1, 2, \ldots, q, \quad j = 1, 2, \ldots, m,$$

який відповідає операції множення двох векторів (i-го рядка матриці **A** і j-го стовпця матриці **B**) і вимагає для виконання n множень і $n-1$ додавань, тобто $2n-1$ операцій. Для всієї процедури множення матриць із урахуванням числа елементів qm матриці **C** буде потрібно nqm множень, $qm(n-1)$ додавань, що в разі множення квадратних матриць ($n = q = m$) дасть верхню оцінку сигнальної функції відповідного поліноміального алгоритму, рівну $f_A(n) = O(n^3)$.

Виявляється, що цю оцінку можна понизити на два порядки, якщо реалізувати процедуру множення матриць на багатопроцесорних паралельних обчислювальних структурах.

Приклад 1.6

Базуючись на даних приклада 1.4, знайдемо за допомогою пакета Mathematica добуток прямої **A** і оберненої $\mathbf{A}^{-1}$ матриць, скориставшись тим, що операція множення матриць позначається крапкою між ними:

```
In[]:= A = {{1, 4}, {-7, 2}};
       B = {{1/15, -2/15}, {7/30, 1/30}};
       H = B . A
Out[]= {{1, 0}, {0, 1}}
```

Як і очікувалося, добуток прямої та оберненої матриць дорівнює одиничній матриці.

1.5. Математичні пакети

Протягом багатьох років у всьому світі накопичено великі бібліотеки комп'ютерних підпрограм, написаних, в першу чергу, мовами FORTRAN та C і призначених для розв'язання типових математичних задач (розв'язання систем лінійних та нелінійних рівнянь, розв'язання диференціальних рівнянь, апроксимації функцій тощо). Крім того, є цілий ряд різних універсальних математичних пакетів, за допомогою яких реалізують різноманітні чисельні методи, а також виконують аналітичні перетворення і розв'язують багато відомих математичних задач аналітично.

Мабуть, найвідомішими сьогодні є такі пакети: Mathematica (компанія Wolfram Research), Maple (компанія Waterloo Maple Inc), MatLab (компанія MathWorks), MathCAD (компанія MathSoft Inc). Вони використовуються у різних сферах діяльності вченими різних спеціальностей, що займаються розв'язанням задач теорії поля, аеродинаміки, космонавтики, математичного моделювання.

Пакет Mathematica є сьогодні найпопулярнішим у наукових колах, особливо серед теоретиків [12, 38]. Даний пакет надає широкі можливості під час проведення символічних (аналітичних) перетворень, проте вимагає значних ресурсів комп'ютера. Пакет Mathematica дозволяє швидко й ефективно розв'язувати багато задач лінійної алгебри, теорії чисел, дискретної математики, математичного аналізу, диференціальних рівнянь. За його допомогою можна виконувати складні алгебраїчні перетворення і спрощення з різними математичними виразами: многочленами, раціональними виразами, елементарними і спеціальними функціями, алгебраїчними і тригонометричними виразами; знаходити скінченні і нескінченні суми, добутки, границі та інтеграли; розв'язувати в символьному вигляді й чисельно алгебраїчні й трансцендентні системи рівнянь; розв'язувати аналітично і чисельно системи звичайних диференціальних рівнянь і деякі класи рівнянь у частинних похідних. У пакеті Mathematica більшість задач розв'язується в діалоговому режимі, без традиційного програмування, з використанням стандартних операторів. Крім того, в цьому пакеті реалізована мова програмування високого рівня, яка дозволяє створювати доволі складні програми. Характерною особливістю пакету є те, що він дозволяє конвертувати документи у формат LaTeX — стандартний формат переважної більшості наукових видавництв світу.

Пакет Maple — найдавніший серед пакетів символьної математики — також дуже популярний у наукових колах [7, 13]. Окрім здійснення аналітичних перетворень за допомогою пакета можна розв'язувати задачі чисельно. Наприклад, ядро пакета Maple-6 студентської версії містить понад три тисячі математичних функцій і правил їх перетворення. Графічний редактор пакета дозволяє одержувати зображення тривимірних фігур із перерізами і конвертувати документи у формат LaTeX. Крім того, ряд інших програмних продуктів використовують інтегрований символічний процесор Maple. Наприклад, пакет підготовки наукових публікацій Scientific WorkPlace (TCI Software Research) дозволяє звертатися до символічного процесора Maple, проводити аналітичні перетворення і вставляти отримані результати в документ.

Як і згадувані вище пакети, MatLab фактично являє собою своєрідну мову програмування високого рівня, орієнтовану переважно на інженерні розрахун-

ки теорії управління, електро- і радіотехніки, а також на моделювання технічних систем [23, 25]. Характерною особливістю пакету є те, що він дозволяє зберігати документи у форматі мови програмування С.

Створювався MatLab як пакет для матричних обчислень. Потім він був оснащений сучасним графічним редактором і доповнений символічним процесором Maple. MatLab став фактично міжнародним стандартом сучасного учбового програмного забезпечення і використовується в багатьох провідних університетах світу і в технічних університетах України та Росії [26].

Характерною особливістю пакету MathCAD [24] є використання звичних стандартних математичних позначень, тобто документ на екрані виглядає так само, як звичайний математичний розрахунок. Даний пакет орієнтований, у першу чергу, на проведення чисельних розрахунків, але має вбудований символічний процесор Maple, що дозволяє виконувати з його допомогою аналітичні перетворення. В останніх версіях передбачена можливість створювати зв'язки документів MathCAD з документами MatLab. На відміну від згаданих вище пакетів, MathCAD є середовищем візуального програмування, тобто не вимагає знання специфічного набору команд.

Останнім часом намітилася тенденція до зближення та інтеграції різних математичних пакетів. Наприклад, останні версії пакетів Mathematica і Maple мають потужні засоби для візуального програмування; в MatLab включена бібліотека аналітичних перетворень Maple; MathCAD дозволяє працювати спільно з MatLab і т. д. Тому для організації практичних занять із чисельних методів підійде будь-який з перелічених вище пакетів, виходячи з можливостей і традицій конкретного навчального закладу.

У даному підручнику пакет Mathematica використовується для виконання обчислень і наочного зображення результатів розв'язку. В багатьох розділах, і перш за все у прикладах, наведено також задачі на виведення розрахункових формул, їхнє обґрунтування, отримання оцінок точності та стійкості. Студентам пропонується запрограмувати й розв'язати їх за допомогою системи Mathematica. Для тих, хто бажає більш досконало освоїти пакет Mathematica і повторити приклади його застосування, наведені в підручнику, пропонується додаткова навчальна література [12].

Звичайно, наявність пакету Mathematica в майбутніх читачів не є обов'язковою умовою використання даного підручника, оскільки всі завдання і приклади можуть бути реалізовані на будь-якій алгоритмічній мові або за допомогою інших наявних математичних пакетів.

Слід відзначити, що згадані математичні пакети (Mathematica, Maple, MatLab, MathCAD) не завжди дозволяють розв'язати ті задачі, які постають перед інженерами й науковцями, і не можуть замінити собою інженерні програмні комплекси комп'ютерного моделювання й проектування виробів і об'єктів сучасної науки і техніки, наприклад системи Cadence, AutoCAD, ANSYS, HSPICE і багато інших, які не лише обробляють математичні задачі великої та дуже великої розмірності, але і самі їх формують за вхідною інформацією щодо структури об'єкту і параметрів його компонентів [22].

Розділ 2

Прямі методи розв'язання систем лінійних рівнянь

- Метод Гаусса
- LU-розкладання матриці
- Метод Холецького
- Уточнення розв'язку
- Обчислення оберненої матриці
- Аналіз точності розв'язку систем
- Числа обумовленості
- Методи розв'язання перевизначених систем

До чисельних методів лінійної алгебри належать методи розв'язання систем лінійних алгебраїчних рівнянь, обернення матриць, обчислення визначників і знаходження власних значень і власних векторів матриць. Завдяки використанню локальної лінеаризації нелінійних залежностей переважна більшість задач обчислювальної математики зводиться до розв'язання систем лінійних алгебраїчних рівнянь. У лінійній алгебрі таку задачу називають першою основною задачею. До неї належать задачі обчислення визначників і обчислення елементів оберненої матриці. Іноді обчислення визначників і елементів оберненої матриці називають другою і третьою основними задачами лінійної алгебри.

2.1. Основні поняття

Нагадаємо основні поняття, що використовуються в цьому розділі. Систему лінійних алгебраїчних рівнянь подамо у такому вигляді:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ \quad \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n, \end{cases} \tag{2.1}$$

або

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, ..., n.$$

Систему лінійних алгебраїчних рівнянь часто записують у матричній формі

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{2.2}$$

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & ... & a_{1n} \\ a_{21} & a_{22} & ... & a_{2n} \\ ... & ... & ... & ... \\ a_{n1} & a_{n2} & ... & a_{nn} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ ... \\ b_n \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ ... \\ x_n \end{bmatrix},$$

де $\mathbf{A}$ — матриця коефіцієнтів системи, $\mathbf{b}$ — вектор вільних членів і $\mathbf{x}$ — вектор невідомих.

Якщо матриця $\mathbf{A}$ неособлива, тобто її визначник не дорівнює нулю, система (2.1) має єдиний розв'язок. У лінійній алгебрі звичайно використовують спосіб розв'язання системи рівнянь (2.2), що заснований на обчисленні оберненої матриці $\mathbf{A}^{-1}$. Дійсно, якщо помножити обидві частини рівняння (2.2) на матрицю $\mathbf{A}^{-1}$, то розв'язок рівняння отримаємо у вигляді

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}. \tag{2.3}$$

Елементи оберненої матриці a_{ij}^{-1} можна обчислити за відомою формулою (1.7)

$$a_{ij}^{-1} = \frac{A_{ij}}{\det \mathbf{A}},$$

де A_{ij} — алгебраїчне доповнення елемента a_{ij} матриці $\mathbf{A}$ і $\det \mathbf{A}$ — визначник цієї матриці. Тоді для знаходження всіх її елементів потрібно знайти значення n^2 визначників порядку n. Остання задача настільки трудомістка, що розв'язати її навіть для $n = 10$ дуже важко.

Менш трудомістким є метод Крамера, у якому значення невідомих x_i, $i = 1, 2, ..., n$ можна отримати за допомогою формули

$$x_i = \frac{\det \mathbf{A}_i}{\det \mathbf{A}}, \quad i = 1, 2, ..., n, \tag{2.4}$$

де матриця $\mathbf{A}_i$ формується з матриці $\mathbf{A}$ заміною її i-го стовпця на стовпець вільних членів. Але для розв'язання системи лінійних алгебраїчних рівнянь з n невідомими у такий спосіб потрібно обчислити $n + 1$ визначників порядку n, що за великого значення n також є дуже трудомісткою операцією, оскільки для розв'язання лінійної системи рівнянь з n невідомими буде потрібно $nn!$ арифметичних операцій. Вже коли $n = 50$, такий об'єм обчислень практично

недоступний сучасним комп'ютерам. Далі ми розглянемо більш зручні для обчислень методи.

Методи чисельного розв'язання системи рівнянь (2.1) діляться на дві групи: прямі та ітераційні. До першої групи належать наведені вище методи розв'язання систем лінійних алгебраїчних рівнянь. В прямих (або точних) методах кількість арифметичних дій, потрібних для отримання розв'язку $\mathbf{x}$ системи (2.1), є скінченним числом. Прикладом прямого методу є також *метод Гаусса*. Ітераційні методи полягають в тому, що розв'язок $\mathbf{x}$ системи (2.1) знаходять як границю послідовних наближень $\mathbf{x}^{(k)}$, коли $k \to \infty$, де k — номер ітерації.

Методи лінійної алгебри на сьогодні добре досліджені та описані в літературі. Є кілька математичних пакетів, з яких, як зазначалося вище, автори вибрали пакет Mathemtica для ілюстративних обчислень і викладок (*study cases*) з огляду на можливість виконувати за допомогою цього пакета операції в символьному вигляді.

Ще раз варто підкреслити, що наявність саме цього математичного пакета не є обов'язковою умовою використання даного підручника: просто без нього читачам доведеться сприймати чисельні приклади «на віру» або повторювати їх за допомогою наявних обчислювальних засобів.

Приклад 2.1

У пакеті Mathematica можна виконувати багато матричних операцій, які дозволяють дуже просто обчислювати визначник матриці, обернену матрицю і розв'язувати систему рівнянь за наведеними вище формулами (2.3), (2.4). Продемонструємо можливості пакета на простих прикладах. Введемо розширену матрицю системи рівнянь $\mathbf{Ax} = \mathbf{b}$:

```
In[]:= Ab = {{1, -6, 0.7, 9}, {0.9, 8, 0.56, 7}, {2, 1.7, 12, 0.8}};
       MatrixForm[Ab]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 1 & -6 & 0.7 & 9 \\ 0.9 & 8 & 0.56 & 7 \\ 2 & 1.7 & 12 & 0.8 \end{pmatrix}$$

Елементи розширеної матриці вводяться послідовно у вигляді списку. Для наочності вона зображена традиційно. Перші три стовпчики в ній — це матриця системи $\mathbf{A}$, останній стовпчик — вектор правих частин рівнянь $\mathbf{b}$. Виділимо з розширеної матриці матрицю системи $\mathbf{A}$:

```
In[]:= A = Ab[[{1, 2, 3}, {1, 2, 3}]]; MatrixForm[A]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 1 & -6 & 0.7 \\ 0.9 & 8 & 0.56 \\ 2 & 1.7 & 12 \end{pmatrix}$$

```
In[]:= b = Ab[[{1, 2, 3}, {4}]]
Out[]= {{9}, {7}, {0.8}}
```

Знайдемо визначник матриці $\mathbf{A}$:

```
In[]:= d = Det[A]
Dut[]= 142.999
```

Він відмінний від нуля. За правилом Крамера знайдемо значення невідомих. Для цього замінимо в матриці системи **A** стовпець, що відповідає номеру невідомої, на стовпець правих частин системи рівнянь. Отримаємо таку матрицю для першої невідомої:

```
In[]:= A1 = A[[{1, 2, 3}, {4, 2, 3}]]; MatrixForm[A1]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 9 & -6 & 0.7 \\ 7 & 8 & 0.56 \\ 0.8 & 1.7 & 12 \end{pmatrix}$$

Тепер за правилом Крамера можемо знайти значення першої невідомої:

```
In[]:= x1 = Det[A1]/d
Out[]= 9.51471
```

Аналогічно знаходимо значення решти невідомих:

```
In[]:= A2 =A[[{1, 2, 3}, {1, 4, 3}]];
       x2 = Det[A2]/d
Out[]= -0.0899587

In[]:= x3 = Det[A[[{1, 2, 3}, {1, 2, 4,}]]]/d
Out[]= -1.50637
```

Використаємо для розв'язання системи формулу (2.3). Спочатку знайдемо обернену матрицю $\mathbf{A}^{-1}$, яку позначимо AI, а потім значення невідомих:

```
In[]:= AI = Inverse[A];
       x = AI . b
Out[]= {{ 9.51471}, { -0.0899587}, { -1.50637}}
```

Якщо це необхідно, можна вивести на екран і обернену матрицю:

```
In[]:= MatrixForm[AI]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 0.664676 & 0.511822 & -0.0626578 \\ -0.0676928 & 0.0741264 & 0.000489514 \\ -0.10119 & -0.0958049 & 0.0937069 \end{pmatrix}$$

Легко перевірити, що, помноживши цю обернену матрицю на її саму, отримаємо одиничну матрицю.

```
In[]:= MatrixForm[AI . A]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 1. & -6.4115 \times 10^{-16} & -2.90891 \times 10^{-16} \\ 0. & 1. & 1.01136 \times 10^{-17} \\ 1.175 \times 10^{-17} & 8.32667 \times 10^{-17} & 1. \end{pmatrix}$$

Похибки обчислення елементів оберненої матриці малі, і можна вважати, що зображена матриця є одиничною. Аналогічні команди і матричні операції присутні й в інших математичних пакетах. Розглянемо більш детально алгоритми, реалізовані такими матричними операціями.

2.2. Метод виключення Гаусса

Розв'язання системи лінійних алгебраїчних рівнянь методом Гаусса полягає в послідовному виключенні невідомих $x_1, x_2, ..., x_n$ із цієї системи. Припустимо, що визначник матриці A відмінний від нуля, що свідчить про те, що система (2.1) має єдиний розв'язок. Якщо $a_{11} \neq 0$, то, поділивши перше рівняння (2.1) на a_{11}, отримаємо:

$$x_1 + c_{12}x_2 + ... + c_{1n}x_n = y_1, \tag{2.5}$$

де

$$c_{1j} = \frac{a_{1j}}{a_{11}}, \quad j = 2, 3, ..., n, \quad y_1 = \frac{b_1}{a_{11}}.$$

Розглянемо решту рівнянь системи (2.1)

$$a_{i1}x_1 + a_{i2}x_2 + ... + a_{in}x_n = b_i, \quad i = 2, 3, ..., n \tag{2.6}$$

і у кожному з них виключимо невідому x_1, виконавши таку послідовність дій. Помножимо (2.5) на a_{i1} і віднімемо отримане рівняння від i-го рівняння системи (2.6), $i = 2, 3, ..., n$.
У результаті отримаємо таку систему рівнянь:

$$\begin{aligned} x_1 + c_{12}x_2 + ... + c_{1j}x_j + ... + c_{1n}x_n &= y_1, \\ a_{22}^{(1)}x_2 + ... + a_{2j}^{(1)}x_j + ... + a_{2n}^{(1)} &= b_2^{(1)}, \\ ... \quad ... \quad ... \quad ... \quad ... & \\ a_{n2}^{(1)}x_2 + ... + a_{nj}^{(1)}x_j + ... + a_{nm}^{(1)}x_n &= b_n^{(1)}. \end{aligned} \tag{2.7}$$

Тут позначено

$$a_{ij}^{(1)} = a_{ij} - c_{1j}a_{i1}, \quad b_i^{(1)} = b_i - y_1a_{i1}, \quad i, j = 2, 3, ..., m. \tag{2.8}$$

У системі (2.7) невідома x_1 є тільки в першому рівнянні, тому надалі достатньо мати справу зі скороченою системою рівнянь:

$$\sum_{j=2}^{n} a_{ij}^{(1)}x_j = b_i^{(1)}, \quad i = 2, 3, ..., m. \tag{2.9}$$

У такий спосіб здійснено перший крок методу Гаусса. Якщо $a_{22}^{(1)} \neq 0$, то з системи (2.9) аналогічно можна виключити x_2 і перейти до системи, яка еквівалентна (2.1). При цьому перше рівняння системи (2.7) залишиться без змін.

Виключаючи послідовно в такий спосіб невідомі $x_3, x_4, ..., x_n$, прийдемо остаточно до системи рівнянь, яка має такий вигляд:

$$\begin{aligned} x_1 + c_{12}x_2 + c_{13}x_3 + ... + c_{1n}x_n &= y_1, \\ x_2 + c_{23}x_3 + ... + c_{2n}x_n &= y_2, \\ ... \quad ... \quad ... \quad ... \quad ... & \\ x_{n-1} + c_{n-1,n}x_n &= y_{n-1}, \\ x_n &= y_n. \end{aligned} \tag{2.10}$$

У матриці цієї системи, що еквівалентна системі (2.1), всі елементи, які розташовані нижче головної діагоналі, дорівнюють нулю. Такі матриці називаються *верхніми трикутними*, на відміну від *нижніх трикутних* матриць, у яких дорівнюють нулю всі елементи, розташовані вище головної діагоналі.

Перехід від системи (2.1) до системи (2.10) являє собою прямий хід методу Гаусса. Зворотний хід полягає в знаходженні невідомих $x_1, x_2, ..., x_n$ з системи (2.10). Це зробити дуже просто, оскільки матриця системи трикутна і за її допомогою можна послідовно, починаючи з x_n, знайти всі невідомі. Загальні формули зворотного ходу мають такий вигляд:

$$x_n = y_n, \quad x_i = y_i - \sum_{j=i+1}^{n} c_{ij}x_j, \quad i = n-1, n-2, ..., 1. \tag{2.11}$$

Тепер виведемо розрахункові формули прямого ходу. Припустимо, що виконані перші $k-1$ кроків, тобто вже виключені змінні $x_1, x_2, ..., x_{k-1}$. Тоді для виключення матимемо аналогічну (2.9) скорочену систему:

$$\begin{aligned} a_{kk}^{(k-1)}x_k + a_{k,k+1}^{(k-1)}x_{k+1} + ... + a_{kn}^{(k-1)}x_n &= b_k^{(k-1)}, \\ ... \quad ... \quad ... \quad ... \quad ... & \\ a_{nk}^{(k-1)}x_k + a_{n,k+1}^{(k-1)}x_{k+1} + ... + a_{nn}^{(k-1)}x_n &= b_n^{(k-1)}. \end{aligned} \tag{2.12}$$

Нехай $a_{kk}^{(k-1)} \neq 0$. Поділивши обидві частини k-го рівняння на $a_{kk}^{(k-1)}$, отримаємо

$$x_k + c_{k,k+1}x_{k+1} + ... + c_{kn}x_n = y_k, \tag{2.13}$$

де

$$c_{kj} = \frac{a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}}, \quad y_k = \frac{b_k^{(k-1)}}{a_{kk}^{(k-1)}}, \quad j = k+1, k+2, ..., n.$$

Помножимо рівняння (2.13) на $a_{ik}^{(k-1)}$ і віднімемо отримане співвідношення з i-го рівняння скороченої системи (2.12), де $i = k+1, k+2, ..., n$. У результаті група рівнянь (2.12) набуде такого вигляду:

$$\begin{aligned} x_k + c_{k,k+1}x_{k+1} + ... + c_{kn}\ x_n &= y_k, \\ a_{k+1,k+1}^{(k)}x_{k+1} + ... + a_{k+1,n}^{(k)}x_n &= b_{k+1}^{(k)}, \\ \cdots\quad \cdots\quad \cdots\quad \cdots\quad \cdots & \\ a_{n,k+1}^{(k)}x_{k+1} + ... + a_{nn}^{(k)}\ x_n &= b_n^{(k)}. \end{aligned}$$

Отже, під час прямого ходу методу Гаусса коефіцієнти системи рівнянь обчислюються за наведеними нижче формулами:

$$\begin{gathered} a_{kj}^{(0)} = a_{kj}, \quad k, j = 1, 2, ..., n, \\ c_{ki} = \frac{a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}}, \quad j = k+1, k+2, ..., n, \quad k = 1, 2, ..., n, \end{gathered} \tag{2.14}$$

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - a_{ik}^{(k-1)}c_{kj}, \quad i, j = k+1, k+2, ..., n, \quad k = 1, 2, ..., n-1. \tag{2.15}$$

Обчислення правих частин системи (2.10) здійснюється за такими формулами:

$$\begin{gathered} b_k^{(0)} = b_k, \quad y_k = \frac{b_k^{(k-1)}}{a_{kk}^{(k-1)}}, \quad k = 1, 2, ..., n, \\ b_i^{(k)} = b_i^{(k-1)} - a_{ik}^{(k-1)}y_k, \quad i = k+1, k+2, ..., n. \end{gathered} \tag{2.16}$$

Основним обмеженням методу є припущення про те, що всі елементи $a_{kk}^{(k-1)}$, на які проводиться ділення на кожному кроці методу, відмінні від нуля. Елемент $a_{kk}^{(k-1)}$ називається ведучим елементом на k-му кроці виключення. Слід мати на увазі, що навіть якщо якийсь ведучий елемент не дорівнює нулю, а просто близький до нього, в процесі обчислень може відбуватися значне накопичення похибок. Уникнути цього дозволяє метод Гаусса з вибором головного елемента. Ідея методу полягає в тому, щоб на черговому кроці виключати ту невідому, коефіцієнт за якої найбільший за модулем. Отже, ведучим елементом тут вибирається головний, тобто найбільший за модулем елемент матриці. Тим самим, якщо $\det \mathbf{A} \neq 0$, то в процесі обчислень не відбуватиметься ділення на нуль.

Для зменшення помилок округлювання чинять так. Серед елементів першого рядка $a_{kj}^{(k-1)}$ кожної проміжної матриці (2.12) вибирають найбільший за модулем елемент $\max |a_{kj}^{k-1}|$, $j = k, k+1, ..., n$ і роблять цей елемент ведучим. Вказаний спосіб виключення називається *методом Гаусса з вибором головного елемента по рядку*. Він еквівалентний застосуванню звичайного методу Гаусса до системи, в якій на кожному кроці виключення проводиться відповідна перенумерація змінних.

Застосовується також метод із вибором головного елемента *по стовпцю* (рис. 2.1, *а*). Він еквівалентний застосуванню звичайного методу Гаусса до сис-

теми, в якій на кожному кроці виключення проводиться відповідна перенумерація рівнянь.

$$\left|a_{rk}^{(k)}\right| = \max_i \left|a_{ik}^{(k)}\right|, \quad k \leqslant i \leqslant n. \tag{2.17}$$

Проте головний елемент можна вибирати і серед всіх елементів неперетвореної частини матриці (рис. 2.1, *б*):

$$\left|a_{rs}^{(k)}\right| = \max_{i,j} \left|a_{ij}^{(k)}\right|, \quad k \leqslant i, \quad j < n. \tag{2.18}$$

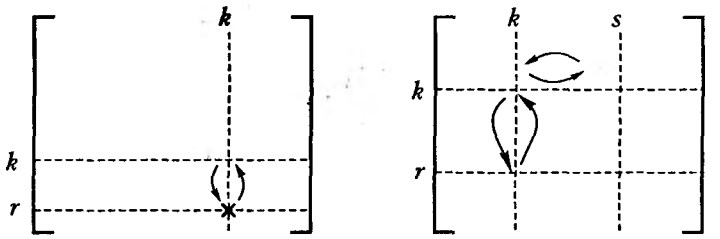


Рис. 2.1. Вибір головного елемента: *а* – серед елементів стовпця матриці; *б* – серед елементів неперетвореної частини матриці

Визначимо складність алгоритму, що реалізує метод Гаусса на однопроцесорній ЕОМ. Розглянемо задачу розв'язання системи лінійних рівнянь з p правими частинами:

$$\mathbf{A}\mathbf{x}_1 = \mathbf{b}_1, \ldots, \mathbf{A}\mathbf{x}_p = \mathbf{b}_p.$$

На кожному k-му кроці прямого ходу Гаусса необхідно виконати $(n-k)$ ділень, $(n-k)(n-k+p+1)$ множень та додавань. Використовуючи формулу для ряду $1^2 + 2^2 + 3^2 + \ldots + m^2 = m(m+1)(2m+1)/6$, отримаємо

$$\sum_{k=1}^{n-1} \{(n-k) + (n-k)(n-k+p+1)\} =$$

$$= \frac{n(n-1)}{2} + \frac{2n^3 - 3n^2 + n}{6} + \frac{n(n-1)(p+1)}{2} =$$

$$= \frac{(n^2-n)(p+2)}{2} + \frac{n^3}{3} - \frac{n^2}{2} + \frac{n}{6} = \frac{n^3}{3} + \frac{n^2(p+1)}{2} - \frac{n(3p+1)}{6}.$$

Під час зворотного ходу необхідно виконати n ділень та $\sum_{k=1}^{n}(k-1) = (n/2)(n-1)$ множень і додавань, тому, коли $p = 1$, маємо

$$f_A(n) = n^3/3 + n^2 = O(n^3). \tag{2.19}$$

Отже, для реалізації методу Гаусса потрібно близько $n^3/3$ операцій множення. Це поліномний алгоритм, але навіть він для великої розмірності задачі ускладнює обчислювальний процес, про що свідчать дані табл. 2.1. У ній наведено тривалість прямого і зворотного ходів залежно від розмірності розв'язуваної системи рівнянь для комп'ютера з середньою швидкістю виконання операції ділення/множення 15 мкс.

Таблиця 2.1. Залежність часу прямого і зворотного ходів від розміру задачі

n	Прямий хід, с	Зворотний хід, с
10	0,005	0,0008
100	5	0,075
1000	5000 (1,5 год)	7,5

Алгоритм Гаусса є послідовним алгоритмом, орієнтованим на виконання на однопроцесорній ЕОМ.

2.3. Розкладання матриці на множники

2.3.1. Зв'язок методу Гаусса з розкладанням матриці на множники

Алгоритм Гаусса можна компактно записати в матричних позначеннях. Він відповідає розкладанню матриці $\mathbf{A}$ на добуток більш простих матриць. Спочатку для наочності розглянемо систему $\mathbf{Ax} = \mathbf{b}$, що складається лише з трьох рівнянь:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1, \\ a_{21}x_1 + a_{22}x_2 + a_{13}x_3 = b_2, \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3. \end{cases} \qquad (2.20)$$

Виключення невідомої x_1 з двох останніх рівнянь системи (2.20) здійснюється виконанням таких операцій:

- ділення першого рівняння на $a_{11} \neq 0$;
- віднімання перетвореного першого рівняння, помноженого на a_{i1}, від рівнянь $i = 2, 3$.

Перша операція еквівалентна множенню системи рівнянь зліва на діагональну матрицю

$$\mathbf{D}_1 = \begin{bmatrix} a_{11}^{-1} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix};$$

друга операція еквівалентна множенню зліва на матрицю

$$\mathbf{Q}_1 = \begin{bmatrix} 1 & 0 & 0 \\ -a_{21} & 1 & 0 \\ -a_{31} & 0 & 1 \end{bmatrix}.$$

Звідси випливає, що виключення x_1 рівносильно множенню системи зліва на матрицю, яку називають елементарною нижньою трикутною матрицею. Виконаємо вказані операції, використавши пакет Mathematica. Введемо матриці $\mathbf{A}_1$, $\mathbf{D}_1$, $\mathbf{Q}_1$ і вектор правих частин $\mathbf{b}$:

```
In[]:= A = Table[a_{i,j}, {i,3}, {j,3}];
       B = Table[b_j, {j,3}];
       D1 = {{a_{1,1}^{-1}, 0, 0}, {0, 1, 0}, {0, 0, 1}};
       Q1 = {{1, 0, 0},{-a2,1, 1, 0},{-a3,1, 0, 1}};
       L1 = Q1 . D1;
```

отримаємо матрицю $\mathbf{L}_1 = \mathbf{Q}_1\mathbf{D}_1$;

$$\mathbf{L}_1 = \begin{bmatrix} 1/a_{11} & 0 & 0 \\ -a_{21}/a_{11} & 1 & 0 \\ -a_{31}/a_{11} & 0 & 1 \end{bmatrix}.$$

Перетворимо за допомогою $\mathbf{L}_1$ початкову систему, тобто запишемо її у вигляді

$$\mathbf{L}_1\mathbf{A}\mathbf{x} = \mathbf{L}_1\mathbf{b}.$$

```
In[]:= A1 = L1 . A;
       B1 = L1 . B;
       X = {{x1}, {x2}, {x3}};
```

У результаті отримаємо систему рівнянь

$$\begin{bmatrix} 1 & \dfrac{a_{12}}{a_{11}} & \dfrac{a_{13}}{a_{11}} \\ 0 & a_{22} - \dfrac{a_{12}a_{21}}{a_{11}} & a_{23} - \dfrac{a_{13}a_{21}}{a_{11}} \\ 0 & a_{32} - \dfrac{a_{12}a_{31}}{a_{11}} & a_{33} - \dfrac{a_{13}a_{31}}{a_{11}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \dfrac{b_1}{a_{11}} \\ b_2 - \dfrac{b_1a_{21}}{a_{11}} \\ b_3 - \dfrac{b_1a_{31}}{a_{11}} \end{bmatrix}. \tag{2.21}$$

Перепишемо її у вигляді

$$\begin{cases} x_1 + a_{12}^{(1)}x_2 + a_{13}^{(1)}x_3 = y_1, \\ a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 = b_2^{(1)}, \\ a_{32}^{(1)}x_2 + a_{33}^{(1)}x_3 = b_3^{(1)} \end{cases} \tag{2.22}$$

і виконаємо другий крок методу Гаусса, тобто виключимо невідому x_2 з останнього рівняння. Це виконується множенням системи (2.22) зліва на елементарну матрицю

$$\mathbf{L}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1/a_{22}^{(1)} & 0 \\ 0 & -a_{32}^{(1)}/a_{22}^{(1)} & 1 \end{bmatrix}. \tag{2.23}$$

Переконаємося в цьому. Елементи $a_{ij}^{(1)}$ і $b_i^{(1)}$ в пакеті Mathematica позначимо через $a1_{ij}$ і $b1_i$. Введемо матрицю і вектор правих частин системи (2.22), а також матрицю $\mathbf{L}_2$:

```
In[]:= A1 = {{1, a112, a113}, {0, a122, a123}, {0, a132, a133}};
       B1 = {y1, b12, b13};
       L2 = {{1, 0, 0}, {0, 1/a122, 0}, {0, -a132/a122, 1}};
       A2 = L2 . A1;
       B2 = L2 . B1;
       X = {{x1}, x2}, {x3}}
```

У результаті отримаємо систему рівнянь

$$\begin{bmatrix} 1 & a_{12}^{(1)} & a_{22}^{(1)} \\ 0 & 1 & \dfrac{a_{23}^{(1)}}{a_{22}^{(1)}} \\ 0 & 0 & a_{33}^{(1)} - \dfrac{a_{23}^{(1)} a_{32}^{(1)}}{a_{22}^{(1)}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ \dfrac{b_2^{(1)}}{a_{22}^{(1)}} \\ b_3^{(1)} - \dfrac{b_2^{(1)} a_{32}^{(1)}}{a_{22}^{(1)}} \end{bmatrix}. \tag{2.24}$$

Отже, після другого кроку виключення приходимо до системи (2.25), яку можна записати в такому вигляді:

$$\mathbf{L}_2\mathbf{L}_1\mathbf{A}\mathbf{x} = \mathbf{L}_2\mathbf{L}_1\mathbf{b}, \tag{2.25}$$

де матриці $\mathbf{L}_1$ і $\mathbf{L}_2$ визначені згідно з (2.21), (2.24). Нарешті, помноживши (2.25) на матрицю

$$\mathbf{L}_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1/a_{33}^{(2)} \end{bmatrix},$$

одержуємо систему

$$\mathbf{L}_3\mathbf{L}_2\mathbf{L}_1\mathbf{A}\mathbf{x} = \mathbf{L}_3\mathbf{L}_2\mathbf{L}_1\mathbf{b}, \tag{2.26}$$

матриця якої

$$\mathbf{U} = \mathbf{L}_3\mathbf{L}_2\mathbf{L}_1\mathbf{A} \tag{2.27}$$

є верхньою трикутною матрицею з одиничною головною діагоналлю.

Перевіримо це, використовуючи пакет Mathematica. Обчислимо

$$a_{33}^{(2)} = a_{33}^{(1)} - \frac{a_{23}^{(1)} a_{32}^{(1)}}{a_{22}^{(1)}},$$

$$y_2 = \frac{b_2^{(1)}}{a_{22}^{(1)}}, \quad b_3^{(2)} = b_3^{(1)} - \frac{b_2^{(1)} a_{32}^{(1)}}{a_{22}^{(1)}}$$

і визначимо $a_{33}^{(2)}$, $\mathbf{L}_3$:

```
In[]:= a233 = a133 - a123a132/a122;
       L3 = {{1, 0 ,0}, {0, 1, 0}, {0, 0, 1/a233}};
```

знайдемо добуток матриць $\mathbf{U} = \mathbf{L}_3\mathbf{A}_2 = \mathbf{L}_3\mathbf{L}_2\mathbf{L}_1\mathbf{A}$.

$$\mathbf{U} = \begin{bmatrix} 1 & a_{12}^{(1)} & a_{22}^{(1)} \\ 0 & 1 & a_{23}^{(2)} \\ 0 & 0 & 1 \end{bmatrix}.$$

Запишемо остаточну систему рівнянь:

```
In[]:= B2 = {y1, y2, b23};
       U = L3 . A2; B3 = L3 . B2;
```

$$\begin{bmatrix} 1 & a_{12}^{(1)} & a_{22}^{(1)} \\ 0 & 1 & a_{23}^{(2)} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \dfrac{b_3^{(2)}}{a_{33}^{(2)}} \end{bmatrix}.$$

З попереднього виразу випливає, що

$$\mathbf{A} = \mathbf{LU}, \tag{2.28}$$

де $\mathbf{L} = \mathbf{L}_1^{-1}\mathbf{L}_2^{-1}\mathbf{L}_3^{-1}$ — нижня трикутна матриця. Отже, LU-розклад матриці $\mathbf{A}$ можна отримати за допомогою елементарних трикутних матриць у такий спосіб: спочатку будують матриці $\mathbf{L}_1$, $\mathbf{L}_2$, $\mathbf{L}_3$ і обчислюють матрицю $\mathbf{U} = \mathbf{L}_3\mathbf{L}_2\mathbf{L}_1\mathbf{A}$, а потім знаходять $\mathbf{L} = \mathbf{L}_1^{-1}\mathbf{L}_2^{-1}\mathbf{L}_3^{-1}$. Відзначимо, що обернені матриці $\mathbf{L}_k^{-1}$ мають простий вигляд:

```
In[]:= MatrixForm[Inverse[L1]]
       MatrixForm[Inverse[L2]]
       MatrixForm[Inverse[L3]]
```

$$\mathbf{L}_1^{-1} = \begin{bmatrix} a_{11} & 0 & 0 \\ a_{21} & 1 & 0 \\ a_{31} & 0 & 1 \end{bmatrix}, \quad \mathbf{L}_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & a_{22}^{(1)} & 0 \\ 0 & a_{32}^{(1)} & 1 \end{bmatrix}, \quad \mathbf{L}_3^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & a_{33}^{(2)} \end{bmatrix}. \tag{2.29}$$

При цьому матриця $\mathbf{L}$ є нижньою трикутною матрицею:

```
In[]:= L = Inverse[L1] . Inverse[L2] . Inverse[L3]
```

$$\mathbf{L} = \begin{bmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22}^{(1)} & 0 \\ a_{31} & a_{32}^{(1)} & a_{33}^{(2)} \end{bmatrix}, \tag{2.30}$$

на головній діагоналі якої розташовані ведучі елементи методу виключення.

2.3.2. Умови застосування методу Гаусса

Все сказане можна перенести і на системи рівнянь довільного порядку. Процес виключення можна записати за допомогою формули

$$\mathbf{L}_n\mathbf{L}_{n-1}\ldots\mathbf{L}_1\mathbf{A}\mathbf{x} = \mathbf{L}_n\mathbf{L}_{n-1}\ldots\mathbf{L}_1\mathbf{b}, \tag{2.31}$$

де елементарна нижня трикутна матриця $\mathbf{L}_k$ на k-му кроці має вигляд

$$\mathbf{L}_k = \begin{bmatrix} 1 & \ldots & 0 & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 1/a_{kk}^{(k-1)} & 0 & \ldots & 0 \\ 0 & \ldots & -a_{k+1,k}^{(k-1)}/a_{kk}^{(k-1)} & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & -a_{nk}^{(k-1)}/a_{kk}^{(k-1)} & 0 & \ldots & 1 \end{bmatrix}. \tag{2.32}$$

Елементарна нижня трикутна матриця $\mathbf{L}_k$ здійснює виключення невідомої x_k з рівнянь із номерами $(k+1), (k+2), \ldots, n$. Матриці $\mathbf{L}_k^{-1}$ існують і мають такий вигляд:

$$\mathbf{L}_k^{-1} = \begin{bmatrix} 1 & \ldots & 0 & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & a_{kk}^{(k-1)} & 0 & \ldots & 0 \\ 0 & \ldots & a_{k+1,k}^{(k-1)} & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & a_{nk}^{(k-1)} & 0 & \ldots & 1 \end{bmatrix}. \tag{2.33}$$

Очевидно, що матриці $\mathbf{L}_k$ існують, коли $a_{kk}^{(k-1)} \neq 0$ для кожного $k = 1, 2, \ldots, n$. Остання умова буде виконана, якщо всі кутові мінори матриці $\mathbf{A}$ відмінні від нуля. Переконаємося в цьому. Позначимо через $|\mathbf{A}_m|$ кутовий мінор матриці $\mathbf{A}$ порядку m:

$$|\mathbf{A}_1| = a_{11},\ |\mathbf{A}_2| = \det\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix},\ \ldots,\ |\mathbf{A}_m| = \det\mathbf{A}.$$

Нехай $\mathbf{A}_m, \mathbf{L}_m, \mathbf{U}_m$ – матриці кутового мінору m-го порядку матриць $\mathbf{A}, \mathbf{L}, \mathbf{U}$, тобто:

$$\mathbf{L}_m = \begin{bmatrix} a_{11} & 0 & 0 & \dots & 0 \\ a_{21} & a_{22}^{(1)} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2}^{(1)} & a_{m3}^{(2)} & \dots & a_{mm}^{(m-1)} \end{bmatrix}, \quad \mathbf{U}_m = \begin{bmatrix} 1 & a_{12}^{(1)} & a_{13}^{(1)} & \dots & a_{1m}^{(1)} \\ 0 & 1 & a_{23}^{(2)} & \dots & a_{2m}^{(2)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix}.$$

Згідно з (2.28)

$$\mathbf{A}_m = \mathbf{L}_m \mathbf{U}_m.$$

Оскільки

$$\det \mathbf{U}_m = |\mathbf{U}_m| = 1, \quad \det \mathbf{L}_m = |\mathbf{L}_m| = a_{11} a_{22}^{(1)} a_{33}^{(2)} \dots a_{mm}^{(m-1)},$$

то

$$|\mathbf{A}_m| = |\mathbf{L}_m||\mathbf{U}_m| = a_{11} a_{22}^{(1)} a_{33}^{(2)} \dots a_{mm}^{(m-1)}.$$

Отже,

$$a_{mm}^{(m-1)} = \frac{|\mathbf{A}_m|}{|\mathbf{A}_{m-1}|} \neq 0, \quad m = 2, 3, \dots, n,$$

оскільки всі кутові мінори матриці $\mathbf{A}$ відмінні від нуля. Якщо хоча б один із кутових мінорів матриці $\mathbf{A}$ дорівнює нулю, то розглянутий вище LU-розклад неможливий. Це легко бачити на прикладі матриць другого порядку. Отже, метод Гаусса можна застосовувати тільки тоді, коли всі кутові мінори матриці $\mathbf{A}$ відмінні від нуля.

Запис методу Гаусса у вигляді (2.31) детально зображує процес виключення. Тепер його можна реалізувати інакше. Нехай задані матриця $\mathbf{A}$ і вектор $\mathbf{b}$. Спочатку проводиться розкладання $\mathbf{A}$ в добуток двох трикутних матриць, $\mathbf{A} = \mathbf{LU}$. Початкова система набуває вигляду $\mathbf{LUx} = \mathbf{b}$, її розв'язання рівносильно послідовному розв'язанню систем рівнянь

$$\mathbf{Ly} = \mathbf{b}, \tag{2.34}$$

$$\mathbf{Ux} = \mathbf{y} \tag{2.35}$$

з трикутними матрицями, звідки й знаходять шуканий вектор $\mathbf{x}$. Розкладання $\mathbf{A} = \mathbf{LU}$ відповідає прямому ходу методу Гаусса, а розв'язання системи (2.34)–(2.35) – зворотному ходу.

Розглянутий вище алгоритм (2.31) приведення системи рівнянь до системи з верхньою трикутною формою ефективно реалізується на паралельних процесорах. Відомо, що систолічний алгоритм перемноження двох матриць розмірності $n \times n$ на SIMD-системі матричного типу реалізується за час, який дорівнює часу виконання n множень, $n-1$ додавань і $3(n-1)$ зсувів, тому зведення системи до вигляду (2.29), тобто прямий хід методу Гаусса, може бути реалізовано виконанням $5n^2 + n$ машинних операцій на матричному процесорі.

Приклад 2.2

Наведемо процедуру для пакета Mathematica, яка дозволяє отримати елементарні матриці виключення невідомих $\mathbf{L}_k$, виключити невідому x_k, обчислити матрицю $\mathbf{A}_k$ і вектор правої частини $\mathbf{b}_k$ системи рівнянь на k-му кроці перетворення Гаусса:

```
In[]:= LU[S_, b_, k_]:= Block[{j, d, P}, d = Dimension[S]; P = IdentityMatrix[d[[1]]];
       P[[k, k]] = 1/S[[k, k]];
       If [Abs[S[[k, k]]] ≤ 10^(-8), Print["a_{k,k} =", S[[k, k]], ", k =", k]; Abort[] ];
       Do [P[[j, k]] = -S[[j, k]]/S[[k, k]], {j, k+1, d[[1]]} ];
       L_k = -P; A_k = P . S; b_k = P . b];
```

Розглянемо приклад її використання. Введемо матрицю і вектор правої частини системи:

```
In[]:= A = {{9, 3, 1}, {-3, 4, 5}, {8, 2, 7}};
       b = Transpose[{6, -4, 5}];
```

Отримаємо всі $\mathbf{L}_k$, $\mathbf{A}_k$, $\mathbf{b}_k$, $k = 1, 2, 3$.

```
In[]:= SA = A; sf = b; Do[LU[SA, sf, i]; sf = b_i; SA = A_i, {i, 3}];
```

Надрукуємо $\mathbf{L}_k$:

```
In[]:= Print["L1 = ", MatrixForm[L1], "L2 = ", MatrixForm[L2], "L3 = ", MatrixForm[L3]];
```

$$\mathbf{L}_1 = \begin{pmatrix} \frac{1}{9} & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ -\frac{8}{9} & 0 & 1 \end{pmatrix}; \quad \mathbf{L}_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{5} & 0 \\ 0 & \frac{2}{15} & 1 \end{pmatrix}; \quad \mathbf{L}_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{45}{307} \end{pmatrix}$$

За формулою (2.27) обчислимо матрицю $\mathbf{U}$:

```
In[]:= U = L1 . L2 . L3 . A; Print["U = ", MatrixForm[U]]
```

$$\mathbf{U} = \begin{pmatrix} 1 & \frac{1}{3} & \frac{1}{9} \\ 0 & 1 & \frac{16}{15} \\ 0 & 0 & 1 \end{pmatrix}$$

Далі визначимо матрицю $\mathbf{L}$:

```
In[]:= L = Inverse[L1] . Inverse[L2] . Inverse[L3]; Print["L =", MatrixForm[L]]
```

$$\mathbf{L} = \begin{pmatrix} 9 & 0 & 0 \\ -3 & 5 & 0 \\ 8 & -\frac{2}{3} & \frac{307}{45} \end{pmatrix}$$

І нарешті за формулою (2.34) знайдемо вектор $\mathbf{y}$:

```
In[]:= y = b3; Print["y = ", MatrixForm[y]]
```

$$\mathbf{y} = \begin{pmatrix} \frac{2}{3} \\ -\frac{2}{5} \\ -\frac{27}{307} \end{pmatrix}$$

За формулою (2.35) знайдемо вектор розв'язку **x**:

```
In[]:= x = Inverse[U] . y; Print["x = ", MatrixForm[x]]
```

$$x = \begin{pmatrix} \frac{239}{307} \\ -\frac{94}{307} \\ -\frac{27}{307} \end{pmatrix}$$

Розв'яжемо ту ж саму задачу за допомогою стандартного оператора Mathematica:

```
In[]:= LinearSolve[A, b]
```

$$\text{Out[]= } x = \left\{\left\{\frac{239}{307}\right\}, \left\{-\frac{94}{307}\right\}, \left\{-\frac{27}{307}\right\}\right\}$$

2.3.3. Матрична форма методу Гаусса з вибором головного елемента

Можна отримати формальний запис методу Гаусса з вибором головного елемента, використовуючи матриці перестановок, які визначаються таким чином.

Матрицею перестановок **P** називається квадратна матриця, у якої в кожному рядку і в кожному стовпці наявний лише один відмінний від нуля і рівний одиниці елемент.

Елементарною матрицею перестановок $\mathbf{P}_{ki}$ називається матриця, отримана з одиничної матриці перестановкою k-го і i-го рядків. Наприклад, елементарними матрицями перестановок третього порядку є матриці

$$\mathbf{P}_{12} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{P}_{13} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{P}_{23} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

Перелічимо властивості елементарних матриць перестановок, що випливають з їх визначення.

1. Добуток двох (а отже, і будь-якої кількості) елементарних матриць перестановок є матрицею перестановок (не обов'язково елементарною).

   Наведемо приклади, використовуючи пакет Mathematica:

```
In[]:= P12 = {{0, 1, 0}, {1, 0, 0}, {0, 0, 1};
       P13 = {{0, 0, 1}, {0, 1, 0}, {1, 0, 0};
       P23 = {{1, 0, 0}, {0, 0, 1}, {0, 1, 0};
       MatrixForm[P12 . P13]
       MatrixForm[P12 . P23]
```

$$\mathbf{P}_{12}\mathbf{P}_{13} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \quad \mathbf{P}_{12}\mathbf{P}_{23} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

2. Матрицю $\mathbf{P}_{ki}\mathbf{A}$ отримують із матриці $\mathbf{A}$ перестановкою k-го і i-го рядків.

3. Матрицю $\mathbf{AP}_{ki}$ отримують із матриці $\mathbf{A}$ перестановкою k-го і i-го стовпців. Нижче наведені відповідні приклади:

```
In[]:= A = Table[aij, {i,3}, {j,3}];
       MatrixForm[A]
       MatrixForm[P12 . A]
       MatrixForm[A . P12]]
```

$$\mathbf{A}=\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad \mathbf{P}_{12}\mathbf{A}=\begin{bmatrix} a_{21} & a_{22} & a_{23} \\ a_{11} & a_{12} & a_{13} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad \mathbf{AP}_{12}=\begin{bmatrix} a_{12} & a_{11} & a_{13} \\ a_{22} & a_{21} & a_{23} \\ a_{32} & a_{31} & a_{33} \end{bmatrix}.$$

Приклад 2.3

Розв'яжемо систему рівнянь методом Гаусса з вибором головного елемента по стовпцю:

$$\begin{aligned} 2x_1 - 5x_2 + 4x_3 &= 1, \\ 3x_1 + 2x_2 - x_3 &= 8, \\ 4x_1 - x_2 - 2x_3 &= -3. \end{aligned}$$

У першому стовпці найбільший елемент знаходиться в третьому рядку. Переставимо ці рядки за допомогою матриці $\mathbf{P}_{13}$:

```
In[]:= A = {{2, -5, 4}, {3, 2, -1}, {4, -1, 2}};
       b = {1, 8, 3};  X = {x1, x2, x3};
       P13 = {{0, 0, 1}, {0, 1, 0}, {1, 0, 0}};
       A1 = P13 . A; b1 = b . P13;
```

Отже, ми ввели матрицю $\mathbf{A}$ і вектор правої частини початкової системи рівнянь, переставили рядки і отримали нову матрицю $\mathbf{A}_1$, вектор $\mathbf{b}_1$ і систему рівнянь

$$\mathbf{P}_{13}\mathbf{A}\mathbf{x} = \mathbf{P}_{13}\mathbf{b}$$

```
In[]:= Print[ MatrixForm[A1], MatrixForm[X], "=", MatrixForm[b1]]
```

$$\begin{pmatrix} 4 & -1 & -2 \\ 3 & 2 & -1 \\ 2 & -5 & 4 \end{pmatrix}\begin{pmatrix} x_3 \\ x_2 \\ x_1 \end{pmatrix} = \begin{pmatrix} -3 \\ 8 \\ 1 \end{pmatrix}$$

Виключимо невідому x_1 з двох останніх рівнянь. Для цього введемо елементарну нижню матрицю $\mathbf{L}_1$ і перетворимо систему з її допомогою. В результаті переходимо до системи

$$\mathbf{L}_1\mathbf{P}_{13}\mathbf{A}\mathbf{x} = \mathbf{L}_1\mathbf{P}_{13}\mathbf{b}$$

```
In[]:= L1 = {{0.25, 0, 0}, {-0.75, 1, 0}, {-0.5, 0, 1}};
       A2 = L1 . A1; b2 = L1 . b1;
       Print[MatrixForm[A2], MatrixForm[X], "=", MatrixForm[b2]]
```

$$\begin{pmatrix} 1 & -0.25 & -0.5 \\ 0 & 2.75 & 0.5 \\ 0 & -4.5 & 5.0 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -0.75 \\ 10.25 \\ 2.5 \end{pmatrix}$$

У разі виключення невідомої x_2 з останнього рівняння виберемо найбільший за модулем елемент другого стовпця, який розташований у третьому рядку. Переставимо другий і третій рядки й отримаємо таку систему:

$$\mathbf{P}_{23}\mathbf{L}_1\mathbf{P}_{13}\mathbf{A}\mathbf{x} = \mathbf{P}_{23}\mathbf{L}_1\mathbf{P}_{13}\mathbf{b}$$

```
In[]:= P23 = {{1, 0, 0}, {0, 0, 1}, {0, 1, 0}};
       A3 = P23 . A2; b3 = P23 . b2;
       Print[MatrixForm[A3], MatrixForm[X], "=", MatrixForm[b3]]
```

$$\begin{pmatrix} 1 & -0.25 & -0.5 \\ 0 & -4.5 & 5.0 \\ 0 & 2.75 & 0.5 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -0.75 \\ 2.5 \\ 10.25 \end{pmatrix}$$

Виключимо невідому x_2 із останнього рівняння. Для цього утворимо елементарну нижню матрицю $\mathbf{L}_2$ і перетворимо систему з її допомогою. Вона набуде такого вигляду:

$$\mathbf{L}_2\mathbf{P}_{23}\mathbf{L}_1\mathbf{P}_{13}\mathbf{A}\mathbf{x} = \mathbf{L}_2\mathbf{P}_{23}\mathbf{L}_1\mathbf{P}_{13}\mathbf{b}$$

```
In[]:= L2 = {{1, 0, 0}, {0, -1/4.5, 0}, {0, 2.75/4.5, 1}};
       A4 = L2 . A3; b4 = L2 . b3;
       Print[MatrixForm[A4], MatrixForm[X], "=", MatrixForm[b4]]
```

$$\begin{pmatrix} 1 & -0.25 & -0.5 \\ 0 & 1 & -1.11111 \\ 0 & 2.77556\times10^{-16} & 3.55556 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -0.75 \\ -0.55556 \\ 11.7778 \end{pmatrix}$$

Заключний крок прямого ходу методу Гаусса полягає в знаходженні значення невідомого x_2, що еквівалентно множенню останнього рівняння на матрицю $\mathbf{L}_3$

```
In[]:= L3 = {{1, 0, 0}, {0, 1, 0}, {0, 0, 1/355556}};
       A5 = L3 . A4; b5 = L3 . b4;
       Print[MatrixForm[A5], MatrixForm[X], "=", MatrixForm[b5]]
```

$$\begin{pmatrix} 1. & -0.25 & -0.5 \\ 0. & 1. & -1.11111 \\ 0. & 7.80625\times10^{-17} & 0.999999 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} -0.75 \\ -0.555556 \\ 3.3125 \end{pmatrix}$$

Отже, для розглянутого прикладу процес виключення Гаусса з вибором головного елемента записується у такий спосіб:

$$\mathbf{L}_3\mathbf{L}_2\mathbf{P}_{23}\mathbf{L}_1\mathbf{P}_{13}\mathbf{A}\mathbf{x} = \mathbf{L}_3\mathbf{L}_2\mathbf{P}_{23}\mathbf{L}_1\mathbf{P}_{13}\mathbf{b}. \tag{2.36}$$

При цьому матриця

$$\mathbf{U} = \mathbf{L}_3\mathbf{L}_2\mathbf{P}_{23}\mathbf{L}_1\mathbf{P}_{13}\mathbf{A} \tag{2.37}$$

є верхньою трикутною матрицею з одиничною головною діагоналлю.

Відмінність від звичайного методу Гаусса полягає в тому, що як співмножники в (2.37) разом із елементарними трикутними матрицями $\mathbf{L}_k$ можуть бути присутні елементарні матриці перестановок $\mathbf{P}_{ki}$.

Покажемо, що з (2.37) випливає розкладання

$$\mathbf{PA} = \mathbf{LU}, \tag{2.38}$$

де $\mathbf{L}$ — нижня трикутна матриця, яка має обернену матрицю, і $\mathbf{P}$ — матриця перестановок. Для цього знайдемо матрицю

$$\tilde{\mathbf{L}}_1 = \mathbf{P}_{23}\mathbf{L}_1\mathbf{P}_{23}.$$

На основі другої властивості елементарних матриць $\mathbf{P}_{23}\mathbf{L}_1$ обчислюють із матриці $\mathbf{L}_1$ перестановкою другого і третього рядків:

```
In[]:= MatrixForm[P23 . L1]
```

$$\mathbf{P}_{23}\mathbf{L}_1 = \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ -\frac{1}{2} & 0 & 1 \\ -\frac{3}{4} & 1 & 0 \end{bmatrix}.$$

Матриця $\tilde{\mathbf{L}}_1$ відповідно до третьої властивості обчислюється з $\mathbf{P}_{23}\mathbf{L}_1$ перестановкою другого і третього стовпців:

```
In[]:= MatrixForm[P23 . L1 . P23]]
```

$$\tilde{\mathbf{L}}_1 = \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ -\frac{3}{4} & 0 & 1 \end{bmatrix}, \tag{2.39}$$

тобто $\tilde{\mathbf{L}}_1$ — нижня трикутна матриця, що має обернену матрицю.

З виразу (2.39), враховуючи рівність $\mathbf{P}_{23} = \mathbf{P}_{23}^{-1}$, отримаємо

$$\tilde{\mathbf{L}}_1\mathbf{P}_{23} = \mathbf{P}_{23}\mathbf{L}_1. \tag{2.40}$$

Звідси і з (2.37) бачимо, що

$$\mathbf{U} = \mathbf{L}_3\mathbf{L}_2\tilde{\mathbf{L}}_1\mathbf{P}_{23}\mathbf{P}_{13}\mathbf{A} = \mathbf{L}^{-1}\mathbf{PA},$$

де позначено $\mathbf{P} = \mathbf{P}_{23}\mathbf{P}_{13}$, $\mathbf{L} = \tilde{\mathbf{L}}_1^{-1}\mathbf{L}_2^{-1}\mathbf{L}_3^{-1}$.

Оскільки $\mathbf{P}$ — матриця перестановок і $\mathbf{L}$ — нижня трикутна матриця, розкладання (2.38) доведено. Воно означає, що метод Гаусса з вибором головного елемента по стовпцю еквівалентний звичайному методу Гаусса, який застосо-

вується до матриці **PA**, тобто до системи, отриманої з початкової системи перестановкою деяких рівнянь. У даному прикладі матриця перестановок **P** має такий вигляд:

```
In[]:= PS = P23 . P13;
       P = MatrixForm[PS]
```

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

Перетворена система рівнянь, застосування до якої звичайного методу Гаусса еквівалентно застосуванню методу Гаусса з вибором головного елемента по стовпцю, запишеться так:

```
In[]:= MatrixForm[PS . A]
       MatrixForm[X]
       MatrixForm[PS . b]
```

$$\begin{bmatrix} 4 & -1 & -2 \\ 2 & -5 & 4 \\ 3 & 2 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -3 \\ 1 \\ 8 \end{bmatrix}.$$

Результат, отриманий тут для окремого прикладу, справедливий і для будь-якої системи рівнянь. Взагалі метод Гаусса з вибором головного елемента по стовпцю можна записати в такому вигляді:

$$\begin{aligned} &\mathbf{L}_n\mathbf{L}_{n-1}\mathbf{P}_{n-1,J_{n-1}}\mathbf{L}_{n-2} \ldots \mathbf{L}_2\mathbf{P}_{2,J_2}\mathbf{L}_1\mathbf{P}_{1,J_1}\mathbf{A}\mathbf{x} = \\ &= \mathbf{L}_n\mathbf{L}_{n-1}\mathbf{P}_{n-1,J_{n-1}}\mathbf{L}_{n-2} \ldots \mathbf{L}_2\mathbf{P}_{2,J_2}\mathbf{L}_1\mathbf{P}_{1,J_1}\mathbf{b}, \end{aligned} \tag{2.41}$$

де $\mathbf{P}_{k,J_k}$ — елементарні матриці перестановок і $\mathbf{L}_k$ — елементарні нижні трикутні матриці.

Звідси, використовуючи співвідношення перестановок, аналогічні (2.40), можна показати, що метод Гаусса з вибором головного елемента по стовпцю еквівалентний звичайному методу Гаусса, який застосовують до системи

$$\mathbf{PAx} = \mathbf{Pb}, \tag{2.42}$$

де **P** — деяка матриця перестановок.

Отже, якщо $\det \mathbf{A} \neq 0$, то існує матриця перестановок **P** така, що матриця **PA** має відмінні від нуля кутові мінори.

І таким чином, якщо $\det \mathbf{A} \neq 0$, то існує матриця перестановок **P** така, що справедливе розкладання

$$\mathbf{PA} = \mathbf{LU}, \tag{2.43}$$

де **L** — нижня трикутна матриця з відмінними від нуля діагональними елементами і **U** — верхня трикутна матриця з одиничною головною діагоналлю.

Приклад 2.4

У пакеті Mathematica існують оператори виключення Гаусса, що реалізовані на основі LU-розкладання матриці **PA** системи рівнянь. Для того щоб скористатися ними, необхідно спочатку відкрити пакет

```
In[]:= <<LinearAlgebra`GaussianElimination`
```

Введемо матрицю системи рівнянь і обчислимо її визначник, який відмінний від нуля

```
In[]:= A = {{1, 4, -5}, {12, -1, 10}, {4, 8, -3}}; Det[A]
Out[]= -273
```

Виконаємо LU-розкладання матриці за допомогою оператора

```
In[]:= A1 = LUFactor[A]
```

$$\text{Out[]= Lu}\left[\left\{\left\{\frac{1}{12}, \frac{49}{100}, -\frac{273}{100}\right\}, \{12, -1, 10\}, \left\{\frac{1}{3}, \frac{25}{3}, -\frac{19}{3}\right\}\right\}, \{2, 3, 1\}\right]$$

Нами отримана відповідь у вигляді списку, в який компактно підставлені матриці **L** і **U**, обчислені в результаті виключення невідомих із вибором головного елемента по стовпцю. Тепер можна отримати розв'язок системи рівнянь із матрицею **A** для будь-якої правої частини **b**:

```
In[]:= b = {1, -2, 5}; X = LUSolve[A1, b]
```

$$\text{Out[]= } \left\{-\frac{22}{39}, \frac{44}{39}, \frac{23}{39}\right\}$$

Компактний запис матриць **L** і **U** може бути отриманий за допомогою оператора

```
In[]:= A1[[1]].
```

$$\text{Out[]= } \left[\left\{\left\{\frac{1}{12}, \frac{49}{100}, -\frac{273}{100}\right\}, \{12, -1, 10\}, \left\{\frac{1}{3}, \frac{25}{3}, -\frac{19}{3}\right\}\right\}\right]$$

Вектор $\mathbf{p} = \{2, 3, 1\}$ визначає порядок рівнянь у разі виключення невідомих із вибором головного елемента по стовпцю. В даному випадку спочатку міняються місцями друге та перше рівняння. На другому кроці третє рівняння стає другим і відбувається виключення невідомої x_2. Цим перестановкам відповідає матриця **P**:

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

У відповідності з формулою (2.43) отримали розклад матриці $\mathbf{PA} = \mathbf{LU}$, що з точністю до порядку рядків, який визначається вектором $\mathbf{p} = [2, 3, 1]$, дає компактний запис матриць **L** і **U**. Самі матриці можна отримати, помноживши матрицю `A1[[1]]` на матрицю перестановок

```
In[]:= lu = P . A1[[1]]
```

$$\text{Out[]= } \left\{\{12, -1, 10\}, \left\{\frac{1}{3}, \frac{25}{3}, -\frac{19}{3}\right\}, \left\{\frac{1}{12}, \frac{49}{100}, -\frac{273}{100}\right\}\right\}$$

На головній діагоналі матриці **LU** і над нею розташовані елементи матриці **U**, нижче головної діагоналі — елементи матриці **L**. Причому елементи на головній діагоналі матриці **L** дорівнюють одиниці. Отримаємо матриці **L** і **U**:

```
In[]:= U = lu*Table[If[i<=j, 1, 0], {i, Length[lu]}, {j, Length[lu]}]; MatrixForm[U]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 12 & -1 & 10 \\ 0 & \frac{25}{3} & -\frac{19}{3} \\ 0 & 0 & -\frac{273}{3} \end{pmatrix}$$

```
In[]:= L = lu - U + IdetityMatrix[Length[lu]]; MatrixForm[L]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{1}{12} & \frac{49}{100} & 1 \end{pmatrix}$$

Для перевірки знайдемо добуток **LU**:

```
In[]:= MatrixForm[L . U]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 12 & -1 & 10 \\ 4 & 8 & -3 \\ 1 & 4 & -5 \end{pmatrix}$$

Отримана матриця ідентична матриці **PA**. Помітно, що структура матриць **L** і **U** в пакеті Mathematica відрізняється від структури цих матриць, прийнятої в (2.43). Там матриця **U** має одиничну діагональ, а в Mathematica матриця **L** має одиничну діагональ. В обох випадках вираз (2.43) справедливий. На закінчення перевіримо отриманий вище розв'язок **x**:

```
In[]:= Z = {z1, z2, z3}; Solve[A . Z == b, Z]
```

$$\text{Out} = \left\{\left\{z1 \to -\frac{22}{39},\ z2 \to \frac{44}{39},\ z3 \to \frac{23}{39}\right\}\right\}$$

2.4. Алгоритми LU-розкладання матриці без операцій матричного множення

2.4.1. Базові формули перетворення

Трикутні матриці **L** та **U**, на які розкладається матриця **A**, на однопроцесорних ЕОМ одержують не виконанням наведеної вище процедури матричних перетворень, що ґрунтуються на використанні послідовності елементарних нижніх

трикутних матриць $\mathbf{L}_k$, а за допомогою прямих рекурентних формул перетворення, отриманих на основі формул множення матриць. Дійсно,

$$a_{sj} = \sum_{k=1}^{j} l_{sk} u_{kj}, \quad s > j, \tag{2.44}$$

$$a_{sj} = \sum_{k=1}^{s} l_{sk} u_{kj}, \quad s \leqslant j, \tag{2.45}$$

тому що

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ a_{31} & a_{32} & \dots & a_{3n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ l_{21} & 1 & 0 & \dots & 0 \\ l_{31} & l_{32} & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ l_{n1} & l_{n2} & l_{n3} & \dots & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & & \dots & u_{1n} \\ 0 & u_{22} & u_{23} & \dots & u_{2n} \\ 0 & 0 & u_{33} & \dots & u_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & u_{nn} \end{bmatrix},$$

при цьому $l_{sk} = 0$ для $k > s$ і $u_{kj} = 0$ для $k > j$.
Оскільки вибрані значення $l_{ss} = 1$, то з виразу (2.44) випливає:

$$a_{sj} = u_{sj} + \sum_{k=1}^{s-1} l_{sk} u_{kj}, \quad \text{звідки} \quad u_{sj} = a_{sj} - \sum_{k=1}^{s-1} l_{sk} u_{kj}.$$

З виразу (2.45) знаходимо:

$$a_{sj} = l_{sj} u_{jj} + \sum_{k=1}^{j-1} l_{sk} u_{kj}, \quad \text{звідки} \quad l_{sj} = \frac{a_{sj} - \sum_{k=1}^{j-1} l_{sk} u_{kj}}{u_{jj}}, \; s = j+1, \dots$$

Отже, елементи нижньої $\mathbf{L}$ і верхньої $\mathbf{U}$ матриць обчислюються за формулами:

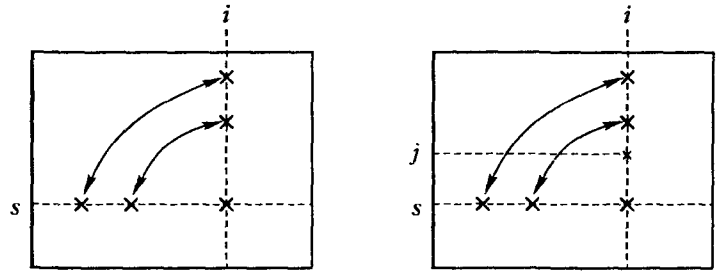
$$\left.\begin{aligned} u_{sj} &= a_{sj} - \sum_{k=1}^{s-1} l_{sk} u_{kj}, \quad j = s, \dots, m, \\ l_{is} &= \frac{a_{is} - \sum_{k=1}^{s-1} l_{ik} u_{ks}}{u_{ss}}, \quad i = s+1, \dots, n, \\ & s = 1, 2, \dots, n, \end{aligned}\right\} \tag{2.46}$$

де $u_{ss} \neq 0$. Нижче, в підрозділі 2.5.3, буде показано можливість застосування формул (2.46) і для випадку $u_{ss} = 0$.

Формули (2.34), (2.35) зводяться до

$$\begin{aligned} y_i &= b_i - \sum_{s=1}^{i-1} l_{is} y_s, \quad i = 1, 2, \dots, n, \\ x_i &= \frac{1}{u_{ii}} \left(y_i - \sum_{s=i+1}^{n} u_{is} x_s \right), \quad i = 1, 2, \dots, n. \end{aligned} \tag{2.47}$$

Порядок віднімання попарних добутків елементів l_{sk} і u_{kj}, розташованих у рядку s і стовпці j, у разі перетворення елемента початкової матриці **A** в елемент матриці **L** або **U** показано на рис. 2.2. При цьому елемент a_{sj}, розташований на діагоналі матриці **A** або справа від неї, перетворюється у відповідний елемент матриці **U** відніманням від його початкового значення вказаних попарних добутків елементів l_{sk} і u_{kj} (рис. 2.2, *а*), а елемент a_{sj}, розташований нижче діагоналі матриці **A**, перетворюється у відповідний елемент матриці **L** відніманням від його початкового значення вказаних попарних добутків елементів l_{sk} і u_{kj} та нормування результату за значенням u_{jj} (рис. 2.2, *б*).



а *б*

Рис. 2.2. Перетворення елементів матриці A на елементи матриць U і L

Неважко бачити, що формули (2.46) обчислення елементів матриць **L** і **U** практично збігаються з формулами (2.14) і (2.15) методу Гаусса для перерахунку значень елементів матриці **A** у разі послідовного обнулювання елементів стовпців, що лежать нижче від діагонального елемента, особливо для випадку, коли віднімання попарних добутків елементів l_{sk} і u_{kj} здійснюється для всіх елементів неперетвореної частини матриці **A** на кожному кроці, пов'язаному з перетворенням елементів поточного рядка (стовпця) матриці **A**. Тому складність LU-розкладання, що виконується на однопроцесорній ЕОМ, оцінюється також величиною

$$f_A(n) = n^3/3 + n^2 = O(n^3),$$

до якої додається складність розв'язку двох систем лінійних рівнянь із трикутними матрицями (2.47), що складає приблизно n^2 операцій множення/ділення.

Цей показник набагато менший, ніж у випадку обчислення трикутних матричних множників за формулою (2.31) на однопроцесорній ЕОМ, коли потрібне виконання n множень двох матриць, або в цілому n^4 операцій множення/ділення, якщо врахувати складність процедури множення двох матриць (1.18).

У порівнянні з методом Гаусса основною перевагою LU-розкладання, яка випливає з формул (2.47), є можливість за знайдених **L** і **U** та різних векторах правої частини **b** за результатами попереднього розкладання багатократно повторювати тільки зворотний хід розв'язання системи лінеаризованих рівнянь.

Ця перевага дозволяє також на порядок зменшити кількість операцій множення/ділення, які необхідно виконати для обчислення оберненої матриці.

Приклад 2.5

Нехай задана матриця A розмірності 3×3. Використовуючи формули (2.46), послідовно перетворимо елементи початкової матриці спочатку в загальному вигляді:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \Rightarrow \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ l_{21} & u_{22} & u_{23} \\ l_{31} & l_{32} & u_{33} \end{bmatrix},$$

де

$$u_{11} = a_{11}, \qquad u_{12} = a_{12}, \qquad u_{13} = a_{13},$$

$$l_{21} = \frac{a_{22}}{u_{11}}, \qquad u_{22} = a_{22} - l_{21}u_{12}, \qquad u_{23} = a_{23} - l_{21}u_{13};$$

$$l_{31} = \frac{a_{31}}{u_{11}}; \qquad l_{32} = \frac{1}{u_{22}}(a_{32} - l_{31}u_{12}); \qquad u_{33} = a_{33} - u_{13}l_{31} - u_{23}l_{32}.$$

Код LU-розкладання заданої матриці в пакеті Mathematica має такий вигляд:

```
In:= A = {{9, 3, 1}, {-3, 4, 5}, {8, 2, 7}}; n = Length[A];
     U = Table[0, {n}, {n}];
     L = IdentityMatrix[n];
     If [A[[1, 1]] == 0, Print ["a [1, 1] =", A [[1, 1]]]; Abort[]]
     Do [{ Do [{summ = 0., Do [{summ = summ + L[[s,k]]*U[[k,j]] }, {k, 1, s-1}],
        U[[s,,j]] = A[[s,j]] - summ}, {j, s, n}],
           Do [{summ = 0., Do [{summ = summ + L[[i,k]]*U[[k,s]] }, {k, 1, s-1}],
     If [Abs[U[s,s]] <= 10^(-10), {Print ["U[s,s] =", U[s,s],", S =", S], Abort []}],
        L[[i,s]] = (A[[i,s]] - summ)/U[[s,s]] }, {i, s+1, n}]}, {s, 1, n}];
     MatrixForm[L]
     MatrixForm[U]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 1 & 0 & 0 \\ -0.333333 & 1 & 0 \\ 0.888889 & -0.133333 & 1 \end{pmatrix}$$

```
Out[]//MatrixForm=
```

$$\begin{pmatrix} 9. & 3. & 1.0 \\ 0 & 5. & 5.33333 \\ 0 & 0 & 6.82222 \end{pmatrix}$$

2.4.2. Метод Дуллітла і Краута

Є декілька методик організації процедури LU-розкладання елементів матриці A. У більшості випадків початкова матриця розгортається у змінних напрямках *«рядок-стовпець»*:

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} & \dots & u_{1n} \\ l_{21} & l_{31} & l_{41} & \dots & l_{n1} \\ u_{22} & u_{23} & u_{24} & \dots & u_{2n} \\ l_{32} & l_{32} & l_{42} & \dots & l_{n2} \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}$$

Тому формули (2.46) використовують у такий спосіб: спочатку перший рядок початкової матриці **A** переноситься без змін як перший рядок до матриці **U**, а елементи першого стовпця матриці **A**, нормовані за значенням діагонального елемента, переносяться в перший стовпець матриці **L**. Потім елементи другого рядка a_{2s} початкової матриці перетворюються в елементи матриці **U** (відніманням отриманих раніше попарних добутків елементів l_{21} і u_{1s}) і елементи другого стовпця a_{s2} — в елементи другого стовпця матриці **L** (відніманням отриманих раніше попарних добутків елементів l_{s1} і u_{12} і ділення результату на u_{22}) і т. д.

Проте відомі й інші підходи. Так, у разі використання *методу Дуллітла* матриця розгортається по рядках у такому порядку:

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} & \dots & u_{1n} \\ l_{21} & & & & \\ u_{22} & u_{23} & u_{24} & \dots & u_{2n} \\ l_{31} & l_{32} & & & \\ u_{33} & u_{34} & u_{35} & \dots & u_{3n} \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}$$

Спочатку перший рядок матриці **A** переноситься як перший рядок до матриці **U**, потім елементи a_{sj} наступних рядків, які стоять до діагональної позиції, перетворюються в елементи матриці **L**, а елементи, що займають діагональні позиції та розташовані праворуч від діагоналі, перетворюються в елементи матриці **U** згідно з формулами (2.46).

У разі застосування *методу Краута* матриця розгортається по стовпцях:

$$\begin{bmatrix} l_{11} & l_{21} & l_{31} & \dots & l_{n1} \\ u_{12} & u_{22} & \dots & & \\ l_{32} & l_{42} & l_{52} & \dots & l_{n2} \\ u_{13} & u_{23} & u_{33} & \dots & \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix}$$

Спочатку елементи першого стовпця початкової матриці, нормовані за значенням діагонального елемента, переносяться в перший стовпець матриці **L**, а потім елементи a_{sj} наступних стовпців, що стоять вище за діагональну позицію і на самій діагоналі початкової матриці, перетворюються в елементи матриці **U**, а елементи, розташовані нижче від діагональної позиції, перетворюються в елементи матриці **L**. При цьому одиничні елементи вибираються не на діагоналі матриці **L** (як це було прийнято раніше), а на діагоналі матриці **U**, що призводить до зміни формул перерахунку значень елементів матриць **L** і **U**. Замість співвідношень (2.42) будуть справедливі такі вирази:

$$\left.\begin{aligned} u_{ij} &= \frac{1}{l_{ii}}\left(a_{ij} - \sum_{k=1}^{i-1} l_{ik}u_{kj}\right), \quad i < j,\ i = 1, 2, \dots, n,\ j = 1, 2, \dots, n, \\ l_{ij} &= a_{ij} - \sum_{k=1}^{j-1} l_{ik}u_{kj}, \qquad\qquad i \geqslant j,\ i = 1, 2, \dots, n,\ j = 1, 2, \dots, n. \end{aligned}\right\} \qquad (2.48)$$

Раніше для однієї й тієї же самої матриці були отримані звичайний LU-розклад у прикладі 2.5 і розклад за Краутом у прикладі 2.2, які звісно, не збігаються.

Розглянуті методи (звичайне LU-перетворення, метод Дуллітла і метод Краута) еквівалентні і відрізняються організацією маршруту обчислень, що позначається за їх програмної реалізації на частоті виклику і порядку зміни підпрограм перерахунку значень елементів початкової матриці, які реалізують співвідношення (2.46) або (2.48). У звичайному LU-перетворенні кожна з таких підпрограм обробляє поперемінно всі елементи рядка або стовпця, що залишилися до даного кроку неперетвореними, а в разі застосування методів Дуллітла і Краута обидві ці підпрограми використовуються для обробки елементів кожного рядка або кожного стовпця. Залежно від конкретної структури матриці і значень її елементів час виконання обчислень може змінюватись.

2.4.3. Метод Холецького

Метод Холецького використовується для розв'язання систем лінійних рівнянь з симетричними додатними матрицями ($a_{ij} = a_{ji}$). У цих випадках приймається, що $u_{kk} = l_{kk}$, $u_{sj} = l_{js}$.

З добутку двох матриць

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ a_{31} & a_{32} & \dots & a_{3n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 & \dots & 0 \\ l_{21} & l_{22} & 0 & \dots & 0 \\ l_{31} & l_{32} & l_{33} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ l_{n1} & l_{n2} & l_{n3} & \dots & l_{nn} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & \dots & \dots & u_{1,n} \\ 0 & u_{22} & u_{23} & \dots & u_{2,n} \\ 0 & 0 & u_{33} & \dots & u_{3,n} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & u_{n,n} \end{bmatrix}$$

за правилами матричного множення знаходимо співвідношення між елементами матриць:

$$a_{11} = l_{11}u_{11} \Rightarrow l_{11} = u_{11} = \sqrt{a_{11}},$$

$$a_{i1} = l_{i1}u_{11} \Rightarrow l_{i1} = \frac{a_{i1}}{l_{11}},$$

$$a_{1i} = l_{11}u_{1i} \Rightarrow u_{1i} = \frac{a_{1i}}{l_{11}}.$$

З формули (2.46) маємо

$$l_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik}u_{kj}}{u_{jj}} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik}l_{jk}}{l_{jj}}, \qquad (2.49)$$

$$a_{ij} = \sum_{k=1}^{j-1} l_{ik}u_{kj} + l_{ij}u_{jj}.$$

Тому перетворення Холецького для симетричних матриць набуває такого вигляду:

$$l_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk}}{l_{jj}}, \quad j = 1, 2, ..., k-1, \tag{2.50}$$

$$l_{jj} = \left(a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2 \right)^{1/2}, \quad j = 1, 2, ..., k. \tag{2.51}$$

Приклад 2.6

Користуючись методом Холецького, розкладемо матрицю

$$\begin{bmatrix} 6 & 15 & 55 \\ 15 & 55 & 225 \\ 55 & 225 & 976 \end{bmatrix}$$

у добуток двох трикутних матриць.

Згідно з формулами (2.50) і (2.51) знаходимо:

$$l_{11} = \sqrt{a_{11}} = \sqrt{6} = 2,4495,$$

$$l_{21} = \frac{a_{21}}{l_{11}} = \frac{15}{2,4495} = 6,1237,$$

$$l_{31} = \frac{a_{31}}{l_{11}} = \frac{55}{2,4495} = 22,4537,$$

$$l_{12} = 0,$$

$$l_{22} = \sqrt{a_{22} - l_{21}^2} = \sqrt{55 - (6,12371)^2} = 4,1833,$$

$$l_{32} = \frac{a_{32} - l_{21} l_{31}}{l_{22}} = \frac{225 - 6,1237(22,4537)}{4,1833} = 20,9165,$$

$$l_{13} = 0. \quad l_{23} = 0,$$

$$l_{33} = \sqrt{a_{33} - l_{31}^2 - l_{32}^2} = \sqrt{976 - (22,4537)^2 - (20,9165)^2} = 5,8595,$$

$$\mathbf{L} = \mathbf{U}^{\mathrm{T}} = \begin{bmatrix} 2,4495 & 0 & 0 \\ 6,1237 & 4,1833 & 0 \\ 22,4537 & 20,9165 & 5,8595 \end{bmatrix}.$$

У складі пакета `LinearAlgebra` системи Mathematica є функція розкладання за методом Холецького, виклик якої здійснюється оператором

```
In[]:= <<LinearAlgebra`Cholesky`
```

Розв'яжемо систему рівнянь методом Холецького. Введемо матрицю системи і вектор правих частин рівнянь:

```
In[]:= A = {{6, 15, 55}, {15, 55, 225}, {55, 225, 976}};
       B = {1, 4, -5};
```

Наведений нижче оператор здійснює розкладання матриці за методом Холецького

```
In[]:= U = CholeskyDecomposition[A]; MatrixForm[U]
Out//MatrixForm =
```

$$\begin{pmatrix} \sqrt{6} & 5\sqrt{\frac{3}{2}} & \frac{55}{\sqrt{6}} \\ 0 & \sqrt{\frac{35}{2}} & 5\sqrt{\frac{35}{2}} \\ 0 & 0 & \sqrt{\frac{103}{2}} \end{pmatrix}$$

Покажемо матрицю **U** з числовими значеннями її елементів:

```
In[]:= MatrixForm[N[U]]
Out//MatrixForm =
```

$$\begin{pmatrix} 2.44949 & 6.12372 & 22.4537 \\ 0 & 4.1833 & 20.9165 \\ 0 & 0 & 5.85947 \end{pmatrix}$$

Перевіримо правильність отриманого розкладу:

```
In[]:= A - Transpose[U] . U
Out[]= {{0, 0, 0}, {0, 0, 0}, {0, 0, 0}}
```

Приклад 2.7

Повторимо, користуючись співвідношеннями (2.46), розв'язання системи лінійних рівнянь з прикладу 2.2, що був виконаний раніше за допомогою послідовності елементарних матриць виключення для матричного варіанта методу Гаусса.

Початковою системою є

$$\begin{bmatrix} 9 & 3 & 1 \\ -3 & 4 & 5 \\ 8 & 2 & 7 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ -4 \\ 5 \end{bmatrix}.$$

Скористаємося LU-розкладом матриці системи рівнянь, отриманим у прикладі 2.5:

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ -0,333333 & 1 & 0 \\ 0,888889 & -0,133333 & 1 \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} 9 & 3 & 1 \\ 0 & 5 & 5,333333 \\ 0 & 0 & 6,822222 \end{bmatrix}.$$

У відповідності з формулами (2.47) запишемо процес розв'язування у пакеті Mathematica:

```
In[]:= L = {{1, 0, 0}, {-0.333333, 1, 0}, {0.888889, -0.133333, 1}};
       U = {{9., 3., 1.}, {0, 5., 5.333333}, {0, 0, 6.822222}};
       b = {6, -4, 5};
       n = Length[L];
In[]:= Do [{summ = 0., Do [{summ = summ + L[[i,s]]*y[s];}, {s, 1, i-1} ],
          y[i] = b[[i]] - summ}, {i, 1, n} ]
In[]:= Y = Array[y, n]
Out[]= {6., -2., -0.6}
```

```
In[]:= Do [{summ = 0., Do [{summ = summ + U[[i,s]]*x[s];}, {s, i+1, n} ],
          x[i] = (Y[[i]] - summ)/U[[i,i]]}, {i, n, 1, -1} ]
In[]:= X = Array[x, n]
Out[]= {0.778502, -0.306189, -0.0879479}
```

Для кращого розуміння процесу розв'язування наведемо його хід у наочній формі:

$$\begin{bmatrix} 1 & 0 & 0 \\ -0,333333 & 1 & 0 \\ 0,888889 & -0,133333 & 1 \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 6 \\ -4 \\ 5 \end{bmatrix}.$$

Звідси випливає

$$y_1 = 6,$$
$$y_2 = -4 + 0,333333 \cdot 4 = -2,$$
$$y_3 = 5 - 0,888889 \cdot 6 + 0,133333 \cdot 2 = -0,6,$$

$$\begin{bmatrix} 9 & 3 & 1 \\ 0 & 5 & 5,333333 \\ 0 & 0 & 6,822222 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ -2 \\ -0,6 \end{bmatrix},$$

$$x_3 = -0,6/6,82222 = -0,087948,$$
$$x_2 = \frac{-2 - 5,33333 \cdot (-0,087948)}{5} = -0,306189,$$
$$x_1 = \frac{6 - (-0,087948) - 3 \cdot (-0,306189)}{9} = 0,778502.$$

2.4.4. Обчислення оберненої матриці на основі LU-розкладу

Знаючи розклад $\mathbf{A} = \mathbf{LU}$, можна обчислити обернену матрицю на основі виразу $\mathbf{A}^{-1} = \mathbf{U}^{-1}\mathbf{L}^{-1}$. Позначимо елементи матриці $\mathbf{L}^{-1}$ через k_{ij}, а елементи матриці $\mathbf{U}^{-1}$ через m_{ij}; обчислимо їх:

$$\mathbf{L}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [\delta_{ij}], \quad \mathbf{U}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [\delta_{ij}], \quad \delta_{ij} = \begin{cases} 1, i = j, \\ 0, i \neq j. \end{cases}$$

$$k_{ij} = \frac{\delta_{ij} - \sum_{k=j}^{i-1} l_{ik}k_{kj}}{l_{ii}}, \quad i = j, ..., n,$$

$$m_{ij} = \frac{\delta_{ij} - \sum_{k=i+1}^{j} u_{ik}k_{kj}}{u_{ii}}, \quad i = j, ..., 1.$$

У відповідності з формулами (2.48) визначаємо послідовно стовпці матриць $\mathbf{L}^{-1}$ і $\mathbf{U}^{-1}$. Можна показати, що ця процедура, сигнальна функція якої $f_A(n) = 2/3\, n^3$, є однією з найефективніших процедур обчислення оберненої матриці.

Приклад 2.8

Знайдемо обернену матрицю для

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 4 & 1 & 2 \end{bmatrix}.$$

Спочатку обчислимо матриці $\mathbf{L}$ і $\mathbf{U}$, користуючись пакетом Mathematica:

```
In[]:= << LinearAlgebra`MatrixManipulation`
In[]:= A = {{1, 2, 3}, {2, 1, 2}, {4, 1, 2}};
In[]:= {lu, p, cn} = LUDecomposition[A]
Out[]= {{{1, 2, 3}, {2, -3, -4}, {4, 7/3, -2/3}}, {1, 2, 3}, 1}
In[]:= {l, u} = LUMatrices[lu]
Out[]= {{{1, 0, 0}, {2, 1, 0}, {4, 7/3, 1}}, {{1, 2, 3}, {0, -3, -4}, {0, 0, -2/3}}}
```

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 4 & 7/3 & 1 \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -3 & -4 \\ 0 & 0 & -2/3 \end{bmatrix}.$$

```
In[]:= l . u    (*перевірка*)
Out[]= {{1, 2, 3}, {2, 1, 2}, {4, 1, 2}}
```

Далі визначимо обернені матриці $\mathtt{LI} = \mathbf{L}^{-1}$ і $\mathtt{UI} = \mathbf{U}^{-1}$:

```
In[]:= LI = Inverse[l]
Out[]= {{1, 0, 0}, {-2, 1, 0}, {2/3, -7/3, 1}}
In[]:= UI = Inverse[u]
Out[]= {{1, 2/3, 1/2}, {0, -1/3, 2}, {0, 0, -3/2}}
```

$$\mathbf{L}^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 2/3 & -7/3 & 1 \end{bmatrix}, \quad \mathbf{U}^{-1} = \begin{bmatrix} 1 & 2/3 & 1/2 \\ 0 & -1/3 & 2 \\ 0 & 0 & -3/2 \end{bmatrix}.$$

```
In[]:= LI . UI   (* перевірка *)
Out[]= {{1, 0, 0}, {0, 1, 0}, {0, 0, 1}}
```

Нарешті обчислимо обернену матрицю $\mathbf{A}^{-1}$:

```
In[]:= AI = UI . LI
Out[]= {{0, -1/2, 1/2}, {2, -5, 2}, {-1, 7/2, -3/2}}
```

$$\mathbf{A}^{-1} = \mathbf{U}^{-1}\mathbf{L}^{-1} = \frac{1}{2}\begin{bmatrix} 0 & -1 & 1 \\ 4 & -10 & 4 \\ -2 & 7 & -3 \end{bmatrix}.$$

2.5. Точність розв'язку систем лінійних рівнянь

2.5.1. Аналіз похибок через число обумовленості матриці A

Нехай $\bar{\mathbf{x}}$ — обчислене значення, $(\mathbf{x} - \bar{\mathbf{x}})$ — похибка розв'язку, $\boldsymbol{\varepsilon} = \mathbf{b} - A\bar{\mathbf{x}}$ — нев'язка розв'язку системи рівнянь $A\mathbf{x} = \mathbf{b}$. Існують системи, для яких нев'язка може бути малою, а похибка — великою. Розглянемо загальний випадок, коли матриця A задана точно, а вектор $\mathbf{b}$ — з деякою абсолютною похибкою $\delta\mathbf{b}$.

$$A(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b},$$
$$\delta\mathbf{x} = A^{-1}\delta\mathbf{b}.$$

Користуючись нормами матриці й вектора, введеними в розділі 1, запишемо:

$$\|\delta\mathbf{x}\| \leqslant \|A^{-1}\|\|\delta\mathbf{b}\|, \quad \|\mathbf{b}\| = \|A\mathbf{x}\| \leqslant \|A\|\|\mathbf{x}\|.$$

Тоді можна знайти залежність відносної похибки норми вектора розв'язку від відносної похибки норми вектора правої частини $\mathbf{b}$. Перемноживши ці дві нерівності, отримуємо

$$\|\mathbf{b}\|\|\delta\mathbf{x}\| \leqslant \|A\|\|A^{-1}\|\|\delta\mathbf{b}\|\|\mathbf{x}\|, \quad \text{звідки} \quad \frac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|} \leqslant \operatorname{cond} A \frac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}. \tag{2.52}$$

У формулі (2.52) $\operatorname{cond} A = \|A\|\|A^{-1}\|$ — число обумовленості матриці A, що дорівнює максимально можливому коефіцієнту підсилення відносної похибки норми правої частини системи.

Тепер припустимо, що вектор $\mathbf{b}$ заданий точно, а матриця A — з похибкою. Знайдемо залежність відносної похибки норми вектора розв'язку від відносної похибки норми матриці коефіцієнтів матриці:

$$\begin{gathered}(A + \delta A)(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b}, \quad A\delta\mathbf{x} + \delta A(\mathbf{x} + \delta\mathbf{x}) = 0, \\ -\delta\mathbf{x} = A^{-1}\delta A(\mathbf{x} + \delta\mathbf{x}), \quad \|\delta\mathbf{x}\| \leqslant \|A^{-1}\|\|\delta A\|\|\mathbf{x} + \delta\mathbf{x}\|, \\ \frac{\|\delta\mathbf{x}\|}{\|\mathbf{x} + \delta\mathbf{x}\|} \leqslant \operatorname{cond} A \frac{\|\delta A\|}{\|A\|}.\end{gathered} \tag{2.53}$$

З урахуванням скінченності розрядної сітки комп'ютера, навіть якщо A і $\mathbf{b}$ відомі точно,

$$|\bar{a}_{ij} - a_{ij}| \leqslant u|a_{ij}|, \quad \max_i \sum_{j=1}^{n} |\bar{a}_{ij} - a_{ij}| \leqslant u \max_i \sum_{j=1}^{n} |a_{ij}|,$$

$$\|\delta A\|_\infty = \|\bar{A} - A\|_\infty \leqslant u\|A\|_\infty, \quad \|\delta\mathbf{b}\|_\infty = \|\bar{\mathbf{b}} - \mathbf{b}\| \leqslant u\|\mathbf{b}\|,$$

$$(A + \delta A)(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b} \Rightarrow A\delta\mathbf{x} + \delta A(\mathbf{x} + \delta\mathbf{x}) = \delta\mathbf{b} \Rightarrow$$

$$\Rightarrow \delta\mathbf{x} = A^{-1}\delta\mathbf{b} - A^{-1}\delta A(\mathbf{x} + \delta\mathbf{x}),$$

$$\|\delta x\| \leqslant \|A^{-1}\|\|\delta b\| + \|A^{-1}\|\|\delta A\|\|x + \delta x\| \Rightarrow$$

$$\Rightarrow \|\delta x\| \leqslant \|A^{-1}\|\|A\|\|x\|\frac{\|\delta b\|}{\|b\|} + \|A^{-1}\|\|A\|\|x + \delta x\|\frac{\|\delta A\|}{\|A\|}, \qquad (2.54)$$

$$\|\delta x\| \leqslant 2u \operatorname{cond} A \, \|x\|,$$

де $u = (1/2)\cdot 10^{1-t}$ — відносна похибка запису мантиси в розрядній сітці (t — розрядність комп'ютера).
Відносна помилка розв'язку мала, якщо мале $\operatorname{cond} A$.

Приклад 2.9

Використовуючи метод Гаусса для розв'язання системи з A і b, наведеними нижче

$$A = \begin{bmatrix} 1,2969 & 0,8648 \\ 0,2161 & 0,1441 \end{bmatrix}, \quad b = \begin{bmatrix} 0,8642 \\ 0,1440 \end{bmatrix},$$

отримаємо замість точного розв'язку $x = [2, -2]^T$ наближений

$$\bar{x} = [0,9981, \ -0,4870]^T,$$

нев'язка якого $\varepsilon = [0,00907829, \ 0,00151271]^T$.
Переконатися в цьому можна, користуючись операторами пакета Mathematica

```
In[]:= A = {{1.2969, 0.8648},
           {0.2161, 0.1441}};
       x = {0.9981, -0.4870};
       b = {0.8642, 0.1440};
       ε = A . x - b
Out[]= {0.00907829, 0.00151271}
```

Це сталося тому, що на результат обчислення істотно впливають помилки округлювання:

$$a_{22}^{(2)} = 0,1441 - 0,8648 \cdot 0,2161/1,2969 = 7,71069 \cdot 10^{-9} \approx 10^{-8}.$$

Для даного прикладу потрібна точність задавання елементів матриці a_{ij} не менша 10^{-8}, тому що система є погано обумовленою. Дійсно, в розглянутому прикладі

$$A^{-1} = \begin{bmatrix} 0,1441 & -0,8648 \\ -0,2161 & 1,2969 \end{bmatrix} \cdot 10^8; \quad \|A^{-1}\| = 1,513 \cdot 10^8; \quad \|A\| = 2,1617;$$

$$\operatorname{cond} A = \|A\|\|A^{-1}\| = 2,1617 \cdot 1,5130 \cdot 10^8 = 3,3 \cdot 10^8.$$

Обумовленість матриці зручно перевіряти засобами пакета Mathematica, в якому передбачені оператори визначення норм прямої `MatrixNorm[A]` і оберненої `InverseMatrixNorm[A]` матриць:

```
In[]:= <<LinearAlgebra`MatrixManipulation`
In[]:= A = {{1.2969, 0.8648},
           {0.2161, 0.1441}};
       c1 = MatrixNorm[A]
Out[]= 2.1617
```

```
In[]:= c2 = InverseMatrixNorm[A]
Out[]= 1.513×10^8
In[]:= condA = c1*c2
Out[]= 3.27065×10^8
```

У пакеті Mathematica існує спеціальний оператор `MatrixConditionNumber[A]`, який істотно спрощує визначення обумовленості матриці:

```
In[]:= <<LinearAlgebra`MatrixManipulation`
In[]:= A = {{1.2969, 0.8648},
           {0.2161, 0.1441}};
       MatrixConditionNumber[A]
Out[]= 3.27065×10^8
```

Покажемо зв'язок числа обумовленості $\operatorname{cond} \mathbf{A}$ з власними значеннями матриці $\mathbf{A}$, для якої проведено згідно з формулою (1.8) перетворення до діагональної форми $\mathbf{A} = \mathbf{Q}_1 \mathbf{D} \mathbf{Q}_2^{\mathrm{T}}$, де $\mathbf{D}$ — діагональна матриця, $\mathbf{Q}_1, \mathbf{Q}_2$ — ортогональні матриці, сформовані з власних векторів матриці. Відомо, що ортогональні матриці не змінюють модуля вектора, тобто його квадратичної норми:

$$\left\| \mathbf{Q}_2^{\mathrm{T}} x \right\|_2^2 = \mathbf{x}^{\mathrm{T}} \mathbf{Q}_2 \mathbf{Q}_2^{\mathrm{T}} \mathbf{x} = \mathbf{x}^{\mathrm{T}} \mathbf{x} = \|\mathbf{x}\|_2^2,$$

тому

$$\|\mathbf{A}\|_2^2 = \frac{\|\mathbf{A}\mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} \frac{\left\|\mathbf{Q}_1 \mathbf{D} \mathbf{Q}_2^{\mathrm{T}} \mathbf{x}\right\|_2^2}{\|\mathbf{x}\|_2^2} = \frac{\|\mathbf{D}\mathbf{x}\|_2^2}{\|\mathbf{x}\|_2^2} = \frac{\sum_{i=1}^{n} (d_i x_i)^2}{\|\mathbf{x}\|_2^2} \leqslant \max_i d_i^2 \frac{\sum_{i=1}^{n} x_i^2}{\|x\|_2^2} \leqslant \max d_i^2,$$

$$\|\mathbf{D}\|_2 \leqslant \max_j d_j = \lambda_{\max} \text{ — максимальне власне значення.}$$

Тобто квадратична норма матриці $\mathbf{A}$ обмежена її найбільшим власним значенням:

$$\|\mathbf{A}\|_2 = \|\mathbf{D}\|_2 \leqslant \lambda_{\max}. \tag{2.55}$$

З огляду на те, що ортогональне перетворення оберненої матриці має вигляд $\mathbf{A}^{-1} = \mathbf{Q}_2 \mathbf{D}^{-1} \mathbf{Q}_1^{\mathrm{T}}$, отримуємо

$$\|\mathbf{D}^{-1}\|_2 \leqslant \max_j (d_j^{-1}) = \frac{1}{\lambda_{\min}}.$$

Це дозволяє записати число обумовленості як

$$\operatorname{cond} \mathbf{A} = \frac{\lambda_{\max}}{\lambda_{\min}}. \tag{2.56}$$

2.5.2. Ітераційне покращення розв'язку

Під час розв'язання погано обумовлених систем рівнянь $\mathbf{Ax} = \mathbf{b}$ виникає нев'язка $\mathbf{A}(\mathbf{x} - \bar{\mathbf{x}})$, що дорівнює

$$\varepsilon_i = b_i - \sum_{i=1}^{n} a_{ik}\bar{x}_k, \quad i = 1, 2, ..., n.$$

Для покращення розв'язку його необхідно обчислювати з подвійною точністю. Тоді можна сформувати додаткове рівняння для визначення поправки вектора розв'язку, яке буде обчислюватися ітераційно:

$$\begin{aligned} \mathbf{x}^{(s+1)} &= \mathbf{x}^{(s)} + \delta\mathbf{x}^{(s)}, \\ \boldsymbol{\varepsilon}^{(s)} &= \mathbf{b} - \mathbf{A}\mathbf{x}^{(s)}, \\ \mathbf{L}(\mathbf{U}\delta\mathbf{x}^{(s)}) &= \boldsymbol{\varepsilon}^{(s)}. \end{aligned} \tag{2.57}$$

За умови $nu\,\mathrm{cond}\,\mathbf{A} \leqslant 0,1$, процес збігається і для числа обумовленості справедлива нерівність

$$\mathrm{cond}\,\mathbf{A} \leqslant \frac{1}{nu}\frac{\|\delta\mathbf{x}^{(1)}\|}{\|\mathbf{x}^{(2)}\|}. \tag{2.58}$$

Сигнальна функція такого уточнення згідно з виразом (2.57) за умови, що матриці $\mathbf{L}$ і $\mathbf{U}$ вже відомі, записується як $f_n(\delta\mathbf{x}) = n^2 + 2 \cdot 1/2\, n^2 = 2n^2$.

Приклад 2.10

Задана система лінійних рівнянь

$$\begin{bmatrix} 0,20000 & 0,16667 & 0,14286 \\ 0,16667 & 0,14286 & 0,12500 \\ 0,14286 & 0,12500 & 0,11111 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1,50953 \\ 0,43453 \\ 0,37897 \end{bmatrix}$$

має точний розв'язок $\mathbf{x} = [1,\ 1,\ 1]^T$ у разі використання для відображення чисел п'яти розрядів $t = 5$.

Виконуючи за допомогою пакета Mathematica LU-розкладання, знаходимо:

```
In[]:= <<LinearAlgebra`MatrixManipulation`
In[]:= A = {{0.2,      1.16667, 0.14286},
           {0.16667, 0.14286, 0.125},
           {0.14286, 0.125,    0.11111}};
       {LU, P, conditionNumber} = LUDecomposition[A]

Out[]= {{{0.2, 0.83335, 0.7143}, {1.16667, -0.829384, 0.85407},
         {0.14286, 0.00594762, 0.00398542}}, {1, 2, 3}, 703.223}
    (*зменшення точності до п'яти розрядів*)
In[]:= LU = SetAccuracy [LU, 5]
Out[]= {{0.2000, 0.8334, 0.7143}, {1.1667, -0.8294, 0.8541}, {0.1429, 0.0059, 0.0040}}
```

Запишемо матриці в звичайній формі, виділивши їх із компактної форми, наведеної вище. Для цього скористаємося функціями пакета Mathematica Upper і Lower.

```
In[]:= Upper[LU_?MatrixQ]:= Transpose[LU]*Table[If[i≤j, 1, 0], {i, Length[LU]}, {j, Length[LU]}]
In[]:= Lower[LU_?MatrixQ]:= Transpose[LU] - Upper[LU] + IdentityMatrix[Length[LU]]
In[]:= U = Upper[LU]; MatrixForm[U]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 0.20000 & 1.16667 & 0.14286 \\ 0 & -0.82938 & 0.00595 \\ 0 & 0 & 0.00399 \end{pmatrix}$$

```
In[]:= L = Lower[LU]; MatrixForm[L]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 1.0000 & 0.\times 10^{-5} & 0.\times 10^{-5} \\ 0.8334 & 1.0000 & 0.\times 10^{-5} \\ 0.7143 & 0.8541 & 1.0000 \end{pmatrix}$$

$$\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ 0{,}8334 & 1 & 0 \\ 0{,}7143 & 0{,}8541 & 1 \end{bmatrix}; \quad \mathbf{U} = \begin{bmatrix} 0{,}2000 & 0{,}1667 & 0{,}1429 \\ 0 & -0{,}8294 & 0{,}0059 \\ 0 & 0 & 0{,}0040 \end{bmatrix}.$$

Проведемо уточнення розв'язку $\varepsilon^{(1)}$, використовуючи $t = 10$ розрядів, і поправки вектора розв'язку:

```
In[]:= b = {1.50953, 0.43453, 0.37897};
In[]:= y = SetAccuracy[LinearSolve[L, b], 5]
Out[]= {1.5095, -0.8234, 0.0040}
In[]:= x = SetAccuracy[LinearSolve[U, y], 5]
Out[]= {0.9855, 1.0001, 1.0186}
In[]:= z0 = ϵ(1) = b - A . x
Out[]= {0.0000707502, 0.0000703716, -0.0000140229}
In[]:= x1 = δx(1) = LinearSolve[A, z0]
Out[]= {0.0145317, -0.000147462, -0.0186445}
```

Результат виведемо на екран, використовуючи 20 розрядів

```
In[]:= x2 = x1 + x, 20
Out[]= {1.0000, 1.0000, 1.0000}
```

Таким чином, після першої ітерації отримаємо:

$$\boldsymbol{\varepsilon}^{(1)} = \begin{bmatrix} -0{,}707502 \cdot 10^{-5} \\ 0{,}703716 \cdot 10^{-5} \\ 0{,}140229 \cdot 10^{-5} \end{bmatrix}, \quad \delta\mathbf{x}^{(1)} = \begin{bmatrix} 0{,}0145317 \\ -0{,}000147462 \\ 0{,}0186445 \end{bmatrix},$$

$$\mathbf{x}_2 = \mathbf{x} + \delta\mathbf{x}^{(1)} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}.$$

Уточнення вектора розв'язку згідно з (2.57) наведено в табл. 2.2.

Таблиця 2.2. Ітераційне уточнення розв'язку

N	$x_1^{(1)}$	$x_2^{(1)}$	$x_3^{(1)}$
1	0,9855	1,0001	1,0186
2	1,0000	1,0000	1,0000

Для оцінювання числа обумовленості заданої системи лінійних рівнянь згідно з формулою (2.58) виконаємо такі оператори:

```
In[]:= g1 = VectorNorm[x2,2]
Out[]= 1.73051
In[]:= g2 = VectorNorm[x1,2]
Out[]= 0.0236392
In[]:= condA = 2*g2*10^5/(3*g1)
Out[]= 909.873
```

Таким чином, наближена оцінка числа обумовленості дорівнює

$$\operatorname{cond} \mathbf{A} = \frac{1}{3 \cdot 1/2 \cdot 10^{-5}} \cdot \frac{0,0236392}{1,73051} \approx 910.$$

2.5.3. Уточнення розв'язку методом діагональної модифікації

Метод діагональної модифікації, що використовується для поліпшення розв'язку, — це метод розв'язання погано обумовлених систем, які можуть утворитися у разі лінеаризації систем нелінійних рівнянь. Попередня упорядкованість системи може буди спотворена на наступному кроці лінеаризації, тому що значення діагональних елементів матриці, які визначаються похідними від нелінійних функцій, змінюються і можуть наближатися до нуля. Упорядкування рівнянь не гарантує, що той самий порядок буде збережено на новому кроці обчислень. Метод діагональної модифікації дозволяє відмовитись від багатократного переупорядкування рівнянь і полягає в корекції матриці, виконанні з її допомогою проміжних обчислень і відновленні дійсного (істинного) розв'язку за проміжними.

Якщо на j-му кроці LU-розкладання чи виключення Гаусса діагональний елемент $a_{kk}^{(j)}$ дуже малий, модифікуємо матрицю

$$\mathbf{A}'_j = \mathbf{A}_j + \mathbf{e}_k g_k \mathbf{e}_k^{\mathrm{T}},$$

де g_k — довільна константа, що вибирається з урахуванням середніх значень елементів матриці, а $\mathbf{e}_k$ — транспонований вектор розмірності n, у якого всі елементи є нулями і лише k-й елемент дорівнює одиниці. Наприклад, для матриці $\mathbf{A}_{4\times4}$, вектор $\mathbf{e}_3 = [0,\ 0,\ 1,\ 0]^{\mathrm{T}}$.

Визначимо первинну матрицю через модифіковану:

$$\mathbf{A}_j = \mathbf{A}_j^{\mathrm{T}}[\mathbf{E} - g_k(\mathbf{A}_j^{\mathrm{T}})^{-1}\mathbf{e}_k\mathbf{e}_k^{\mathrm{T}}].$$

Нехай $\mathbf{z} = (\mathbf{A}_j^{\mathrm{T}})^{-1}\mathbf{e}_k$, звідси

$$\mathbf{A}_j^{\mathrm{T}}\mathbf{z} = \mathbf{e}_k. \tag{2.59}$$

Тоді $\mathbf{x} = \mathbf{A}_j^{-1}\mathbf{b} = [\mathbf{E} - g_k\mathbf{z}\mathbf{e}_k^{\mathrm{T}}]^{-1}(\mathbf{A}_j')^{-1}\mathbf{b} = [\mathbf{E} - g_k\mathbf{z}\mathbf{e}_k^{\mathrm{T}}]^{-1}\mathbf{x}'$, де $\mathbf{x}' = (\mathbf{A}_j')^{-1}\mathbf{b}$ — проміжний розв'язок системи

$$\mathbf{A}_j'\mathbf{x}' = \mathbf{b}. \tag{2.60}$$

Для обчислення оберненої матриці $[\mathbf{E} - g_k\mathbf{z}\mathbf{e}_k^{\mathrm{T}}]^{-1}$ скористаємося формулою Хаусхолдера, яка зв'язує прямі й обернені матриці, що відрізняються елементами рядка:

$$\mathbf{A} = \mathbf{B} + \mathbf{p}\mathbf{q}^{\mathrm{T}}, \quad \mathbf{A}^{-1} = \mathbf{B}^{-1} - \frac{\mathbf{B}^{-1}\mathbf{p}\mathbf{q}^{\mathrm{T}}\mathbf{B}^{-1}}{\mathbf{E} - \mathbf{q}^{\mathrm{T}}\mathbf{B}^{-1}\mathbf{p}}.$$

Використовуючи її, запишемо

$$[\mathbf{E} - g_k\mathbf{z}\mathbf{e}_k^{\mathrm{T}}]^{-1} = \mathbf{E} + \frac{g_k\mathbf{z}\mathbf{e}_k^{\mathrm{T}}}{1 - \mathbf{e}_k^{\mathrm{T}}g_k\mathbf{z}} = \mathbf{E} - \frac{\mathbf{z}\mathbf{e}_k^{\mathrm{T}}}{\mathbf{z}_k - 1/g_k},$$

де $\mathbf{z}_k = \mathbf{e}_k^{\mathrm{T}}\mathbf{z}$.
Тоді

$$\mathbf{x} = \left[\mathbf{E} - \frac{\mathbf{z}\mathbf{e}_k^{\mathrm{T}}}{\mathbf{z}_k - 1/g_k}\right]\mathbf{x}', \quad \mathbf{x} = \mathbf{x}' - \left[\frac{\mathbf{x}_k'}{\mathbf{z}_k - 1/g_k}\right]\mathbf{z}, \tag{2.61}$$

тобто дійсний розв'язок формується об'єднанням двох розв'язків (2.59) і (2.60) систем із модифікованою матрицею.

Приклад 2.11

Задана система лінійних рівнянь із матрицею і вектором правої частини:

$$\mathbf{A} = \begin{bmatrix} 2 & 0 & 1 & 0 \\ 1 & 0 & 2 & 3 \\ 0 & 2 & 4 & 0 \\ 3 & 0 & 0 & 2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 3 \\ 2 \\ 4 \end{bmatrix},$$

у матриці якої діагональний елемент a_{22} дорівнює нулю, тому для розв'язання системи необхідне переупорядкування рівнянь. Щоб запобігти переупорядкуванню, модифікуємо матрицю, вибираючи значення константи $g_k = a_{22} = 3$, і відобразимо це за допомогою операторів пакета Mathematica:

```
In[]:= <<LinearAlgebra`MatrixManipulation`
In[]:= A = {{2, 0, 1, 0}, {1, 0, 2, 3}, {0, 2, 4, 0}, {3, 0, 0, 2}};
In[]:= A[[2, 2]] = 3; A
   (* перевизначення елемента матриці a22 = 3 *)
Out[]= {{2, 0, 1, 0}, {1, 3, 2, 3}, {0, 2, 4, 0}, {3, 0, 0, 2}}
```

Далі скористаємося формулами (2.59), (2.60) і отримаємо два розв'язки x1 та z з модифікованою матрицею і двома різними правими частинами b і b1:

```
In[]:= b = {1, 3, 2, 4};
       b1 = {0, 1, 0, 0};
       x1 = LinearSolve[A, b]
```

$$\text{Out[]= } \left\{-\frac{4}{9},\ -\frac{25}{9},\ \frac{17}{9},\ \frac{8}{3}\right\}$$

```
In[]:= z = LinearSolve[A, b1]
```

$$\text{Out[]= } \left\{\frac{2}{9},\ \frac{8}{9},\ -\frac{4}{9},\ -\frac{1}{3}\right\}$$

```
In[]:= g2 = x1[[2]]/(z[[2]] -1/3)
Out[]= -5
In[]:= x = x1 - g2*z
```

$$\text{Out[]= } \left\{\frac{2}{3},\ \frac{5}{3},\ -\frac{1}{3},\ 1\right\}$$

Таким чином, комбінуючи два отриманих розв'язки x1 та z згідно з (2.61), отримуємо розв'язок погано обумовленої системи:

$$\mathbf{x} = \mathbf{x}' - 5\mathbf{z} = \begin{bmatrix} -4/9 \\ -25/9 \\ 17/9 \\ 8/3 \end{bmatrix} - 5 \cdot \begin{bmatrix} 2/9 \\ 8/9 \\ -4/9 \\ -1/3 \end{bmatrix} = \begin{bmatrix} 2/3 \\ 5/3 \\ -1/3 \\ 1 \end{bmatrix}.$$

Пересвідчитися в достовірності отриманого результату можна за допомогою стандартного оператора пакета Mathematica, призначеного для розв'язання лінійних систем рівнянь, яке передбачає упорядкування рівнянь перед обчисленням і в процесі обчислення:

```
In[]:= A1 = {{2, 0, 1, 0}, {1, 0, 2, 3}, {0, 2, 4, 0}, {3, 0, 0, 2}};
       b = {1, 3, 2, 4};
       LinearSolve [A1, b]
```

$$\text{Out[]=} \left\{\frac{2}{3},\ \frac{5}{3},\ -\frac{1}{3},\ 1\right\}$$

2.6. Розв'язання перевизначених систем лінійних рівнянь

В інженерній практиці часто постає необхідність у розв'язанні систем лінійних рівнянь (розріджених чи не розріджених)

$$\mathbf{A}\mathbf{x} = \mathbf{b},$$

де $\mathbf{A}$ — прямокутна матриця $m \times n$ ($m > n$). Наприклад, це трапляється у разі апроксимації поліномом експериментальних даних, обсяг яких перевищує порядок вибраного полінома.

Така система має безліч розв'язків, але можна вибрати серед них такий, що мінімізує нев'язку розв'язку

$$\boldsymbol{\varepsilon} = \mathbf{b} - \mathbf{A}\mathbf{x}.$$

Можна довести, що коли $\mathbf{x}$ задовольняє умову ортогональності

$$\mathbf{A}^{\mathrm{T}}(\mathbf{b} - \mathbf{A}\mathbf{x}) = \mathbf{A}^{\mathrm{T}}\boldsymbol{\varepsilon} = 0,$$

то для будь-якого іншого розв'язку $\mathbf{y}$ виконується нерівність $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 < \|\mathbf{b} - \mathbf{A}\mathbf{y}\|_2$.

Це можна зробити, скориставшись оцінками квадратичної норми для нев'язок $\boldsymbol{\varepsilon}_x = \mathbf{b} - \mathbf{A}\mathbf{x}$, $\boldsymbol{\varepsilon}_y = \mathbf{b} - \mathbf{A}\mathbf{y}$, записавши

$$\boldsymbol{\varepsilon}_y = (\mathbf{b} - \mathbf{A}\mathbf{x}) + (\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{y}) = \boldsymbol{\varepsilon}_x + \mathbf{A}(\mathbf{x} - \mathbf{y}). \tag{2.62}$$

З урахуванням умови $\mathbf{A}^{\mathrm{T}}\boldsymbol{\varepsilon} = 0$ у разі піднесення рівняння (2.62) до квадрату отримаємо:

$$\begin{aligned} \boldsymbol{\varepsilon}_y^{\mathrm{T}}\boldsymbol{\varepsilon}_y &= \boldsymbol{\varepsilon}_x^{\mathrm{T}}\boldsymbol{\varepsilon}_x + (\mathbf{x} - \mathbf{y})^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{A}(\mathbf{x} - \mathbf{y}), \\ \|\boldsymbol{\varepsilon}_y\|_2^2 &= \|\boldsymbol{\varepsilon}_x\|_2^2 + \|\mathbf{A}(\mathbf{x} - \mathbf{y})\|_2^2 > \|\boldsymbol{\varepsilon}_x\|_2^2. \end{aligned} \tag{2.63}$$

Тому початкове перевизначене рівняння зводиться до так званої нормальної форми множенням правої та лівої його частин на транспортовану матрицю коефіцієнтів цього рівняння:

$$\begin{aligned} (\mathbf{A}^{\mathrm{T}}\mathbf{A})\mathbf{x} &= \mathbf{C}\mathbf{x} = \mathbf{A}^{\mathrm{T}}\mathbf{b}, \\ \mathbf{x} &= (\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}, \end{aligned} \tag{2.64}$$

де $\mathbf{C} = \mathbf{A}^{\mathrm{T}}\mathbf{A}$, розмірності $n \times n$; $c_{ij} = a_i^{\mathrm{T}} a_j$, де a_i, a_j — рядки початкової матриці $\mathbf{A}$.

Матриця $\mathbf{C}$ неособлива, якщо стовпці матриці $\mathbf{A}$ лінійно незалежні. Обчислення добутків $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ і $\mathbf{A}^{\mathrm{T}}\mathbf{b}$ потребує $1/2\,n(n+1)m + nm = 1/2\,nm(n+3)$ операцій. Саме розв'язання виконується за $3/6n$ операцій.

Приклад 2.12

Знайдемо нормальну форму заданого рівняння:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \\ 2 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} 3 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \\ 6 \end{bmatrix}.$$

Із розв'язку нормального рівняння

$$\begin{bmatrix} 3 & -1 & -1 \\ & 8/3 & -4/3 \\ & & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2/3 \\ 6 \end{bmatrix}$$

знаходимо невідомі $x_3 = 3$, $x_2 = 7/4$, $x_1 = 5/4$.
Можна показати, що нев'язка $\varepsilon = 1/4[-1,\ 1,\ 0,\ 2,\ 3,\ -3]^T$ ортогональна стовпцям **A**, тобто $\varepsilon^T \mathbf{a}_j = 0$.
Розв'яжемо систему рівнянь із прикладу 2.12 за допомогою пакета Mathematica:

```
In[]:= A = {{1, 0, 0}, {0, 1, 0}, {0, 0, 1}, {-1, 1, 0}, {0, -1, 1}, {-1, 0, 1}};
       B = Transpose[A]
Out[]= {{1, 0, 0, -1, 0, -1}, {0, 1, 0, 1, -1, 0}, {0, 0, 1, 0, 1, 1}}

In[]:= C = B . A
Out[]= {{3, -1, -1}, {-1, 3, -1}, {-1, -1, 3}}

In[]:= d = {1, 2, 3, 1, 2, 1};
       b = B . d
Out[]= {-1, 1, 6}

In[]:= NSolve[{3x - y - z == -1, -x + 3y - z == 1, -x -y + 3z == 6}, {x, y, z}]
Out[]= {{x → 1.25, → 1.75, z → 3.}}
```

2.7. Розв'язання систем рівнянь із комплексними коефіцієнтами

Щоб запобігти необхідності зміни форми запису комплексного числа (перетворення з арифметичної форми на тригонометричну і навпаки) під час виконання операцій множення/ділення та додавання/віднімання за методом Гаусса чи за методом LU-розкладання для розв'язання системи рівнянь із комплексними коефіцієнтами

$$\mathbf{A}(i\omega)\mathbf{X}(i\omega) = \mathbf{b}(i\omega)$$

або

$$(\mathrm{Re}\,\mathbf{A} + i\,\mathrm{Im}\,\mathbf{A})(\mathrm{Re}\,\mathbf{X} + i\,\mathrm{Im}\,\mathbf{X}) = \mathrm{Re}\,\mathbf{b} + i\,\mathrm{Im}\,\mathbf{b},$$

рекомендується звести цю задачу для кожного значення ω_k, $k = 1, 2, 3, \ldots$ до вже відомої задачі розв'язання лінійних систем з дійсними коефіцієнтами. Проте в цьому разі розмірність задачі збільшиться вдвічі:

$$\begin{bmatrix} \mathrm{Re}\,\mathbf{A} & -\mathrm{Im}\,\mathbf{A} \\ \mathrm{Im}\,\mathbf{A} & \mathrm{Re}\,\mathbf{A} \end{bmatrix} \begin{bmatrix} \mathrm{Re}\,\mathbf{X} \\ \mathrm{Im}\,\mathbf{X} \end{bmatrix} = \begin{bmatrix} \mathrm{Re}\,\mathbf{b} \\ \mathrm{Im}\,\mathbf{b} \end{bmatrix}. \tag{2.65}$$

Приклад 2.13

Розв'яжемо систему лінійних рівнянь з заданою матрицею комплексних коефіцієнтів **A** і вектором правої частини **b** методом Гаусса:

$$\mathbf{A} = \begin{bmatrix} 1-2i & 2+3i \\ 2+i & 3-i \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1+3i \\ 2-5i \end{bmatrix}.$$

Скористаємось виразом (2.65) і, враховуючи, що в нашому прикладі

$$\mathrm{Re}\,\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}, \quad \mathrm{Re}\,\mathbf{b} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad \text{і} \quad \mathrm{Im}\,\mathbf{A} = \begin{bmatrix} -2 & 3 \\ 1 & -1 \end{bmatrix}, \quad \mathrm{Im}\,\mathbf{b} = \begin{bmatrix} 3 \\ -5 \end{bmatrix},$$

сформуємо систему лінійних рівнянь, яку належить розв'язувати:

$$\begin{bmatrix} 1 & 2 & 2 & -3 \\ 2 & 3 & -1 & 1 \\ -2 & 3 & 1 & 2 \\ 1 & -1 & 2 & 3 \end{bmatrix} \begin{bmatrix} \mathrm{Re}\,x_1 \\ \mathrm{Re}\,x_2 \\ \mathrm{Im}\,x_1 \\ \mathrm{Im}\,x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ -5 \end{bmatrix}.$$

Скористаємось оператором NSolve пакета Mathematica для безпосереднього розв'язання цієї системи лінійних рівнянь, змінивши для спрощення запису позначення невідомих: $x = \mathrm{Re}\,x_1$, $y = \mathrm{Re}\,x_2$, $z = \mathrm{Im}\,x_1$, $v = \mathrm{Im}\,x_2$.
Тоді вхідний запис рівнянь у форматі пакета Mathematica і результат розрахунку матиме такий вигляд:

```
In: = NSolve[{x + 2y + 2z - 3v == 1,
              2x + 3y - z + v == 2,
             -2x + 3y + z + 2v == 3,
              x - y + 2z + 3v == - 5},
              {v, x, y, z}]
Out[]= {{v → -0.466667, x → -0.8, y → 1.06667, z → -0.866667}}
```

Тобто розв'язок нашого прикладу

$$x_1 = \mathrm{Re}\,x_1 + i\,\mathrm{Im}\,x_1 = -0{,}8 - i0{,}866667,$$
$$x_2 = \mathrm{Re}\,x_2 + i\,\mathrm{Im}\,x_2 = 1{,}06667 - i0{,}466667.$$

Ту ж систему рівнянь із комплексними коефіцієнтами безпосередньо можна розв'язати за допомогою засобів пакета Mathematica, як то показано в наступному розділі.

Слід відмітити, що оператори пакета Mathematica настільки потужні, що дають змогу розв'язувати системи лінійних рівнянь із комплексними коефіцієнтами. Так, систему рівнянь із прикладу 2.13 можна розв'язати таким чином:

```
In[]:= A = {{1 - 2 I,  2 + 3 I},
           {2 + I,     3 - I}};
       b = {1 + 3 I,  2 -5 I};
       X = LinearSolve[A, b]
Out[]= {-4/5 - 13 I/15, -16/15 - 7 I/15}
```

2.8. Засоби пакета Mathematica, призначені для розв'язання систем лінійних рівнянь

Нагадаємо, що в пакеті Mathematica передбачено декілька можливостей для розв'язання систем лінійних рівнянь. Якщо ми маємо звичайний запис рівнянь, то доцільно користуватися оператором Solve з форматом

Solve[{eqn1, eqn2, ..., eqnr}, {x_1, x_2, ..., x_r}].

Наприклад,

```
In[]:= Solve [{x + 5 y == -4, 2 x + y == 1}, {x, y}]
Out[]= {{x → 1, y → -1}}
```

Якщо система лінійних рівнянь записана в матричній формі $\mathbf{Ax} = \mathbf{b}$, то більш ефективним виявляється оператор LinearSolve [A, b], в якому використовується метод ітераційного покращення розв'язку (2.57). Для попереднього прикладу маємо:

```
In[]:= A = {{1, 5}, {2, 1}};  b = {-4, 1};  v = LinearSolve [A, b]
Out[]= {1, -1}
```

У пакеті Mathematica передбачена можливість автоматизованого переходу від звичайного запису рівнянь до їх запису в матричній формі за допомогою оператора [A,b] = LinearEquationsToMatrices[eqn, x]. Так, для попереднього прикладу запишемо:

```
In[]:= << LinearAlgebra`MatrixManipulation`
In[]:= LinearEquationsToMatrices [{x + 5 y == -4, 2 x + y == 1}, {x, y}]
Out[]= {{{1, 5}, {2, 1}}, {-4, 1}}
```

Якщо визначник матриці системи рівнянь дорівнює нулю, тобто Det[A]=0, отримати розв'язок за допомогою стандартних засобів пакета Mathematica неможливо. Однак, використовуючи оператор RowReduce[A], можна виявити дефект матриці, спростити її, а потім добудувати систему рівнянь. Наприклад, для матриці з двома однаковими рядками:

```
In[]:= A ={{1, 2}, {1, 2}};
       RowReduce[A]
Out[]= {{1, 2}, {0, 0}}
```

Дія операторів Solve і LinearSolve базується головним чином на реалізації методу виключення Гаусса з попереднім упорядкуванням рівнянь. Коли ж виникає необхідність розв'язання системи $\mathbf{Ax} = \mathbf{b}_i$ з різними правими частинами для однієї й тієї ж самої матриці коефіцієнтів **A**, то рекомендується використовувати послідовність двох інших операторів: спочатку LUDecomposition[A], а потім LUBackSubstitution[data, bi] для кожного вектора $\mathbf{b}_i$:

```
In[]:= << LinearAlgebra`MatrixManipulation`
In[]:= A = {{1, 5}, {2, 1}};  lud = LUDecomposition[A]
Out[]= {{1, 5}, {2, -9}}, {1, 2}, 1}

In[]:= v = LUBackSubstitution[lud, b = {-4, 1}]
Out[]= {1, -1}
```

Висновки

1. Розв'язання систем лінійних рівнянь є, по суті, базовою процедурою в пакетах математичного моделювання складних об'єктів і систем, що являють собою основний інструментарій спеціалістів з інформаційних технологій. До розв'язання лінеаризованих систем рівнянь, як буде показано далі, зводяться практично всі задачі визначення динамічних і статичних характеристик складних нелінійних об'єктів і систем, при цьому обчислення лінеаризованих систем рівнянь ітераційно повторюється сотні й тисячі разів із цілеспрямованою зміною деяких параметрів (в задачах статики і динаміки) або з випадковими їх змінами (в задачах статистики). Тому питанням оптимізації базової обчислювальної процедури приділяється така велика увага.

2. Для розв'язання лінеаризованих систем рівнянь найчастіше використовується метод LU-розкладання, який дозволяє на окремих ітераціях зберегти матрицю коефіцієнтів рівнянь і замінювати лише правую частину рівнянь результатами, отриманими на попередній ітерації.

3. Під час розв'язування систем лінійних рівнянь здійснюється їх упорядкування з метою мінімізації похибки обчислень. Таке упорядкування проводиться або попередньо (у випадку лінійної системи), або ж у процесі обчислень, коли елементи матриці коефіцієнтів лінеаризованої системи, що визначаються похідними від нелінійних функцій, змінюються від ітерації до ітерації.

4. Згаданого повторного переупорядкування можна запобігти, скориставшись методом діагональної модифікації, який дозволяє провести поточний «ремонт» матриці коефіцієнтів і знайти розв'язок.

5. Верхньою межею складності алгоритмів прямих методів розв'язання лінеаризованих рівнянь є число n^3, де n – розмірність системи рівнянь. Однак для розріджених систем рівнянь, які розглядаються у наступному розділі, цей показник набагато нижчий, оскільки він пропорційний числу ненульових елементів матриці коефіцієнтів.

6. Через застосування багатопроцесорних ЕОМ існуючі оцінки ефективності окремих чисельних методів, отриманих для однопроцесорних ЕОМ, що широко розповсюджені сьогодні, потребують перегляду. Зокрема, на основі багатопроцесорних ЕОМ успішно можуть використовуватися матричні варіанти методів Гаусса і LU-розкладання, які передбачають здебільшого виконання операції множення матриць.

7. Може статися, що потужний арсенал сучасних чисельних методів буде не в змозі забезпечити необхідну точність під час розв'язування якоїсь конкретної системи лінеаризованих рівнянь через її погану обумовленість. Це, як правило, трапляється внаслідок неправильного формування математичної задачі інженером або вченим (наприклад, через надмірну ідеалізацію компонентів системи, бажання одночасно врахувати властиві їй швидкі й повільні процеси, що відрізняються на багато порядків, тощо).

Контрольні запитання та завдання

1. Знайти коефіцієнти параболи $y = ax^2 + bx + c$, яка проходить через точки (2,8), (1,3), (–2,5).

2. Розв'язати систему лінійних рівнянь із заданою матрицею коефіцієнтів **A** і вектором правої частини **b** методом Гаусса, скориставшись алгоритмом, описаним у підрозділі 2.2:

$$\mathbf{A} = \begin{bmatrix} 0,21 & 2,02 & -10 \\ 0,32 & 6,01 & -25 \\ 0,44 & 8,06 & 10,01 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0,66 \\ 1,85 \\ 2,91 \end{bmatrix}.$$

   Обчислити визначник матриці **A** перемноженням ведучих елементів під час використання методу Гаусса. Повторити обчислення, використовуючи метод обрання головного елемента по стовпцю, і порівняти отримані результати. Підрахувавши нев'язку наближеного розв'язку системи, провести ітераційне її уточнення.

3. Розв'язати систему лінійних рівнянь із заданою матрицею коефіцієнтів **A** і вектором правої частини **b** методом LU-розкладання:

$$\mathbf{A} = \begin{bmatrix} 1 & 3 & -1 \\ 2 & -5 & 1 \\ 3 & -4 & 2 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} -8 \\ 12 \\ 16 \end{bmatrix}.$$

   Знайти визначник матриці **A** та обернену матрицю $\mathbf{A}^{-1}$.

4. Виконати LU-розкладання матриці **A**

$$\mathbf{A} = \begin{bmatrix} 2 & 5 & 8 & -1 \\ -7 & 4 & 9 & -2 \\ 6 & 2 & -1 & 4 \\ 9 & 5 & 3 & -3 \end{bmatrix}$$

   і розв'язати систему рівнянь $\mathbf{Ax} = \mathbf{b}_i$, використовуючи знайдений LU-розклад:

$$\mathbf{b}_1 = \begin{bmatrix} 2 \\ -1 \\ 6 \\ -3 \end{bmatrix}, \quad \mathbf{b}_2 = \begin{bmatrix} 6 \\ -1 \\ 9 \\ -3 \end{bmatrix}, \quad \mathbf{b}_3 = \begin{bmatrix} 0 \\ -7 \\ 4 \\ -3 \end{bmatrix}.$$

5. Знайти точний розв'язок системи $\mathbf{Ax} = \mathbf{b}$ із матрицею Гільберта:

$$\mathbf{A} = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

Знайти наближений розв'язок тієї ж системи $\mathbf{Ax} = \mathbf{b}$, якщо коефіцієнти матриці Гільберта подані приблизно з округленням до чотирьох знаків:

$$\mathbf{A} = \begin{bmatrix} 1,0000 & 0,5000 & 0,3333 & 0,2500 \\ 0,5000 & 0.3333 & 0,2500 & 0,2000 \\ 0,3333 & 0,2500 & 0,2000 & 0,1667 \\ 0,2500 & 0,2000 & 0,1667 & 0,1429 \end{bmatrix}.$$

6. Розв'язати систему лінійних рівнянь

$$\begin{bmatrix} 3 & -0,1 & -0,2 \\ 0,1 & 7 & -0,3 \\ 0,3 & -0,2 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 20 \\ 50 \\ 15 \end{bmatrix}$$

методом LU-розкладання без матричних операцій (за алгоритмом, описаним у підрозділі 2.5.2)

7. Для системи лінійних рівнянь

$$\begin{bmatrix} 3 & -0,1 & -0,2 \\ 0,1 & 7 & -0,3 \\ 0,3 & -0,2 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 20 \\ 50 \\ 15 \end{bmatrix}$$

знайти обернену матрицю і число обумовленості, користуючись алгоритмом, описаним у підрозділі 2.4.5.

8. Довести можливість застосування методу Холецького до симетричної матриці

$$\mathbf{A} = \begin{bmatrix} 6 & 15 & 55 \\ 15 & 55 & 225 \\ 55 & 225 & 979 \end{bmatrix}$$

і виконати LU-розкладання матриці.

9. Розв'язати без переупорядкування лінійну систему

$$\begin{bmatrix} 0 & 1 & 3 \\ 1 & 1 & 5 \\ -2 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 0 \end{bmatrix},$$

скориставшись методом діагональної модифікації під час вибору додаткової сталої $g_{11} = 1$, довести, що проміжні розв'язки $[\mathbf{x}_1]^{\mathrm{T}} = [-1/6, -1/3, 1/2]$ і $[\mathbf{z}]^{\mathrm{T}} = [5/6, 5/3, -1/2]$, а шуканий — $[\mathbf{x}]^{\mathrm{T}} = [-1, -2, 1]$.

10. Розв'язати лінійну систему рівнянь із точністю до трьох значущих розрядів:

$$\begin{bmatrix} 10^{-5} & 10^{-5} & 1 \\ 10^{-5} & -10^{-5} & 1 \\ 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \cdot 10^{-5} \\ -2 \cdot 10^{-5} \\ 1 \end{bmatrix}$$

а) без переупорядкування;

б) з повним переупорядкуванням;

в) з повним переупорядкуванням після зміни масштабу $x_3^1 = 10^5 x_3$.

11. Розв'язати методом LU-розкладання систему лінійних алгебраїчних рівнянь

$$\begin{bmatrix} 10 & 7 & 8 & 7 \\ 7 & 5 & 6 & 5 \\ 8 & 6 & 10 & 9 \\ 7 & 5 & 9 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 4 \\ 3 \\ 3 \\ 1 \end{bmatrix}$$

й знайти число обумовленості $\operatorname{cond} \mathbf{A}$, а також похибку розв'язку $\|\delta\mathbf{x}\|/\|\mathbf{x}\|$, якщо $\|\Delta\mathbf{b}\| \leqslant 0{,}01$.

12. Розв'язати методом LU-розкладання лінійну систему рівнянь із матрицею Ван-дер-Монда четвертого порядку:

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 9 & 16 \\ 1 & 8 & 27 & 64 \\ 1 & 16 & 81 & 256 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 2 \\ 10 \\ 44 \\ 190 \end{bmatrix}$$

і знайти визначник цієї матриці.

Розділ 3

Розв'язання систем лінійних рівнянь великої розмірності

- Кодування розріджених матриць
- Системи зі стрічковими матрицями
- Ітераційні методи
- Умови збіжності ітераційних методів

У цьому розділі розглядаються особливості розв'язання систем лінійних алгебраїчних рівнянь із великими та розрідженими матрицями. Вважатимемо матрицю розрідженою, якщо є сенс скористатися наявністю багатьох нулів [41]. Для особливих матриць, зокрема стрічкових і блочно-діагональних, пропонуються спеціальні методи розв'язання рівнянь: методи прогону та визначальних величин, спрощене LU-розкладання. Особлива увага приділена розв'язанню систем лінійних алгебраїчних рівнянь ітераційними методами.

3.1. Кодування розріджених матриць

Розріджені матриці зустрічаються дуже часто в інженерних задачах, особливо в задачах проектування великих інтегральних схем та задачах моделювання складних об'єктів і оптимального управління ними. В пам'ять комп'ютерів великі розріджені матриці вміщують в упакованому форматі, коли зберігаються лише ненульові елементи матриці разом із необхідною інформацією про їх положення в матриці. Значення всіх ненульових елементів (НЕ) матриці записуються в одновимірному масиві (позначимо його *VA*). Додатково формуються два вказівних масиви — стовпців (*LJ*) та рядків (*LI*). У масиві *LJ* записуються номери стовпців ненульових елементів a_{ij} в тій послідовності, в якій елементи записані в масиві *VA*. В масиві *LI* записуються відносні адреси в масиві *VA* перших ненульових елементів кожного рядка.

Нехай p — кількість ненульових елементів матриці. Тоді розмірності масивів *VA* та *LJ* рівні p. Розмірність масиву *LI* дорівнює $n+1$. Перший елемент масиву *LI* дорівнює одиниці, а останній — $p+1$.

Приклад 3.1

Побудуємо масиви *VA*, *LJ*, *LI* для матриці

$$\begin{bmatrix} a_{11} & 0 & 0 & a_{14} & 0 \\ 0 & a_{22} & 0 & 0 & a_{25} \\ a_{31} & 0 & a_{33} & 0 & 0 \\ a_{41} & a_{42} & 0 & a_{44} & 0 \\ 0 & 0 & a_{53} & 0 & a_{55} \end{bmatrix}.$$

Враховуючи, що в даному випадку $p = 11$ і $n = 5$, будуємо:

VA =	a_{11}	a_{14}	a_{22}	a_{25}	a_{31}	a_{33}	a_{41}	a_{42}	a_{44}	a_{53}	a_{55}

LJ =	1	4	2	5	1	3	1	2	4	3	5

LI =	1	3	5	7	10	12

Застосування кодуючих масивів під час виконання процедур Гаусса чи LU-розкладання створює проблему упорядкування рівнянь системи для збереження розрідженості їх структури. Наскільки це важливо на практиці, свідчить наведений нижче приклад, в якому ненульові елементи позначені хрестиками.

Приклад 3.2

Якщо виконати LU-розкладання матриці системи рівнянь без її упорядкування, то ненульова структура буде втрачена і всі нульові елементи стануть ненульовими:

$$\begin{matrix} a \\ b \\ c \\ d \end{matrix}\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & 0 & 0 \\ \times & 0 & \times & 0 \\ \times & 0 & 0 & \times \end{bmatrix}\cdot\begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix} = \begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}\cdot\begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix} = \begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix}.$$

Перед LU-розкладанням Після LU-розкладання

Переконатися в цьому можна, якщо проаналізувати формулу перетворення елементів матриці $a_{ij}^{(k+1)} = a_{ij}^{(k)} - l_{im}u_{mj}$. Коли $a_{ij}^{(k)} = 0$, з'являється новий ненульовий елемент, якщо одночасно $l_{im} \neq 0$ та $u_{mi} \neq 0$ для $m < j$.

Однак після упорядкування системи рівнянь (простої зміни їх послідовності на зворотну) під час виконання LU-розкладання структура матриці зберігається:

$$\begin{matrix} d \\ c \\ b \\ a \end{matrix}\begin{bmatrix} \times & 0 & 0 & \times \\ 0 & \times & 0 & \times \\ 0 & 0 & \times & \times \\ \times & \times & \times & \times \end{bmatrix}\cdot\begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix} = \begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} \times & 0 & 0 & \times \\ 0 & \times & 0 & \times \\ 0 & 0 & \times & \times \\ \times & \times & \times & \times \end{bmatrix}\cdot\begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix} = \begin{bmatrix} \times \\ \times \\ \times \\ \times \end{bmatrix}.$$

Перед LU-розкладанням Після LU-розкладання

Найбільш відомим алгоритмом упорядкування системи рівнянь із метою збереження нульових елементів матриці або мінімального їх збільшення є алгоритм Марковиця. Згідно з цим алгоритмом під час упорядкування рівнянь новий головний елемент вибирається на перетині тих рядків та стовпців, яким відповідає мінімальна вага w_{ij} і серед яких головний елемент ще не вибирався. При цьому

$$w_{ij} = \min_{i,j}[(r_i - 1)(c_j - 1)], \tag{3.1}$$

де r_i — число НЕ в рядку i, c_j — число НЕ у стовпці j.

Приклад 3.3

Упорядкуємо матрицю, користуючись алгоритмом Марковиця. Нулями в структурі матриці показані десять нових НЕ, які виникають під час LU-розкладання:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} \times & & \times & \times & & \times \\ & \times & \times & \times & \times & \\ \times & \times & \times & 0 & 0 & 0 \\ \times & \times & 0 & \times & 0 & \times \\ & \times & 0 & 0 & \times & 0 \\ \times & & 0 & \times & 0 & \times \end{bmatrix} \end{array}.$$

Спочатку проведемо тільки діагональне упорядкування (вибираємо a_{55}, a_{33}, a_{66}, a_{11}, a_{22}, a_{44}), тобто виконаємо пошук елементів найменшої ваги лише на головній діагоналі:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} 3\times3 & & \times & \times & & \times \\ & 3\times3 & \times & \times & \times & \\ \times & \times & 2\times2 & & & \\ \times & \times & & 3\times3 & & \times \\ & \times & & & 1\times1 & \\ \times & & & \times & & 2\times2 \end{bmatrix} \end{array} \Rightarrow \begin{array}{c} \\ 5 \\ 3 \\ 6 \\ 1 \\ 2 \\ 4 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} 5 & 3 & 6 & 1 & 2 & 4 \end{array} \\ \begin{bmatrix} \times & & & & \times & \\ & \times & & \times & \times & \\ & & \times & \times & & \times \\ & \times & \times & \times & 0 & \times \\ \times & \times & & 0 & \times & \times \\ & & \times & \times & \times & \times \end{bmatrix} \end{array}.$$

Матриця з вагою діагональних елементів — Структура після LU з двома НЕ

Для отримання кращих результатів необхідно застосувати вищенаведену процедуру рекурсивно. Спочатку вибираємо перший головний елемент, враховуючи вагу всіх елементів:

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} 3\times3 & & 2\times3 & 3\times3 & & 2\times3 \\ & 3\times3 & 2\times3 & 3\times3 & 1\times3 & \\ 2\times3 & 2\times3 & 2\times2 & & & \\ 3\times3 & 3\times3 & & 3\times3 & & 2\times3 \\ & 1\times3 & & & 1\times1 & \\ 2\times3 & & & 2\times3 & & 2\times2 \end{bmatrix} \end{array} \quad \begin{array}{c} \\ 5 \\ 1 \\ 2 \\ 3 \\ 4 \\ 6 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} 5 & 1 & 2 & 3 & 4 & 6 \end{array} \\ \begin{bmatrix} \times & & \times & & & \\ & \times & & \times & \times & \times \\ \times & & \times & \times & \times & \\ & \times & \times & \times & & \\ & \times & \times & & \times & \times \\ & \times & & & \times & \times \end{bmatrix} \end{array}.$$

Потім вибираємо другий головний елемент з-серед елементів, де головний елемент ще не вибирався, потім третій і так далі:

$$\begin{array}{c} \\ 5 \\ 1 \\ 2 \\ 3 \\ 4 \\ 6 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} 5 & 2 & 1 & 3 & 4 & 6 \end{array} \\ \begin{bmatrix} \times & \times & & & & \\ \times & \times & & \times & \times & \times \\ & & \times & \times & \times & \\ & \times & \times & \times & & \\ & \times & \times & & \times & \times \\ & & \times & & \times & \times \end{bmatrix} \end{array} \quad \begin{array}{c} \\ 5 \\ 1 \\ 2 \\ 3 \\ 4 \\ 6 \end{array}\!\!\begin{array}{c} \begin{array}{cccccc} 5 & 2 & 3 & 1 & 4 & 6 \end{array} \\ \begin{bmatrix} \times & \times & & & & \\ \times & \times & \times & & & \\ & \times & \times & \times & & \\ & & \times & \times & \times & \times \\ & \times & 0 & \times & \times & \times \\ & & & \times & \times & \times \end{bmatrix} \end{array}.$$

Як результат отримуємо структуру матриці, в якій після LU-розкладання з'являється тільки один НЕ.

Зрозуміло, що наведене упорядкування системи рівнянь із точки зору збереження її ненульової структури не збігається з упорядкуванням, розглянутим раніше, в розділі 2, де основною метою був вибір головних елементів із найбільшими значеннями. Тому для їх погодження на практиці діють таким чином: перевагу віддають алгоритму Марковица, але слідкують, щоб значення елемента з мінімальною вагою, який претендує на місце головного, було не менше, ніж деяке граничне значення, що вибирається з умов забезпечення точності обчислювань. А якщо ці умови не виконуються, тоді розглядається як претендент інший елемент, хай навіть із більшим коефіцієнтом ваги.

Слід відмітити, що складність алгоритму LU-розкладання чи методу Гаусса, за допомогою якого обробляють тільки ненульові елементи матриці рівнянь, зменшується і оцінюється не як раніше ($n^3/3$, де n – розмірність системи рівнянь, що розв'язується), а через лінійну залежність від загальної кількості ненульових елементів матриці **N**.

3.2. Рівняння зі стрічковими матрицями

3.2.1. Спрощене LU-розкладання

Застосування формул LU-розкладання (2.46) – (2.47) істотно спрощується для деяких особливих матриць коефіцієнтів розріджених систем рівнянь [14]. Наприклад, для системи $\mathbf{Ax} = \mathbf{b}$ із тридіагональною матрицею

$$\begin{bmatrix} \alpha_1 & \gamma_1 & & & & \\ \beta_2 & \alpha_2 & \gamma_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & \beta_{n-1} & \alpha_{n-1} & \gamma_{n-1} \\ & & & & \beta_n & \alpha_n \end{bmatrix} =$$

$$= \begin{bmatrix} 1 & & & & \\ \delta_2 & 1 & & & \\ \ddots & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & 1 & \\ & & & \delta_n & 1 \end{bmatrix} \cdot \begin{bmatrix} \sigma_1 & \gamma_1 & & & \\ & \sigma_2 & \gamma_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \\ & & & \sigma_{n-1} & \gamma_{n-1} \\ & & & & \sigma_n \end{bmatrix}$$

одержуємо дводіагональні матриці **L** і **U**, для яких

$$\sigma_1 = \alpha_1, \quad \delta_k = \frac{\beta_k}{\alpha_{k-1}}, \quad \sigma_k = \alpha_k - \delta_k \gamma_{k-1}, \quad k = 2, 3, \ldots, n. \tag{3.2}$$

Такі матриці утворюються під час розв'язування крайових задач, в яких використовують різницеві апроксимації перших і других похідних (див. розділ 12).

Неважко помітити, що вирази (2.47) із урахуванням формул (3.2) дуже спрощуються:

$$y_1 = b_1, \quad y_i = b_i - \delta_i y_{i-1}, \quad i = 2, ..., n, \tag{3.3}$$

$$x_n = y_n, \quad x_i = \frac{1}{\sigma_{ii}}(y_i - \gamma_i x_{i+1}), \quad i = n-1, ..., 1, \tag{3.4}$$

де b_i — компоненти вектора правої частини системи.

Обчислимо загальну кількість операцій, необхідних для розв'язання системи рівнянь із тридіагональною матрицею. Формули (3.2) LU-розкладання містять всього три арифметичні операції (дві з яких операції множення/ділення), а формули (3.3), (3.4) — п'ять (три з яких — операції множення/ділення), що припадають на кожний крок обчислення однієї невідомої x_i. Отже, на відміну від методу Гаусса, сигнальна функція складності розглянутого алгоритму лінійно залежить від розмірності системи рівнянь n.

Розглянутий вище підхід (3.3) і (3.4) можна поширити і на матрицю **A** з п'ятьма діагоналями, яка розкладається на добуток двох тридіагональних матриць у такий спосіб:

$$\begin{bmatrix} \alpha_1 & \beta_1 & \gamma_1 & & & & & \\ \delta_2 & \alpha_2 & \beta_2 & \gamma_2 & & & & \\ \sigma_3 & \delta_3 & \alpha_3 & \beta_3 & \gamma_3 & & & \\ & \ddots & \ddots & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \ddots & \gamma_{n-2} & \\ & & & \ddots & \ddots & \ddots & \beta_{n-1} & \\ & & & & \sigma_n & \delta_n & \alpha_n & \end{bmatrix} =$$

$$= \begin{bmatrix} 1 & & & & & & & \\ \eta_2 & 1 & & & & & & \\ \mu_3 & \eta_3 & 1 & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & \ddots & \ddots & \ddots & & & \\ & & & \ddots & \ddots & 1 & & \\ & & & & \mu_n & \eta_n & 1 & \end{bmatrix} \begin{bmatrix} \upsilon_1 & \nu_1 & \gamma_1 & & & & & \\ & \upsilon_2 & \nu_2 & \gamma_2 & & & & \\ & & \upsilon_3 & \nu_3 & \gamma_3 & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & \ddots & \ddots & \gamma_{n-2} & \\ & & & & & \ddots & \nu_{n-1} & \\ & & & & & & \upsilon_n & \end{bmatrix}.$$

Елементи діагоналей $\sigma_i, \delta_i, \alpha_i, \beta_i$ (елементи діагоналі γ_i не змінюються) перераховуються в діагональні елементи μ_i, η_i матриці **L** і υ_i, ν_i матриці **U** за рекурентними формулами, які базуються на загальних співвідношеннях (2.46) LU-розкладання:

$$\begin{aligned} \mu_i &= \frac{\sigma_i}{\upsilon_{i-2}}, & i &= 3, ..., n, \\ \eta_i &= \frac{\delta_i - \mu_i \nu_{i-2}}{\upsilon_{i-1}}, & i &= 2, ..., n, \\ \nu_i &= \delta_i - \mu_i \gamma_{i-1}, & i &= 1, ..., n-1, \\ \upsilon_i &= \alpha_i - \eta_i \nu_{i-1}, & i &= 1, ..., n. \end{aligned} \tag{3.5}$$

Як випливає з формул (3.5), для виконання LU-розкладання необхідно лише $6n$ операцій множення/ділення, де n — розмірність розв'язуваної розрідженої системи рівнянь.

Далі за відомими матрицями **L** і **U** знаходять розв'язок системи лінійних рівнянь аналогічно до формул (2.47):

$$\mathbf{Ly} = \mathbf{b} \Rightarrow y_i = b_i - \gamma_i y_{i-1} - \sigma_i y_{i-2}, \quad i = 1, ..., n, \tag{3.6}$$

$$\mathbf{Ux} = \mathbf{y} \Rightarrow x_i = \frac{y_i - \beta_i x_{i+1} - c_i x_{i+2}}{\alpha_i}, \quad i = n, ..., 1. \tag{3.7}$$

Отже, для знаходження розв'язку необхідно виконати $5n$ операцій множення/ділення. Тому система лінійних рівнянь з п'ятидіагональною матрицею розв'язується виконанням $11n$ операцій множення/ділення.

Це доцільніше, ніж розв'язувати системи рівнянь ітераційним методом верхньої релаксації (3.25), який вважається одним з найефективніших методів розв'язання дуже великих за розміром систем (розглядається в підрозділі 3.4).

3.2.2. Метод прогону

Для розв'язання систем лінійних рівнянь зі стрічковими матрицями ефективний також *метод прогону*. Розглянемо його застосування на прикладі тієї ж самої системи з тридіагональною матрицею, яка розв'язувалась раніше LU-розкладанням. Запишемо систему в такому вигляді:

$$\begin{cases} \alpha_1 x_1 + \gamma_1 x_2 = b_1, \\ \beta_2 x_1 + \alpha_2 x_2 + \gamma_2 x_3 = b_2, \\ \beta_3 x_2 + \alpha_3 x_3 + \gamma_3 x_4 = b_3, \\ \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \\ \beta_n x_{n-1} + \alpha_n x_n = b_n. \end{cases} \tag{3.8}$$

Розв'яжемо систему (3.8), виконавши спочатку процедуру, аналогічну прямому ходу методу Гаусса:

$$\begin{cases} x_1 - w_1 x_2 = v_1 \Rightarrow x_1 - \left(-\dfrac{\gamma_1}{\alpha_1}\right) x_2 = \dfrac{b_1}{\alpha_1}, \\ x_2 - w_2 x_3 = v_2 \Rightarrow x_2 - \left(-\dfrac{\gamma_2}{\alpha_2 + \beta_2 w_1}\right) x_3 = \dfrac{b_2 - \beta_2 v_1}{\alpha_2 + \beta_2 w_1}, \\ x_3 - w_3 x_4 = v_3 \Rightarrow x_3 - \left(-\dfrac{\gamma_3}{\alpha_3 + \beta_3 w_2}\right) x_4 = \dfrac{b_3 - \beta_3 v_2}{\alpha_3 + \beta_3 w_2}, \\ \ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \\ x_n = v_n, \end{cases} \tag{3.9}$$

де коефіцієнти визначаються за такими рекурентними співвідношеннями:

$$w_i = -\frac{\gamma_i}{\beta_i w_{i-1} + \alpha_i}, \quad i = 2, 3, ..., n,$$
$$v_i = \frac{b_i - \beta_i v_{i-1}}{\beta_i w_{i-1} + \alpha_i}, \quad i = 2, 3, ..., n. \tag{3.10}$$

Обчислення прогонних коефіцієнтів за формулами (3.10) з урахуванням початкових значень $w_1 = (-\gamma_1/\alpha_1)$, $v_1 = b_1/\alpha_1$ називають *прямим ходом* методу прогону. Після цього згідно з (3.9) знаходять значення невідомих:

$$x_n = v_n, \quad x_i = w_i x_{i+1} + v_i, \quad i = n-1, ..., 1. \tag{3.11}$$

Обчислення за формулами (3.11) називають *зворотним ходом* методу прогону. Основний час обчислень витрачається на визначення прогонних коефіцієнтів за формулами (3.10). Для цього потрібно виконати вісім операцій для обчислення кожної пари коефіцієнтів, з яких тільки п'ять є операціями множення/ділення.

Приклад 3.4

Методом прогону розв'яжемо систему лінійних рівнянь з тридіагональною матрицею:

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 \\ 1 & 3 & 1 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 1 & 5 & 1 \\ 0 & 0 & 0 & 1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 2 \\ -3 \\ 2 \end{bmatrix}.$$

Знайдемо прогонні коефіцієнти, скориставшись формулами (3.10) і врахувавши значення елементів діагоналей матриці — головної $\alpha = [2, 3, 4, 5, 3]$, верхньої $\gamma = [1, 1, 1, 1, 0]$, нижньої $\beta = [0, 1, 1, 1, 1]$, — а також вектор правої частини $\mathbf{b} = [1, -1, 2, -3, 2]^T$.

```
In[]:= α = {2, 3, 4, 5, 3}; β = {0, 1, 1, 1, 1};
       γ = {1, 1, 1, 1, 0}; b = {1, -1, 2, -3, 2};
    (* Визначення коефіцієнтів прогону *)
       Do [w[i] = -γ[[i]]/(β[[i]]*w[i-1] + α[[i]]);
          v[i] = (b[[i]] - β[[i]]*v[i-1])/(β[[i]]*w[i-1] + α[[i]]), {i, Length[α]} ]
       Array[w, Length[α]]
Out[]= {1/2, -2/5, -5/18, -18/85, 0}

In[]:=Array[v, Length[α]]
Out[]= {1/2, -3/5, 13/18, -67/85, 1}

    (* Визначення значень невідомих *)
In[]:= Do [x[i] = w[i]*x[i+1] + v[i], {i, Length[α], 1, -1} ]
       Array[x, Length[α]]
Out[]= {1, -1, 1, -1, 1}
```

3.3. Метод визначальних величин

Сучасному інженеру під час розв'язування багатьох практичних задач доводиться мати справу з системами лінійних рівнянь, розмірність яких складає десятки і сотні тисяч. Збільшення розмірності систем лінійних (або лінеаризованих) рівнянь, призводить до зниження ефективності прямих методів, навіть коли мова йде про кодовані розріджені матриці.

У цьому випадку доцільно застосовувати підходи *діакоптики*, тобто розв'язувати великі системи рівнянь по частинах. Якщо система лінійних рівнянь дуже великої розмірності ($n \times n$) має розріджену матрицю коефіцієнтів, то цю матрицю можна звести до блочно-діагональної з обрамленням і сформувати допоміжну систему рівнянь значно меншої розмірності ($m \times m$) для обчислення вектора $\mathbf{X}_2$ так званих *визначальних* величин, або змінних зв'язку (рис. 3.1).

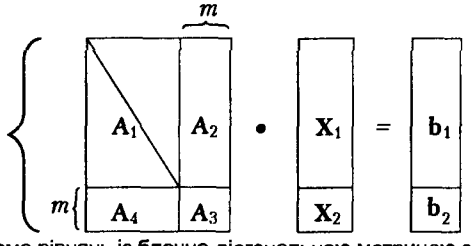


Рис. 3.1. Система рівнянь із блочно-діагональною матрицею з обрамленням

$$\mathbf{A}^{*}\mathbf{X}_2 = \mathbf{b}^{*}. \qquad (3.12)$$

Наведемо алгоритм логічного сортування для пошуку визначальних величин.

1. Визначити рядок вихідної матриці з мінімальним числом змінних k.
2. Якщо таких рядків декілька, то вибрати рядок, число ненульових елементів якого у стовпцях, що визначаються k змінними, максимальне. При цьому $(k-1)$ змінні належатимуть до визначальних величин, а k-та змінна знаходитиметься з відповідного рівняння.

   Вибрані k-ті змінні виключити з подальшого розгляду, перейти до наступного рядка.

Приклад 3.5

Зведемо задану матрицю до блочно-діагональної форми з обрамленням:

	1	2	3	4	5	6	7	
1	×	×	×					(3) – кількість НЕ
2	×	×			×			(3)
3	×		×	×	×			(4)
4			×	×	×	×		(4)
5			×	×	×			(3)
6	×			×		×		(3)
7					×		×	(2)

Побудуємо допоміжну таблицю, де у стовпцях будемо відмічати нові номери рядків $N_н$ (рівнянь), їх старі номери $N_с$, складові вектора $\mathbf{X}_1$ і складові вектора змінних зв'язку $\mathbf{X}_2$.

Починаючи з сьомого рядка, який містить найменшу кількість ненульових елементів, заповнимо таблицю:

$N_н$	$N_с$	$\mathbf{X}_1$	$\mathbf{X}_2$
1	7	7	5
2	2	1	2
3	1	3	
4	5	4	
5	6	6	
6	4		
7	3		

Після упорядкування згідно з послідовністю компонентів векторів $\mathbf{X}_1$ та $\mathbf{X}_2$ отримаємо упорядковану систему рівнянь із обрамленням, в яку ввійшли друга і п'ята змінні:

	7	1	3	4	6	2	5
7	×						
2		×				×	
1		×	×			×	
5			×	×			×
6		×		×	×		
4			×	×	×		×
3		×	×	×			×

Для формування допоміжної системи рівнянь розмірності $m \times m$ із метою знаходження вектора $\mathbf{X}_2$ визначальних величин запишемо матричне рівняння $\mathbf{A} = \hat{\mathbf{A}} + \mathbf{CD} + \mathbf{D}^{\mathrm{T}}\mathbf{D}$, виділивши в структурі отриманої блочно-діагональної матриці з обрамленням складові, показані на рис. 3.2.

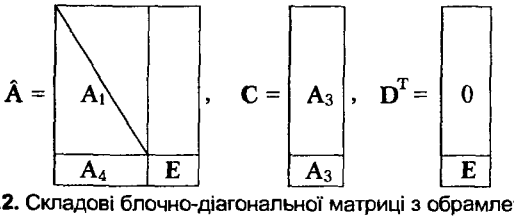


Рис. 3.2. Складові блочно-діагональної матриці з обрамленням

Тепер, використовуючи трикутну матрицю $\hat{\mathbf{A}}$, $m+1$ разів (m — розмірність вектора $\mathbf{X}_2$) розв'язуємо систему

$$\hat{\mathbf{A}}\mathbf{X}^j = \mathbf{b}^j, \quad j = 0, 1, ..., m, \tag{3.13}$$

де $(\mathbf{b}^j)_{j=0} = \mathbf{b}$, $(\mathbf{b}^j)_{j=1,\,m} = \mathbf{C}^j$ (j-й стовпець матриці $\mathbf{C}$); $\mathbf{b}^j = \mathbf{b}$, для $j = 0$, $\mathbf{b}^j = \mathbf{C}_j$, для $j = 1, ..., m$ ($\mathbf{C}^j$ — j-й стовпець матриці $\mathbf{C}$).

Матриця $\mathbf{A}^*$ і вектор $\mathbf{b}^*$ правої частини рівняння (3.12) набирають по стовпцям із отриманих розв'язків рівняння (3.13), використовуючи матрицю-маску **D**:

$$\mathbf{A}^* = [\mathbf{DX}^{(1)}, \mathbf{DX}^{(2)}, ..., \mathbf{DX}^{(m)}] = [\mathbf{X}_2^{(1)}, ..., \mathbf{X}_2^{(m)}],$$
$$\mathbf{b}^* = \mathbf{DX}^{(0)} = \mathbf{X}_2^{(0)}. \tag{3.14}$$

Слід відмітити, що обчислення за формулою (3.14) можна виконувати *паралельно* на багатопроцесорних комп'ютерах, що і реалізовано на практиці в потужних програмах моделювання інтегральних схем.

Після того як буде розв'язано рівняння (3.12) і знайдено вектор визначальних величин $\mathbf{X}_2$, змінні вектора $\mathbf{X}_1$ обчислюються з рівняння, матриця якого має блочно-діагональну форму з обрамленням:

$$X_{1i} = b_i - \sum_{j=1}^{i-1} A_{1,ij} X_j - \sum_{r=1}^{m} A_{2,ir} X_{2r}, \tag{3.15}$$

де $A_{1\,ij}$, $A_{2\,ij}$ елементи матриць $\mathbf{A}_1$ і $\mathbf{A}_2$ відповідно.

Приклад 3.6

Користуючись методом визначальних величин, розв'яжемо систему рівнянь:

$$\begin{bmatrix} 1 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 & 0 \\ 1 & 0 & 2 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 2 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 0 \\ 3 \\ 0 \\ 1 \end{bmatrix}.$$

Після упорядкування зведемо систему до блочно-діагональної форми з обрамленням:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 2 \\ 0 & 2 & 0 & 0 & 0 & 1 \\ 0 & 2 & 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 3 & 0 & 0 & 2 & 0 \\ 0 & 0 & 4 & 2 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_4 \\ x_6 \\ x_5 \\ x_3 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 3 \\ 0 \\ 0 \\ 1 \end{bmatrix} \Rightarrow \quad \mathbf{AX} = \mathbf{b}_0.$$

Визначальною змінною буде тільки змінна x_2, тому систему з трикутною матрицею

$$\mathbf{A}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 3 & 0 & 0 & 2 & 0 \\ 0 & 0 & 4 & 2 & 0 & 1 \end{bmatrix}$$

необхідно розв'язати двічі, використовуючи стовпчик $\mathbf{c} = [2, 1, 0, 0, 0, 1]^T$ матриці $\mathbf{C}$ і вихідний вектор правої частини $\mathbf{b}_0 = [2, 1, 3, 0, 0, 1]^T$. Виконаємо ці розрахунки в пакеті Mathematica:

```
In[]:= A = {{1, 0, 0, 0, 0, 2}, {0, 2, 0, 0, 0, 1}, {0, 2, 2, 0, 0, 0},
          {1, 0, 0, 1, 0, 0}, {1, 3, 0, 0, 2, 0}, {0, 0, 4, 2, 0, 1}};
In[]:= A2 = TakeColumns[A, 5];
       A3 = TakeColumns[IdentityMatrix[6], {6}];
       Do [Q[i] = Append[A2[[i]], A3[[i,1]]], {i, Length[A]} ];
       A1 = Table[Q[i], {i, Length[A]}]; MatrixForm[A1]
```

$$\begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 3 & 0 & 0 & 2 & 0 \\ 0 & 0 & 4 & 2 & 0 & 1 \end{vmatrix}$$

```
In[]:= b0 = {2, 1, 3, 0, 0, 1};
       c = A[[All, 6]]
Out[]= {2, 1, 0, 0, 0, 1}

In[]:= Do [x[i] = (b0[[i]] - Sum[A1[[i,j]]*x[j], {j, i-1}])/A1[[i,i]], {i, Length[A]} ]
In[]:= X = Array[x, 6]
```

$$\text{Out[]}= \left\{2, \frac{1}{2}, 1, -2, -\frac{7}{4}, 1\right\}$$

```
In[]:= Do [x1[i] = (c[[i]] - Sum[A1[[i,j]]*x1[j], {j, i-1}])/A1[[i,i]], {i, 6}]
In[]:= X1 = Array[x1, 6]
```

$$\text{Out=} \left\{2, \frac{1}{2}, -\frac{1}{2}, -2, -\frac{7}{4}, 7\right\}$$

Допоміжна система рівнянь для визначальних величин (3.12) містить у нашому випадку тільки одне рівняння $A^* x_2 = b^*$, коефіцієнти якого згідно з формулою (3.14) знаходять так:

```
In[]: = b* = X[[6]]
Out[]= 1

In[]:= A* = X1[[6]]
Out[]= 7
```

Тобто отримаємо рівняння $1 \cdot x_2 = 7$, яке не складно розв'язати:

```
In[]:= x2 = b*/A*
Out[]= 1/7
```

Інші змінні згідно з виразом (3.15) обчислюються за такими формулами:

$$x_1 = 2 - 2x_2 = 12/7, \quad x_4 = \frac{1 - x_2}{2} = 3/7, \quad x_6 = \frac{3 - 2x_4}{2} = 15/14,$$

$$x_5 = -x_1 = -12/7, \quad x_3 = \frac{-x_1 - 3x_4}{2} = -3/2.$$

Перевіримо результат за допомогою пакета Mathematica:

```
In[]:= X = {12/7, 1/7, -3/2, 3/7, -12/7, 15/14};  A . X - b0
Out[]= {0, 0, 0, 0, 0, 0}
```

Як бачимо, значення окремих складових вектора розв'язку збігаються.

3.4. Метод простої ітерації

Для великих розріджених систем рівнянь досить добрі результати можна отримати з використанням методу визначальних величин. Для рівнянь із заповненими матрицями коефіцієнтів застосовують *ітераційні* методи, які покращують початково обраний вектор розв'язку без обробки матриці коефіцієнтів (тобто зведення її до трикутної форми чи розкладання на добуток двох трикутних матриць), у результаті чого зберігається ненульова структура матриці і не з'являються нові ненульові елементи.

Оскільки ітераційність є однією з головних властивостей чисельних методів (розділ 1), розглянемо більш детально ітераційний метод розв'язання системи лінійних рівнянь типу $\mathbf{Ax} = \mathbf{b}$.

Для розв'язання такої системи рівнянь обирають деяке початкове наближення $\mathbf{x}^{(0)}$, що використовується для обчислення наступних наближень $\mathbf{x}^{(k)}$ за деякою рекурентною формулою обчислень:

$$\mathbf{x}^{(k+1)} = \mathbf{M}\mathbf{x}^{(k)} + \mathbf{C}^{(k)}, \tag{3.16}$$

де k — номер ітерації, $\mathbf{M}$, $\mathbf{C}^{(k)}$ — ітераційні оператори.

У виразі (3.16) під час обчислення $\mathbf{x}^{(k+1)}$ використовується значення наближення тільки на одній попередній ітерації $\mathbf{x}^{(k)}$. Тому цей метод називають *однокроковим* чи *одношаровим*. Але існують і більш складні рекурентні формули наближень, в яких, наприклад для обчислення $\mathbf{x}^{(k+1)}$, використовують значення на двох попередніх ітераціях $\mathbf{x}^{(k)}$ і $\mathbf{x}^{(k-1)}$, тоді їх називають *двокроковими* або *двошаровими*.

У разі розв'язання системи ітераційними методами звичайно задають деяку похибку наближення $\varepsilon > 0$ і припиняють обчислення, якщо виконується умова

$$\left\|\mathbf{x}^{(k+1)} - \xi\right\| \leqslant \varepsilon \left\|\mathbf{x}^{(0)} - \xi\right\|, \tag{3.17}$$

де ξ — точний розв'язок системи лінійних рівнянь.

На практиці використовувати умову (3.17) для завершення обчислень неможливо, тому що значення розв'язку ξ невідоме. (Замість ξ зазвичай використовують $\mathbf{x}^{(k)}$.)

Найпростіша рекурентна формула наближень типу (3.16), яка відповідає *методу простої ітерації*, будується на основі використання нев'язки системи лінійних рівнянь, що розв'язуються:

$$\mathbf{r}^{(k)} = \mathbf{A}\mathbf{x}^{(k)} - \mathbf{b}. \tag{3.18}$$

Для цього методу формула наближень набуває вигляду

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{r}^{(k)} = \mathbf{x}^{(k)} - \mathbf{A}\mathbf{x}^{(k)} + \mathbf{b} = (\mathbf{E} - \mathbf{A})\mathbf{x}^{(k)} + \mathbf{b}. \tag{3.19}$$

Порівнюючи вирази (3.16) і (3.19), пересвідчуємось, що для методу простої ітерації маємо

$$\mathbf{M} = \mathbf{E} - \mathbf{A} \quad \text{і} \quad \mathbf{C}^{(k)} = \mathbf{b}.$$

Тобто початковим наближенням $\mathbf{x}^{(0)}$ у цьому методі можна вважати вектор правої частини системи рівнянь $\mathbf{b}$.

Розглянемо умову збіжності обчислень ітераційними методами за рекурентною формулою (3.16). Запишемо вираз для глобальної похибки (3.17) на двох послідовних ітераціях:

$$\mathbf{x}^{(k+1)} - \xi = \mathbf{M}(\mathbf{x}^{(k)} - \xi) = \ldots = \mathbf{M}^{k+1}(\mathbf{x}^{(0)} - \xi),$$

де, як і раніше, ξ — точний розв'язок системи лінійних рівнянь.

Оскільки $\xi = \mathbf{M}\xi + \mathbf{C}^{(k)}$ і $x^{(k+1)} = \mathbf{M}\mathbf{x}^{(k)} + \mathbf{C}^{(k)}$, то глобальні похибки на двох сусідніх ітераціях зв'язані лінійною залежністю:

(ð'

$$\mathbf{x}^{(k+1)} - \xi = \mathbf{M}(\mathbf{x}^{(k)} - \xi)$$

або

$$\left\|\mathbf{x}^{(k+1)} - \xi\right\| \leqslant \|\mathbf{M}\|\left\|\mathbf{x}^{(k)} - \xi\right\|. \tag{3.20}$$

Для того, щоб процес обчислень збігався (або глобальна похибка за абсолютною величиною зменшувалася), необхідно, щоб норма матриці $\mathbf{M}$ була меншою за одиницю:

$$\|\mathbf{M}\| < 1. \tag{3.21}$$

Під час обчислень оцінювання глобальної похибки $\left\|\mathbf{x}^{(k+1)} - \xi\right\|$ зручно виконувати за значеннями *локальної* похибки, тобто за значеннями, які отримані на двох сусідніх ітераціях. Для цього перепишемо формули (3.20), додавши до правої частини і віднявши від неї значення $\mathbf{x}^{(k+1)}$:

$$\begin{aligned}\left\|\mathbf{x}^{(k+1)} - \xi\right\| &\leqslant \|\mathbf{M}\|\left\|\mathbf{x}^{(k)} - \xi + \mathbf{x}^{(k+1)} - \mathbf{x}^{(k+1)}\right\| = \\ &= \|\mathbf{M}\|\left\|\mathbf{x}^{(k+1)} - \xi\right\| + \|\mathbf{M}\|\left\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right\|,\end{aligned} \tag{3.22}$$

звідки отримуємо:

$$\left\|\mathbf{x}^{(k+1)} - \xi\right\| \leqslant \frac{\|\mathbf{M}\|}{1 - \|\mathbf{M}\|}\left\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\right\| \tag{3.23}$$

або

$$\left\|\mathbf{x}^{(k+1)} - \xi\right\| \leqslant \frac{\|\mathbf{M}\|^{(k+1)}}{1 - \|\mathbf{M}\|}\left\|\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right\|. \tag{3.24}$$

Нагадаємо, що для методу простої ітерації відповідно до (3.16) і (3.19) $\mathbf{M} = \mathbf{E} - \mathbf{A}$, тому накладання умови збіжності обчислень $\|\mathbf{M}\| < 1$ обмежує використання цього методу для розв'язання багатьох практичних математичних задач. Проте умови збіжності можна покращити, якщо у формулу (3.18) ввести коефіцієнт демпфірування $\dot{\omega} \leqslant 1$ для врахування нев'язки:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \dot{\omega}\mathbf{r}^{(k)}. \tag{3.25}$$

Тоді метод простої ітерації перетворюється на *метод верхньої релаксації* (якщо вибрати $1 < \dot{\omega} < 2$ для прискорення збіжності), який застосовується для розв'язання систем лінійних рівнянь великої розмірності, які, зокрема, утворюються під час розв'язування диференціальних рівнянь із частинними похідними у разі використання методу кінцевих різниць (розділ 12). Якщо коефіцієнт демпфірування $\dot{\omega}$ не залежить від номера ітерації, ітераційний метод називається *стаціонарним*.

Існує ще один різновид умови збіжності обчислень ітераційними методами, який базується на використанні оцінки власних значень матриці перетворення **M**, про це вже йшлося у підрозділі 1.4. Згідно з цією умовою для збіжності обчислень необхідно і достатньо, щоб

$$\max_i |\lambda_i(\mathbf{M})| = \lambda_{\max} < 1, \tag{3.26}$$

тобто всі власні значення цієї матриці за модулем мають бути меншими за одиницю.

Доказ умови (3.26) базується на використанні ортогонального координатного базису, який утворюють власні вектори матриці **M**, для розкладання вектора початкової глобальної похибки:

$$\mathbf{x}_0 - \xi = \alpha_1\mathbf{U}_1 + \alpha_2\mathbf{U}_2 + \ldots + \alpha_n\mathbf{U}_n.$$

Вектори глобальної похибки на наступних ітераціях згідно з (3.20) обчислюються множенням похибки на матрицю **M**, якій у вибраному координатному базисі відповідає процедура множення окремих проекцій вектора поточної глобальної похибки на відповідне власне значення матриці **M**:

$$\mathbf{x}^{(1)} - \xi = \mathbf{M}(\mathbf{x}^{(0)} - \xi) = \alpha_1\lambda_1\mathbf{U}_1 + \alpha_2\lambda_2\mathbf{U}_2 + \ldots + \alpha_n\lambda_n\mathbf{U}_n,$$
$$\mathbf{x}^{(k+1)} - \xi = \mathbf{M}^{k+1}(\mathbf{x}^{(0)} - \xi) = \alpha_1\lambda_1^{k+1}\mathbf{U}_1 + \ldots + \alpha_n\lambda_n^{k+1}\mathbf{U}_n.$$

Якщо власні значення матриці **M** істотно різняться, умова збіжності, що збігається з умовою зменшення глобальної похибки, набуває бажаного вигляду:

$$\max_i |\lambda_i(\mathbf{M})| = \lambda_{\max} < 1. \tag{3.27}$$

Умова (3.27) має лише теоретичну цінність, тому що пошук власних значень матриці є більш складною задачею, ніж розв'язання систем рівнянь.

3.5. Метод Якобі

Ітераційний метод Якобі — метод розв'язання системи рівнянь $\mathbf{Ax} = \mathbf{b}$ за умови $a_{ii} \neq 0$ — передбачає розв'язання кожного рівняння системи окремо відносно тільки однієї змінної у припущенні, що всі інші змінні фіксовані. Ітераційна процедура методу Якобі має такий вигляд:

$$x_i^{(k+1)} = \frac{b_i - \sum\limits_{j=1, i\neq j}^{n} a_{ij} x_j^{(k)}}{a_{ii}}, \quad i = 1, 2, ..., n. \tag{3.28}$$

Запишемо метод Якобі в матричній формі. Розділимо матрицю A на матриці — діагональну **D** та нижню **L** і верхню **U** трикутні:

$$\mathbf{A} = \mathbf{D}(\mathbf{L} + \mathbf{E} + \mathbf{U}), \tag{3.29}$$

де

$$\mathbf{D} = \begin{bmatrix} a_{11} & 0 & ... & 0 \\ 0 & a_{22} & ... & 0 \\ ... & ... & ... & ... \\ 0 & 0 & ... & a_{nn} \end{bmatrix}, \quad \mathbf{L} = \begin{bmatrix} 0 & & & \\ a_{21}/a_{22} & 0 & & \\ ... & ... & & \\ a_{n1}/a_{nn} & a_{n2}/a_{nn} & ... & 0 \end{bmatrix},$$

$$\mathbf{U} = \begin{bmatrix} 0 & a_{12}/a_{11} & ... & a_{1n}/a_{11} \\ & 0 & ... & a_{2n}/a_{22} \\ & & ... & ... \\ & & & 0 \end{bmatrix}.$$

Тоді рівняння (3.16) для методу Якобі можна записати у вигляді

$$\mathbf{x}^{(k+1)} = -(\mathbf{L} + \mathbf{U})\mathbf{x}^{(k)} + \mathbf{D}^{-1}\mathbf{b},$$

з якого легко доводиться, що матриця перетворення

$$\mathbf{M} = -(\mathbf{L} + \mathbf{U}), \tag{3.30}$$

при цьому

$$m_{i,j} = \begin{cases} -a_{ij}/a_{ii}, & i \neq j, \\ 0, & i = j. \end{cases}$$

Матриця **A** має домінуючу головну діагональ, якщо кожний діагональний елемент цієї матриці за модулем більший, ніж сума модулів інших елементів цього ж рядка:

$$|a_{ii}| > \sum_{k=1, k\neq i}^{n} |a_{ik}|, \quad i = 1, 2, ..., n. \tag{3.31}$$

Покажемо, що для збіжності методу Якобі достатньо, щоб матриця A мала домінуючу головну діагональ. Дійсно, враховуючи вираз (3.31), отримаємо:

$$\|\mathbf{M}\|_\infty = \max_i \sum_{k=1,k\neq i}^{n} \left|\frac{a_{ik}}{a_{ii}}\right| < 1. \qquad (3.32)$$

Приклад 3.7

Розв'яжемо лінійну систему рівнянь із прикладу 3.4 ітераційним методом Якобі:

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 \\ 1 & 3 & 1 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 1 & 5 & 1 \\ 0 & 0 & 0 & 1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 2 \\ -3 \\ 2 \end{bmatrix}.$$

Перш за все оцінимо максимальну норму матриці перетворення **M**, використавши формулу (3.32) для $\|\mathbf{M}\|_\infty$ для рядків: $\|\mathbf{M}\|_\infty = \max(1/2,\ 2/3,\ 2/4,\ 2/5,\ 1/3) = 2/3 = 0{,}666667 < 1$, тобто метод Якобі для даної системи буде збігатися. Після ітерацій за формулою (3.28) методу Якобі отримують значення, зведені у табл. 3.1.

Таблиця 3.1. Результати розв'язання системи рівнянь методом Якобі

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$x_4^{(k)}$	$x_5^{(k)}$
1	0,5	–0,333333	0,5	–0,6	0,666667
2	0,666667	–0,666667	0,733333	–0,833333	0,866667
3	0,833333	–0,8	0,875	–0,92	0,944444
4	0,9	–0,902778	0,93	–0,963889	0,973333
5	0,951389	–0,943333	0,966667	–0,980667	0,987963
6	0,971667	0,972685	0,981	–0,990926	0,993556
7	0,986343	–0,984222	0,990903	–0,994911	0,996975
8	0,992111	–0,992415	0,994783	–0,997576	0,998304
⋮	⋮	⋮	⋮	⋮	⋮
∞	1	–1	1	–1	1

Оцінимо похибку методу Якобі після п'ятої ітерації, скориставшись формулою (3.23):

$$\|\mathbf{x}^{(5)} - \xi\|_\infty \leqslant \frac{0{,}667}{1-0{,}667}\|\mathbf{x}^{(5)} - \mathbf{x}^{(4)}\|_\infty =$$

$$= \max_i(5{,}14,\ 4{,}05,\ 3{,}67,\ 1{,}17,\ 1{,}46)\cdot 10^{-2} = 10{,}28\cdot 10^{-2}.$$

ПРИМІТКА

У математичних пакетах типу Mathematica, які не призначені для розв'язання великих за розміром систем лінійних рівнянь, відсутні вбудовані засоби, що реалізують ітераційні методи Якобі та Гаусса–Зейделя. Проте завдяки можливостям пакета відповідні методи можна запрограмувати.

Приклад 3.8

Розв'яжемо методом Якобі систему рівнянь із матрицею **A** і вектором правої частини **b**:

```
In[]:= A = {{5, -1, -2, 0}, {-1, 6, 2, -1}, {-1, 1, 4, -1}, {0, -1, -1, -3}};
       b = {1, 2, 0, 1};
```

Перепишемо систему у вигляді (3.29), сформувавши для цього діагональну матрицю DE:

```
In[]:= DE = DiagonalMatrix[{5, 6, 4, -3}]; MatrixForm[DE]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 5 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & -3 \end{pmatrix}$$

Знайдемо матрицю перетворення $\mathbf{M} = -\mathbf{D}^{-1}(\mathbf{A} - \mathbf{DE})$ та вектор $\mathbf{g} = \mathbf{D}^{-1}\mathbf{b}$:

```
In[]:= M = -Inverse[DE] . (A - DE); MatrixForm[M]
Out[]//MatrixForm=
```

$$\begin{pmatrix} 0 & \frac{1}{5} & \frac{2}{5} & 0 \\ \frac{1}{6} & 0 & -\frac{1}{3} & \frac{1}{6} \\ \frac{1}{4} & -\frac{1}{4} & 0 & \frac{1}{4} \\ 0 & -\frac{1}{3} & -\frac{1}{3} & 0 \end{pmatrix}$$

```
In[]:= g = Inverse[DE] . b
```

$$\text{Out[]=} \left\{\frac{1}{5}, \frac{1}{3}, 0, -\frac{1}{3}\right\}$$

і зведемо систему до вигляду (3.16).

Використаємо процедури Mathematica, які визначають норму матриці й норму вектора. Визначимо норму матриці **M**:

```
In[]:= M4 = Norm[M, ∞]
```

$$\text{Out[]=} \frac{3}{4}$$

Оскільки норма **M** менше одиниці, умова збіжності метода Якобі задовольняється. Задамо необхідну точність обчислень і початкове наближення:

```
In[]:= ep = 0.00001; x0 = g;
```

Виконаємо перший крок обчислень згідно з (3.16)

```
In[]:= x1 = M . x0 + g
```

$$\text{Out[]=} \left\{\frac{4}{15}, \frac{14}{15}, -\frac{7}{60}, -\frac{4}{9}\right\}$$

Оцінимо похибку на першому кроці за (3.23):

```
In[]:= S = M4/(1 - M4); ε1 = N[Norm[[x1 - x0, ∞]*S]
Out[]= 0.35
```

Виконаємо другий крок згідно з формулою (3.16):

```
In[]:= x2 = M . x1 + g
```

$$\text{Out[]=} \left\{\frac{97}{450}, \frac{37}{108}, -\frac{11}{90}, -\frac{43}{108}\right\}$$

Оцінимо похибку на другому кроці за формулою (3.26)

```
In[]:= ϵ2 = N[Norm[[x2 - x1, ∞]*S]
Out[]= 0.153333
```

Похибка зменшується за геометричною прогресією не повільніше, ніж у $\|\mathbf{M}\|$ разів. Виконаємо необхідну кількість кроків для досягнення заданої похибки

```
In[]:= ek = 10; k = 0; x = g;
       While[ek > ep, y = N[M . x + g]; ek = Max[Abs[y - x]*S]; x = y; k = k ++];
```

Виведемо на екран отриманий розв'язок і число ітерацій:

```
In[]:= Print["x = ", x, "k = ", k - 1];
Out[]= x = {0.215806, 0.346504, -0.133738, -0.404256}, k = -1
```

Отримаємо розв'язок системи рівнянь за допомогою стандартного оператора Mathematica:

```
In[]:= N[LinearSolve[A, b]]
Out[]= {0.215805, 0.346505, -0.133739, -0.404255}
```

Як бачимо, результати збігаються у межах заданої похибки.

3.6. Метод Гаусса–Зейделя

Ітераційний метод Гаусса–Зейделя – метод розв'язання лінійної системи рівнянь $\mathbf{Ax} = \mathbf{b}$ за умови $a_{ii} \neq 0$ — також передбачає розв'язання кожного рівняння системи окремо відносно тільки однієї змінної. Однак під час обчислення i-ї компоненти вектора розв'язку $(k+1)$-го наближення на поточній $(k+1)$-й ітерації використовуються вже знайдені компоненти $(k+1)$-го наближення з меншими індексами:

$$x_i^{(k+1)} = \frac{b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i-1}^{n} a_{ij} x_j^{(k)}}{a_{ii}}, \quad i = 1, 2, ..., n. \tag{3.33}$$

Для реалізації методу Гаусса–Зейделя необхідно менше оперативної пам'яті, ніж для реалізації методу Якобі, тому що після обчислення i-ї компоненти вектора $\mathbf{x}^{(k+1)}$ відповідна компонента вектора $\mathbf{x}^{(k)}$ стає непотрібною. Оскільки дані $(k+1)$-го наближення використовуються для знаходження самого $(k+1)$-го наближення, то перший із вказаних вище методів іноді називають *неявним* ітераційним методом на відміну від другого, який вважається *явним*.

Отже, рівняння (3.16) для методу Гаусса–Зейделя можна записати у вигляді:

$$\mathbf{x}^{(k+1)} = -\mathbf{L}\mathbf{x}^{(k+1)} - \mathbf{U}\mathbf{x}^{(k)} + \mathbf{D}^{-1}\mathbf{b},$$

звідки матриця перетворення

$$\mathbf{M} = -(\mathbf{E} + \mathbf{L})^{-1}\mathbf{U}. \tag{3.34}$$

Таким чином, для збіжності методу Гаусса–Зейделя необхідно і достатньо, щоб усі власні значення цієї матриці були за модулем меншими за одиницю або норма матриці (3.34) була меншою за одиницю.

Щоб уникнути обчислення оберненої матриці у виразі (3.34), ще раз перепишемо рекурентну формулу методу через нормовані значення коефіцієнтів (3.31):

$$x_i^{(k+1)} = \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{(k+1)} + \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} x_j^{(k)} + \frac{b_i}{a_{ii}},$$
$$\left\|x_i^{(k+1)}\right\| \leqslant \|s_i\| \left\|x_i^{(k+1)}\right\| + \|r_i\| \left\|x_i^{(k)}\right\| + \|c_i\|, \tag{3.35}$$

де

$$\|r_i\| = \sum_{k=i+1}^{n} \left|\frac{a_{ik}}{a_{ii}}\right|, \quad \|s_i\| = \max_i \sum_{k=1}^{i-1} \left|\frac{a_{ik}}{a_{ii}}\right|, \quad \|c_i\| = \left|\frac{b_i}{a_{ii}}\right|. \tag{3.36}$$

Тому для симетричної позитивно визначеної матриці **A** з домінуючою головною діагоналлю умова збіжності обчислень приймає вигляд:

$$\|\mathbf{M}\|_\infty = \max_i \left|\frac{r_i}{1 - s_i}\right| < 1. \tag{3.37}$$

Необхідну кількість ітерацій у разі вибору початкового наближення нормованої правої частини $c_i = b_i/a_{ii}$ з метою одержання наближеного розв'язку лінійної системи із заданою точністю ε, можна визначити, скориставшись формулою (3.24):

$$\varepsilon = \left\|\mathbf{x}^{(k+1)} - \xi\right\| \leqslant \frac{\|\mathbf{M}\|^{(k+1)}}{1 - \|\mathbf{M}\|} \|\mathbf{C}\|, \tag{3.38}$$

звідки

$$(k+1)\lg\|\mathbf{M}\| \leqslant \lg\varepsilon - \lg\|\mathbf{C}\| + \lg(1 - \|\mathbf{M}\|)$$

або

$$k \geqslant \frac{1}{\lg\|\mathbf{M}\|}[\lg\|\mathbf{C}\| - \lg(\varepsilon(1 - \|\mathbf{M}\|))] - 1. \tag{3.39}$$

Приклад 3.9

Розв'яжемо методом Гаусса–Зейделя систему лінійних рівнянь, яка розглядалася в прикладах 3.4 і 3.7:

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 \\ 1 & 3 & 1 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 1 & 5 & 1 \\ 0 & 0 & 0 & 1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 2 \\ -3 \\ 2 \end{bmatrix}.$$

Оцінимо максимальну норму матриці перетворення $\|\mathbf{M}\|_\infty$, використавши формули (3.36) та (3.37) для рядків матриці:

1-й $s_1 = 0,\ r_1 = 1/2,\ M_1 = 1/2,$
2-й $s_2 = 1/3,\ r_2 = 1/3,\ M_2 = 1/3/(1-1/3) = 1/2,$
3-й $s_3 = 1/4,\ r_3 = 1/4,\ M_3 = 1/3,$
4-й $s_4 = 1/5,\ r_4 = 1/5,\ M_4 = 1/4,$
5-й $s_5 = 1/3,\ r_5 = 0,\ M_5 = 0.$

$\|\mathbf{M}\|_\infty = \max(1/2,\ 1/2,\ 1/3,\ 1/4,\ 0) = 1/2 = 0,5 < 1$, тобто метод Гаусса–Зейделя теж стійкий для даного прикладу.

Внаслідок виконання ітерацій за формулою (3.34) метода Гаусса–Зейделя отримаємо значення, зведені у табл. 3.2.

Таблиця 3.2. Результати розв'язання системи рівнянь методом Гаусса–Зейделя

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$x_4^{(k)}$	$x_5^{(k)}$
1	0,5	–0,333333	0,5	–0,6	0,666667
2	0,666667	–0,722222	0,830556	–0,899444	0,966481
3	0,861111	–0,897222	0,949167	–0,98313	0,994377
4	0,948611	–0,965926	0,987264	–0,996328	0,998776
5	0,982963	–0,990076	0,996601	–0,999075	0,999692
⋮	⋮	⋮	⋮	⋮	⋮
∞	1	–1	1	–1	1

Для оцінки похибки розв'язку системи методом Гаусса–Зейделя після п'ятої ітерації скористаємося формулою (3.23):

$$\|\mathbf{x}^{(5)} - \xi\|_\infty \leqslant \frac{0,5}{1-0,5}\|\mathbf{x}^{(5)} - \mathbf{x}^{(4)}\|_\infty = \max_i(2,23,\ 1,7,\ 0,69,\ 0,21,\ 0,069)\cdot 10^{-2} = 2,33\cdot 10^{-2}.$$

Тобто для вибраного прикладу похибка обчислень після п'ятої ітерації методу Гаусса–Зейделя приблизно в чотири рази менша, ніж похибка обчислень після п'ятої ітерації методу Якобі, а для досягнення тієї ж точності першого методу знадобилася менша кількість ітерацій, ніж кількість ітерацій другого.

Результати оцінювання норми вектора похибки і дані табл. 3.2 свідчать про те, що найбільша за значенням похибка припадає на невідому x_1, оскільки під час її визначення використовується найбільш застаріла інформація.

Ситуацію можна виправити, якщо разом із формулою (3.33) використовувати симетричну формулу:

$$x_i^{(k+1)} = \frac{b_i - \sum\limits_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum\limits_{j=i+1}^{n} a_{ij}x_j^{(k+1)}}{a_{ii}}, \quad i = 1, 2, ..., n,$$

так щоб одна ітерація (прогін рівнянь) виконувалася зверху вниз (від першого рівняння до останнього), а друга ітерація — навпаки, знизу вверх (від останнього рівняння до першого). Потім отримані дані двох ітерацій усереднюються (у таб-

лиці це рядок с.з.), і з ними виконуються наступні дві ітерації з протилежними напрямками і т. д.

Таблиця 3.3. Результати розв'язання системи рівнянь симетричним методом Гаусса–Зейделя

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$x_4^{(k)}$	$x_5^{(k)}$
1	0,5	–0,333333	0,5	–0,6	0,666667
2	0,666667	–0,722222	0,830556	–0,899444	0,966481
с.з.	0,583334	–0,5277775	0,665278	–0,749722	0,816574
3	0,583333	–0,527778	0,665278	–0,749722	0,816574
4	0,763889	–0,809722	0,889861	–0,941287	0,980429
с.з.	0,673611	–0,66875	0,7775695	–0,8455045	0,8985015
⋮	⋮	⋮	⋮	⋮	⋮
∞	1	–1	1	–1	1

Розглянемо геометричну інтерпретацію метода Гаусса–Зейделя на прикладі розв'язування системи рівнянь:

$$\begin{cases} 2x + y = 2, \\ x - 2y = -2, \end{cases} \quad \mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & -2 \end{bmatrix},$$

для якої згідно з виразом (3.37) необхідні і достатні умови збіжності виконуються:

$$\|\mathbf{M}\|_\infty = \max(1/2,\ 0) = 1/2 = 0{,}5 < 1.$$

Оскільки під час обчислення змінної x зберігається незмінним значення y, то геометричним еквівалентом методу є рух по горизонталі до перетину з прямою, що відображує перше рівняння (рис. 3.3).

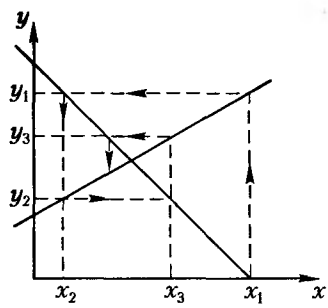


Рис. 3.3. Геометрична інтерпретація методу Гаусса–Зейделя

Потім рух здійснюється по вертикалі за збереження незмінним знайденого значення x до перетину з прямою, що зображує друге рівняння:

$$x^{(k)} = \frac{1}{2}(2 - y^{(k-1)}), \quad y^{(k)} = \frac{1}{2}(x^{(k)} + 2).$$

Значення координат виділених точок x_i, $i = 0, 1, 2, 3$ зведені у табл. 3.4.

Таблиця 3.4. Координати точок перетину

k	x	y
0	0	0
1	1	3/2
2	1/4	9/8
3	7/16	39/32

Приклад 3.10

Повторимо процес розв'язування системи рівнянь із матрицею **A** і вектором правої частини **b**, наведеними в прикладі 3.8, методом Гаусса–Зейделя, запрограмованим у пакеті Mathematica:

```
In[]:= A = {{5, -1, -2, 0}, {-1, 6, 2, -1}, {-1, 1, 4, -1}, {0, -1, -1, -3}};
       b = {1, 2, 0, 1};
```

Обчислення невідомих $x_j^{(k+1)}$ будемо виконувати за формулою (3.35) послідовно для кожної компоненти $x_j^{(k+1)}$, $j = 1, 2, ..., n$, тому що нові, отримані на $(k+1)$-й ітерації значення мають використовуватися у наступних обчисленнях. Знайдемо матриці **D**, **L** і **U** векторної формули (3.34):

```
In[]:= DE = DiagonalMatrix[{5, 6, 4, -3}]; g = Inverse[DE] . b; n = Length[A];
       L = Table[If [i > j, l[i,j] = A[[i,j]]/A[[i,i]], l[i,j] = 0], {i,n}, {j,n} ];
       MatrixForm[L]
```

```
Out[]//MatrixForm=
```

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ -\frac{1}{6} & 0 & 0 & 0 \\ -\frac{1}{4} & \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{3} & \frac{1}{3} & 0 \end{pmatrix}$$

```
In[]:= U = Table[If[i <j, u[i,j] = A[[i,j]]/ A[[i,i]], U[i,j] = 0], {i,n}, {j,n}];
       MatrixForm[U]
```

```
Out[]//MatrixForm=
```

$$\begin{pmatrix} 0 & -\frac{1}{5} & -\frac{2}{5} & 0 \\ 0 & 0 & \frac{1}{3} & -\frac{1}{6} \\ 0 & 0 & 0 & -\frac{1}{4} \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Тепер можна знайти матрицю **M** (MJ) і перевірити умову збіжності методу Гаусса–Зейделя:

```
In[]:= MJ = -Inverse[(IdentityMatrix[n] + L)] . U; NMJ = Norm[MJ, ∞]
```

```
Out[]= 3/5
```

Оскільки норма **M** менша за одиницю, забезпечена збіжність методу Гаусса–Зейделя зі швидкістю геометричної прогресії, знаменник якої дорівнює 0,6. Під час розв'язання цього прикладу методом Якобі (приклад 3.8) швидкість збіжності визначалася знаменником, що дорівнював 0,75. Таким чином, метод Гаусса–Зейделя буде збігатися швидше, ніж метод Якобі. Складемо програму. Задамо необхідну похибку обчислень, початкове наближення і використаємо значення матриці **M** із прикладу 3.8:

```
In[]:= ek = 10; ep = 0.00001; k = 0; x = g; S = NMJ/(1 - NMJ); M = -Inverse[DE] . (A - DE);
       While [ek > ep, ek = 0;
         Do[y = N[g[[i]] + Sum[M[[i,j]]*x[[j]], {j,n}] ];
            ek = Max[Abs[y - x[[i]]], ek]; x[[i]] = y, {i,n} ]; ek = ek*S; k = k + 1];
       Print["x = ", x, ", ek = ", ek, ", k = ", k - 1]

Out[]= x = {0.215806, 0.346504, -0.133738, -0.404255}, ek = 5.65249x10^-6, k = 6.
```

Бачимо, що розв'язок із тією ж похибкою отримано всього за шість ітерацій, а не за дев'ять, як у разі використання методу Якобі.

3.7. Засоби пакета Mathematica для розв'язання систем лінійних рівнянь із розрідженою матрицею

Враховуючи значне поширення систем лінійних рівнянь із тридіагональними матрицями типу

$$\begin{bmatrix} \alpha_1 & \gamma_1 & & & & & \\ \beta_2 & \alpha_2 & \gamma_2 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & \beta_{n-1} & \alpha_{n-1} & \gamma_{n-1} \\ & & & & & \beta_n & \alpha_n \end{bmatrix}$$

у пакеті Mathematica передбачено спеціальний оператор `TridiagonalSolve`, що реалізує метод прогону (3.9) – (3.11) і для рівняння $\mathbf{Ax} = \mathbf{b}$ має такий формат:

```
TridiagonalSolve[β, α, γ, b],
```

де β_i, $i = 2, ..., n$, α_i, $i = 1, ..., n$, γ_i, $i = 1, ..., n-1$ — діагоналі матриці коефіцієнтів рівнянь, $\mathbf{b}$ — права частина системи рівнянь. Наприклад, для системи рівнянь, розглянутої в прикладі 3.4,

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 \\ 1 & 3 & 1 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 1 & 5 & 1 \\ 0 & 0 & 0 & 1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \\ 2 \\ -3 \\ 2 \end{bmatrix}$$

розв'язок отримуємо таким послідовним записом:

```
In[]:= <<LinearAlgebra`Tridiagonal`
In[]:= TridiagonalSolve[{1, 1, 1, 1}, {2, 3, 4, 5, 3}, {1, 1, 1, 1}, {1, -1, 2, -3, 2}]
Out[]= {1, -1, 1, -1, 1}
```

До речі, той самий оператор дозволяє знайти розв'язок і за комплексних елементів матриці:

```
In[]:= <<LinearAlgebra`Tridiagonal`
In[]:= TridiagonalSolve[{2.2, 5.2}, {4.3, 3.22, 2.1}, {8.1 + i, 3.4}, {1, 2.3i , 3}]
Out[]= {-0.885678 + 0.391313i, 0.559458 - 0.276803i, 0.0432458 + 0.685417i}
```

Якщо система лінійних рівнянь розріджена, то краще її розв'язувати безпосередньо за допомогою оператора Solve чи NSolve, не використовуючи матричну форму запису, запобігаючи введенню її нульових елементів.

У пакеті передбачена також окрема процедура розв'язування систем лінійних рівнянь з розрідженими матрицями. Спочатку такі матриці **M** записуються у формі $i_1, j_1 \to a_1, i_2, j_2 \to a_2, \ldots$ з вказівкою лише на розташування ненульових елементів j_k, i_k та їх значень a_k. Потім розріджена система рівнянь розв'язується за допомогою оператора LinearSolve[M, vec], де vec — вектор правої частини розрідженої системи рівнянь. При цьому використовуються алгоритми упорядкування за Марковицем (3.1) і виключення Гаусса.

Для системи рівнянь, розглянутої в прикладі 3.4, отримуємо:

```
In[]:= M = SparseArray[
          {{1, 1} → 2, {1, 2} → 1,
           {2, 1} → 1, {2, 2} → 3,  {2, 3} → 1,
                       {3, 2} → 1,  {3, 3} → 4,  {3, 4} → 1,
                                    {4, 3} → 1,  {4, 4} → 5,  {4, 5} → 1,
                                                 {5, 4} → 1,  {5, 5} → 3}]
Out[]= SparseArray[<13>, {5, 5}]
In[]:= Normal[M]// MatrixForm
Out[]//MatrixForm=
```

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 \\ 1 & 3 & 1 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 1 & 5 & 1 \\ 0 & 0 & 0 & 1 & 3 \end{bmatrix}$$

```
In[]:= LinearSolve[M, {1, -1, 2, -3, 2}]
Out[]= {1, -1, 1, -1, 1}
```

Передбачена також окрема процедура обробки розріджених матриць. Спочатку за допомогою пакета <<LinearAlgebra`MatrixManipulation` і оператора LinearSparseEquationsToMatrices [eqns, vars] для розріджених систем формується матриця **M** у вигляді $i_1, j_1 \to a_1, i_2, j_2 \to a_2, \ldots$ із вказівкою лише на розташування ненульових елементів j_k, i_k та їх значень a_k, а також вектор правої частини системи рівнянь vec. Потім розріджена система рівнянь розв'язується допоміжним оператором SparseLinearSolve[M, vec]. Оператор автоматично включається за умови ви-

користання операторів Solve і Nsolve, при цьому застосовується алгоритм упорядкування за Марковицем (3.1) і виключення Гаусса.
Для системи рівнянь, розглянутої в прикладі 3.4, отримуємо:

```
In[]:= Solve[{2 * x + y == 1, x + 3 * y + z == -1, y + 4 * z + s == 2,
              z + 5 * s + t == -3, s + 3 * t == 2}, {x, y, z, s, t}]
Out[]= {{x → 1, y → -1, z → 1, s → -1, t → 1}}
```

Нагадаємо: в документації до пакету Mathematica вказується, що використовуючи оператори цього пакету, можна розв'язувати розріджені лінійні системи рівнянь, які містять до мільйона невідомих, якщо кількість ненульових елементів матриць їх коефіцієнтів досягає сотень тисяч.

Висновки

1. У багатьох галузях сучасної техніки інженери використовують системи лінійних алгебраїчних рівнянь, що містять тисячі й сотні тисяч невідомих. У цих умовах прямі методи розв'язання рівнянь втрачають свою ефективність, і на передній план висуваються методи діакоптики, або методи розв'язання систем рівнянь по частинах. Найпростішими з таких методів є ітераційні методи розв'язання систем лінійних рівнянь, які потребують багаторазового розв'язання кожного рівняння окремо відносно однієї змінної у припущенні, що інші змінні відомі.
2. Застосування ітераційних методів розв'язання систем лінійних рівнянь дозволяє збільшувати розмірність задачі за рахунок додаткового обсягу обчислень, що пов'язано з необхідністю повторного розв'язання (прогонів) задачі. Але для забезпечення збіжності обчислень всі ітераційні методи накладають істотні обмеження на рекурентні формули ітерацій; зокрема, для зменшення *глобальної похибки* вони вимагають, щоб норма матриці перетворення була меншою за одиницю.
3. Умова збіжності ітераційних методів еквівалентна вимозі *домінантності* матриць коефіцієнтів систем лінійних рівнянь, тобто вимозі, щоб їх діагональні елементи за абсолютними значеннями перевищували суми абсолютних значень усіх інших елементів, розташованих у відповідному рядку чи стовпці таких матриць. У математичних задачах, які описують фізичні системи, вимога домінантності матриць коефіцієнтів рівнянь задовольняється відносно рідко. Прикладом такої задачі є моделювання пасивних електротехнічних схем за допомогою методу вузлових напруг. Під час розв'язання диференціальних рівнянь із частинними похідними методом скінченних різниць домінантність матриць систем лінеаризованих рівнянь забезпечується безпосередньо формулами різницевої апроксимації першої та другої похідних.
4. Збіжність ітераційних методів Якобі та Гаусса–Зейделя різна, тому що в останньому на черговій ітерації для знаходження невідомої використовуються значення інших невідомих, що були отримані на попередніх ітераціях.

5. Системи лінійних рівнянь великої розмірності, як правило, є розрідженими. Їх матриці містять багато нульових елементів, що дозволяє використовувати для опису матриць кодуючі масиви. Це істотно зменшує обсяг обчислень.
6. В інженерній практиці поширені системи рівнянь зі стрічковими матрицями, для розв'язання яких застосовуються методи спрощеного LU-розкладання, прогону і верхньої релаксації.
7. Перевага методу визначальних величин над класичними ітераційними методами полягає в тому, що він дозволяє зберегти послідовність обчислень, притаманну прямим методам Гаусса і LU-розкладання. При цьому істотно знижується розмірність системи рівнянь для визначальних величин (або змінних зв'язку). Крім того, даний метод не вимагає виконання умови домінантності матриці.
8. Принципово метод Якобі дозволяє провести повне розпаралелювання процедури обчислень, оскільки кожне рівняння розв'язується незалежно від інших. Розпаралелювання процесу розв'язування систем лінійних рівнянь великої розмірності (без вимоги їх домінантності!) здійснюється за допомогою масивно-паралельних комп'ютерів або в локальних комп'ютерних мережах.

Контрольні запитання та завдання

1. Побудувати ненульову структуру матриці за її кодуючими списками

$$LJ = [1, 3, 2, 4, 1, 3, 5, 2, 4, 1, 4, 5], \quad LI = [1, 3, 5, 8, 10, 13]$$

і визначити, скільки нових ненульових елементів (НЕ) з'являється у разі її LU-розкладання.

2. Побудувати ненульову структуру матриці за її кодуючими списками

$$LJ = [1, 3, 2, 4, 1, 2, 3, 5, 1, 2, 4, 1, 4, 5], \quad LI = [1, 3, 5, 9, 12, 15]$$

і звести її до блочно-діагонального вигляду з обрамленням. До якого порядку може бути скорочений розмір системи рівнянь змінних зв'язку?

3. Побудувати ненульову структуру матриці за її кодуючими списками

$$LJ = [1, 2, 3, 2, 4, 1, 2, 3, 5, 1, 2, 4, 1, 4, 5], \quad LI = [1, 4, 6, 10, 13, 16]$$

і провести упорядкування діагональних елементів матриці за критерієм Марковиця. Як зміниться кількість нових ненульових елементів (НЕ) у разі LU-розкладання?

4. Розв'язати ітераційним методом Гаусса–Зейделя систему лінійних алгебраїчних рівнянь:

$$\begin{cases} 3x_1 - 0,1x_2 - 0,2x_3 = 7,85, \\ 0,1x_1 + 7x_2 - 0,3x_3 = -19,3, \\ 0,3x_1 - 0,2x_2 + 10x_3 = 71,4. \end{cases}$$

Знайти вектори розв'язку після першої та другої ітерацій і похибку розв'язку після другої ітерації, якщо вектор точного розв'язку дорівнює $\mathbf{x}^T = [3, -25, 7]$.

5. Розв'язати з похибкою $\Delta = 0,05$ ітераційним методом Якобі систему лінійних рівнянь

$$\begin{bmatrix} 8 & -1 & 2 \\ 1 & 9 & 3 \\ 2 & -3 & 10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 8 \\ 18 \\ -5 \end{bmatrix}.$$

6. Розв'язати ітераційним методом Гаусса–Зейделя систему лінійних рівнянь

$$\begin{bmatrix} 7 & 2 & -4 \\ 1 & 5 & -3 \\ 2 & -1 & -9 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -18 \\ -40 \\ -26 \end{bmatrix}.$$

Розділ 4

Обчислення власних значень і власних векторів матриці

- Метод Фаддєєва–Левер'є
- Метод Крилова
- QR-алгоритм
- Перетворення подібності Хаусхолдера
- Степеневий метод

У даному розділі розглянуті прямі та ітераційні методи обчислення власних значень і власних векторів матриці [35, 37]. До першої групи методів належать методи, що базуються на обчисленні коренів характеристичного рівняння матриці, наприклад методи Фаддєєва–Левер'є, Данилевського і Крилова. Їх звичайно застосовують для матриць невеликої розмірності. До другої групи відносять методи, що використовують ортогональні перетворення подібності і зводять матрицю до блочно-діагонального виду з блоками першого і другого порядків. Серед таких методів слід виділити методи Якобі, Гівенса, перетворення Хаусхолдера та QR-алгоритм. Сюди ж можна віднести також степеневий метод і різні його модифікації.

4.1. Метод характеристичного рівняння матриці

У загальному випадку результатом множення матриці **A** на довільний ненульовий вектор **x** є новий вектор $\mathbf{y} = \mathbf{Ax}$, який відрізняється від початкового вектора **x** за довжиною (модулем) і напрямком (орієнтацією) у багатовимірному просторі. Але в деяких випадках після такої операції множення новий вектор **y** не змінює свою орієнтацію, а змінює тільки довжину, тобто окремі його компоненти стають пропорційними відповідним компонентам вектора **x**. Такі вектори **x** називають *власними векторами* матриці **A**, а коефіцієнти пропорційності λ – власними значеннями (або числами) матриці **A**. Ця властивість лежить в основі обчислення власних значень і власних векторів матриці. Квадратна матриця розмірності $n \times n$ має n власних значень і власних векторів.

Власними значеннями матриці називають такі скалярні величини, які є коренями рівняння

$$\mathbf{Ax} = \lambda\mathbf{x}$$

або

$$(\mathbf{A} - \lambda\mathbf{E})\mathbf{x} = 0. \tag{4.1}$$

Звідси, якщо $\mathbf{x} \neq 0$, перепишемо тотожність (4.1) у такому вигляді:

$$\det(\mathbf{A} - \lambda\mathbf{E}) = P(\lambda) = 0. \tag{4.2}$$

Якщо визначник (4.2) розкрити відносно значень λ, то отримаємо так зване *характеристичне рівняння* матриці $\mathbf{A}$ у вигляді полінома n-степеня $P(\lambda)$ відносно власних значень. Розв'язок цього рівняння визначає множину всіх власних значень матриці, яка називається її *спектром*. Кожному власному значенню матриці відповідає свій власний вектор.

4.1.1. Метод Фадєєва–Левер'є

Для обчислення коефіцієнтів характеристичного рівняння крім прямого розкриття самого визначника існує декілька методів, серед яких виділяється метод Фаддєєва–Левер'є, за допомогою якого обчислюються:

- коефіцієнти характеристичного полінома

$$\det(\mathbf{A} - \lambda\mathbf{E}) = \lambda^n + b_{n-1}\lambda^{n-1} + b_{n-2}\lambda^{n-2} + \ldots + b_1\lambda + b_0; \tag{4.3}$$

- обернена матриця

$$\mathbf{A}^{-1} = -\frac{\mathbf{K}_0}{b_0}; \tag{4.4}$$

- резольвента матриці

$$(\mathbf{A} - \lambda\mathbf{E})^{-1} = \frac{\mathbf{K}_{n-1}\lambda^{n-1} + \mathbf{K}_{n-2}\lambda^{n-2} + \ldots + \mathbf{K}_1\lambda + \mathbf{K}_0}{\det(\mathbf{A} - \lambda\mathbf{E})}. \tag{4.5}$$

Метод Фаддєєва–Левер'є базується на результатах обчислення слідів матриці $\mathbf{A}$ і добутків матриць $\mathbf{AK}_{n-1}$, де $\mathbf{K}_{n-1}$ — коефіцієнти чисельника резольвенти матриці (4.5). Коефіцієнти чисельника і знаменника виразу (4.5) визначаються ітераційно (спочатку коефіцієнт чисельника $\mathbf{K}_{n-k}$, а потім коефіцієнт знаменника b_{n-k}) відповідно до наведеної нижче процедури, де $\mathbf{E}$ — одинична матриця, $\mathrm{Tr}(\mathbf{A}) = \sum_{i=1}^{n} a_{ii}$ — слід матриці.

$$\begin{aligned}
\mathbf{K}_{n-1} &= \mathbf{E}, & b_{n-1} &= -\mathrm{Tr}(\mathbf{AK}_{n-1}),\\
\mathbf{K}_{n-2} &= \mathbf{AK}_{n-1} + b_{n-1}\mathbf{E}, & b_{n-2} &= -1/2\ \mathrm{Tr}(\mathbf{AK}_{n-2}),\\
&\ldots\ \ldots\ \ldots & &\ldots\ \ldots\ \ldots\\
\mathbf{K}_{n-k} &= \mathbf{AK}_{n-k+1} + b_{n-k+1}\mathbf{E}, & b_{n-k} &= -1/k\ \mathrm{Tr}(\mathbf{AK}_{n-k}),\\
\mathbf{K}_0 &= \mathbf{AK}_1 + b_1\mathbf{E}, & b_0 &= -1/n\ \mathrm{Tr}(\mathbf{AK}_0),
\end{aligned}$$

$\mathbf{AK}_0 + b_0\mathbf{E} = 0$ — умова перевірки.

Приклад 4.1

Побудуємо характеристичне рівняння матриці

$$\mathbf{A} = \begin{bmatrix} -2 & 1 & 3 \\ 0 & -3 & 0 \\ 0 & 2 & -2 \end{bmatrix}.$$

Згідно з методом Фаддєєва–Левер'є (4.5) маємо:

$$(\mathbf{A} - \lambda\mathbf{E})^{-1} = \frac{\mathbf{K}_2\lambda^2 + \mathbf{K}_1\lambda + \mathbf{K}_0}{\lambda^3 + b_2\lambda^2 + b_1\lambda + b_0}.$$

Коефіцієнти чисельника і знаменника знаходять у такий спосіб:

$$\mathbf{K}_2 = \mathbf{E},$$

$$b_2 = -\operatorname{Tr}\mathbf{A} = 7,$$

$$\mathbf{K}_1 = \mathbf{A} + 7\mathbf{E} = \begin{bmatrix} 5 & 1 & 3 \\ 0 & 4 & 0 \\ 0 & 2 & 5 \end{bmatrix}, \quad \mathbf{AK}_1 = \begin{bmatrix} -10 & 8 & 9 \\ 0 & -12 & 0 \\ 0 & 4 & -10 \end{bmatrix},$$

$$b_1 = -\frac{1}{2}\operatorname{Tr}(\mathbf{AK}_1) = 16,$$

$$\mathbf{K}_0 = \mathbf{AK}_1 + 16\mathbf{E} = \begin{bmatrix} 6 & 8 & 9 \\ 0 & 4 & 0 \\ 0 & 4 & 6 \end{bmatrix}, \quad \mathbf{AK}_0 = \begin{bmatrix} -12 & 0 & 0 \\ 0 & -12 & 0 \\ 0 & 0 & -12 \end{bmatrix},$$

$$b_0 = \operatorname{Tr}(\mathbf{AK}_0) = 12.$$

Для контролю правильності перевіряємо умову

$$\mathbf{AK}_0 + b_0\mathbf{E} = 0,$$

$$\begin{bmatrix} -12 & 0 & 0 \\ 0 & -12 & 0 \\ 0 & 0 & -12 \end{bmatrix} + 12\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = 0.$$

Таким чином, характеристичне рівняння матриці має вигляд

$$\lambda^3 + 7\lambda^2 + 16\lambda + 12 = 0,$$

з нього можна знайти власні значення матриці $\lambda = [-3, -2, -2]^{\mathrm{T}}$.
Отже, обернена матриця:

$$\mathbf{A}^{-1} = -\frac{\mathbf{K}_0}{b_0} = \begin{bmatrix} -1/2 & -2/3 & -3/4 \\ 0 & -1/3 & 0 \\ 0 & -1/3 & -1/2 \end{bmatrix}$$

і резольвента матриці:

$$(\mathbf{A}-\lambda\mathbf{E})^{-1}=\frac{1}{\det(\mathbf{A}-\lambda\mathbf{E})}\begin{bmatrix}\lambda^2+5\lambda+6 & \lambda+8 & 3\lambda+9\\ 0 & \lambda^2+4\lambda+4 & 0\\ 0 & 2\lambda+4 & \lambda^2+5\lambda+6\end{bmatrix}=$$

$$=\begin{bmatrix}\dfrac{\lambda^2+5\lambda+6}{-\lambda^3-7\lambda^2-16\lambda-12} & \dfrac{\lambda+8}{-\lambda^3-7\lambda^2-16\lambda-12} & \dfrac{3\lambda+9}{-\lambda^3-7\lambda^2-16\lambda-12}\\ 0 & \dfrac{\lambda^2+4\lambda+4}{-\lambda^3-7\lambda^2-16\lambda-12} & 0\\ 0 & \dfrac{2\lambda+4}{-\lambda^3-7\lambda^2-16\lambda-12} & \dfrac{\lambda^2+5\lambda+6}{-\lambda^3-7\lambda^2-16\lambda-12}\end{bmatrix}.$$

Розглянутий приклад можна розв'язати і в пакеті Mathematica, використовуючи операції обчислення сліду матриці Tr[A] і множення матриць:

```
In[]:= A = {{-2, 1, 3}, {0, -3, 0}, {0, 2, -2}};
       n = Length[A];
       b[n] = -Tr[A]; b[n+1] = 1; K[n] = IdentityMatrix[n];
       Print["b[",n,"] = ",b[n]," K[",n,"] = ",K[n]];
       Do [
       K[n-i] = A . K[n - i + 1] + b[n - i + 1]*IdentityMatrix[n];
       b[n-i] = -1/(i + 1)*Tr[A . K[n-i]];
       Print["b[",n-i,"] = ",b[n-i]," K[",n-i,"] = ",K[n-i]], {i, 1, n-1}]
```

У результаті отримуємо:

```
b[3]=  7 K[3] = {{1,0,0}, {0,1,0}, {0,0,1}}
b[2]= 16 K[2] = {{5,1,3}, {0,4,0}, {0,2,5}}
b[1]= 12 K[1] = {{6,8,9}, {0,4,0}, {0,4,6}}
```

Характеристичний поліном матриці будуємо за допомогою команд:

```
In[]:= Pol[λ_]:= Sum[λ^i*b[i+1], {i, 0, n}]; Pol[λ]
Out[]= 12 + 16λ + 7λ² + λ³
```

Власні значення матриці отримуємо як розв'язок алгебраїчного рівняння:

```
In[]:= NSolve[Pol[λ] == 0, λ]
Out[]= {{λ→ -3.}, {λ → -2.}, {λ→ -2.}}
```

Резольвенту матриці обчислюємо, використовуючи вираз (4.5):

```
In[]:= MatrixForm[Inverse[A - λ*IdentityMatrix[n]]]
Out[]//MatrixForm=
```

$$\begin{bmatrix}\dfrac{6+5\lambda+\lambda^2}{-12-16\lambda-7\lambda^2-\lambda^3} & \dfrac{8+\lambda}{-12-16\lambda-7\lambda^2-\lambda^3} & \dfrac{9+3\lambda}{-12-16\lambda-7\lambda^2-\lambda^3}\\ 0 & \dfrac{4+4\lambda+\lambda^2}{-12-16\lambda-7\lambda^2-\lambda^3} & 0\\ 0 & \dfrac{4+2\lambda}{-12-16\lambda-7\lambda^2-\lambda^3} & \dfrac{6+5\lambda+\lambda^2}{-12-16\lambda-7\lambda^2-\lambda^3}\end{bmatrix}$$

4.1.2. Метод Крилова

Крім методу Фаддєєва–Левер'є досить поширеним є метод Крилова, який базується на відомій теоремі Келі–Гамільтона, яка стверджує, що будь-яка матриця A задовольняє своє характеристичне рівняння:

$$\mathbf{A}^n + b_{n-1}\mathbf{A}^{n-1} + b_{n-2}\mathbf{A}^{n-2} + \ldots + b_1\mathbf{A} + b_0 = 0. \tag{4.6}$$

Якщо вираз (4.6) помножити на довільний вектор **x**, можна отримати такий запис:

$$\mathbf{x}^{(n)} = \mathbf{A}^n\mathbf{x} = -b_{n-1}\mathbf{x}^{(n-1)} - b_{n-2}\mathbf{x}^{(n-2)} - \ldots - b_1\mathbf{x}^{(1)} - b_0\mathbf{x} = 0, \tag{4.7}$$

де вектори обчислюються за формулами

$$\mathbf{x}^{(1)} = \mathbf{A}\mathbf{x}, \quad \mathbf{x}^{(2)} = \mathbf{A}\mathbf{x}^{(1)} = \mathbf{A}^2\mathbf{x}, \quad \ldots, \quad \mathbf{x}^{(i)} = \mathbf{A}^{(i)}\mathbf{x}, \quad \ldots \tag{4.8}$$

або у компонентному вигляді:

$$x_k^{(i)} = \sum_{j=1}^{i} a_{kj}x_j^{(i-1)}. \tag{4.9}$$

На основі рівняння (4.7) формуємо систему лінійних алгебраїчних рівнянь для коефіцієнтів характеристичного полінома матриці:

$$\begin{bmatrix} x_1^{(n-1)} & x_1^{(n-2)} & \ldots & x_1^{0} \\ x_2^{(n-1)} & x_2^{(n-2)} & \ldots & x_2^{(0)} \\ \ldots & \ldots & \ldots & \ldots \\ x_n^{(n-1)} & x_n^{(n-2)} & \ldots & x_n^{(0)} \end{bmatrix} \begin{bmatrix} b_{n-1} \\ b_{n-2} \\ \ldots \\ b_0 \end{bmatrix} = \begin{bmatrix} -x_1^{(n)} \\ -x_2^{(n)} \\ \ldots \\ -x_n^{(n)} \end{bmatrix}. \tag{4.10}$$

Якщо система (4.10) буде виродженою, необхідно замінити початковий вектор **x**, який для зручності звичайно обирають як $\mathbf{x} = [1, 0, 0, \ldots, 0]^T$.

Приклад 4.2

Користуючись методом Крилова, побудуємо характеристичне рівняння матриці:

$$\mathbf{A} = \begin{bmatrix} -2 & 1 & 3 \\ 0 & -3 & 0 \\ 0 & 2 & -2 \end{bmatrix}.$$

Вибираючи початковий вектор $\mathbf{x}_0 = [1, 1, 0]^T$, знаходимо необхідні вектори (4.8) і будуємо та розв'язуємо систему рівнянь (4.10):

```
In[]:= x[0] = {1, 1, 0};
       A = {{-2, 1, 3}, {0, -3, 0}, {0, 2, -2}};
       n = Length[A];
       Do[x[j] = MatrixPower[A, j]. x[0], {j, n}]
       Array[x, 3]
Out[]= {{-1, -3, 2}, {5, 9, -10}, {-31, -27, 38}}
```

```
In[]:= K = {x[2], x[1], x[0]}        (* Формування матриці системи рівнянь *)
Out[]= {{5, 9, -10}, {-1, -3, 2}, {1, 1, 0}}
In[]:= p = -x[3]                     (* Формування правої частини системи рівнянь *)
Out[]= {31, 27, -38}
In[]:= LinearSolve[Transpose[K], p] (* Обчислення коефіцієнтів характеристичного рівняння *)
Out[]= {7, 16, 12}
```

Отже, отримуємо вектор коефіцієнтів $\mathbf{b} = [7, 16, 12]^T$, який збігається з результатом, отриманим у прикладі 4.1. Під час виконання програми були обчислені степені матриці

$$\mathbf{A}^2 = \begin{bmatrix} 4 & 1 & -12 \\ 0 & 9 & 0 \\ 0 & -10 & 4 \end{bmatrix} \quad \text{та} \quad \mathbf{A}^3 = \begin{bmatrix} -8 & -23 & -12 \\ 0 & -27 & 0 \\ 0 & 38 & -8 \end{bmatrix},$$

знайдені власні вектори $\mathbf{x}_1 = [-1, -3, 2]^T$, $\mathbf{x}_2 = [5, 9, -10]^T$ і $\mathbf{x}_3 = [-31, -27, 38]^T$, а також побудована і розв'язана система рівнянь:

$$\begin{bmatrix} 5 & -1 & 1 \\ 9 & -3 & 1 \\ -10 & 2 & 0 \end{bmatrix} \begin{bmatrix} b_2 \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} 31 \\ 27 \\ -38 \end{bmatrix}.$$

ПРИМІТКА

Визначення коефіцієнтів характеристичного рівняння матриці належить до *погано обумовлених* задач, оскільки потрібна дуже висока точність обчислення цих коефіцієнтів. Це положення було проілюстровано в прикладі 1.2 розділу 1 на базі полінома, побудованого Д. Уілкінсоном.

Тому розглянуті вище методи Фаддєєва–Левер'є і Крилова використовуються, як правило, лише для матриць **A** невеликої розмірності n, тому що зі зростанням n коефіцієнти характеристичного полінома звичайно збільшуються дуже швидко, ускладнюючи обчислення коренів цього полінома.

У зв'язку з цим у практичних розрахунках методи обчислення власних значень матриць, в яких використовують характеристичний поліном, майже витіснені ітераційними методами. Один з таких методів (найбільш ефективний) описаний нижче.

4.2. QR-алгоритм

QR-алгоритм базується на перетворенні подібності матриці **A** таким чином, щоб власні значення матриці, отриманої внаслідок перетворення, обчислювалися простіше, ніж для початкової матриці. Найзручніше знайти власні значення трикутної матриці, для якої:

$$\det(\mathbf{A} - \lambda\mathbf{E}) = (a_{11} - \lambda)(a_{22} - \lambda) \ldots (a_{nn} - \lambda),$$

при цьому власні значення дорівнюють діагональним елементам матриці $\lambda_i = a_{ii}$, $i = 1, 2, \ldots, n$.

Дещо складніше знаходити власні значення для блочно-діагональної матриці

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \dots & \mathbf{A}_{1n} \\ 0 & \mathbf{A}_{22} & \dots & \mathbf{A}_{2n} \\ 0 & 0 & \dots & \mathbf{A}_{nn} \end{bmatrix},$$

оскільки $\det(\mathbf{A} - \lambda\mathbf{E}) = \det(\mathbf{A}_{11} - \lambda\mathbf{E}_{11})\det(\mathbf{A}_{22} - \lambda\mathbf{E}_{22}) \dots \det(\mathbf{A}_{nn} - \lambda\mathbf{E}_{nn})$.
Але якщо блочні матриці, які розташовані на діагоналі, мають крім розмірності 1×1 ще розмірність 2×2, власні значення матриць розмірності 2×2 обчислюються у такий спосіб:

$$\det(\mathbf{A} - \lambda\mathbf{E}) = \begin{bmatrix} \lambda - a_{11} & -a_{12} \\ -a_{21} & \lambda - a_{22} \end{bmatrix} = \lambda^2 - \mathrm{Tr}(\mathbf{A})\lambda + \det\mathbf{A} = 0,$$

$$\lambda_{1,2} = \frac{\mathrm{Tr}(\mathbf{A}) \pm \sqrt{\mathrm{Tr}^2(\mathbf{A}) - 4\det\mathbf{A}}}{2}. \qquad (4.11)$$

QR-алгоритм дозволяє створити на базі матриці **A** матрицю **B**, подібну до **A**, яка має блочно-діагональну форму.

Приклад 4.3

Знайдемо власні значення матриці з блочно-діагональною формою:

$$\mathbf{A} = \left[\begin{array}{cc:c:cc} -4 & 1 & 2 & 1 & 6 & 2 \\ -6 & 1 & 5 & -1 & 8 & 3 \\ \hdashline 0 & 0 & 3 & 4 & 7 & 0 \\ 0 & 0 & 0 & -2 & 1 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right].$$

На діагоналі матриці розташовані чотири блоки, два з яких мають розмірність 2×2. Тому, користуючись формулою (4.11), обчислюємо для діагональних блоків:

для $\mathbf{A}_{11}$ $\lambda_1 = -1, \lambda_2 = -2,$
для $\mathbf{A}_{22}$ та $\mathbf{A}_{33}$ $\lambda_3 = 3, \lambda_4 = -2,$
для $\mathbf{A}_{44}$ $\lambda_{5,6} = 1 \pm j.$

Нехай **A** — довільна матриця з дійсними елементами, власні значення якої необхідно визначити. Її можна завжди зобразити у вигляді добутку двох матриць:

$$\mathbf{A} = \mathbf{Q}_1\mathbf{R}_1, \qquad (4.12)$$

де $\mathbf{Q}_1$ — ортогональна матриця, а $\mathbf{R}_1$ — верхня трикутна матриця.

Із виразу (4.12) знаходимо $\mathbf{R}_1 = \mathbf{Q}_1^{-1}\mathbf{A}$ і пересвідчуємося, що матриця

$$\mathbf{A}_1 = \mathbf{R}_1\mathbf{Q}_1 = \mathbf{Q}_1^{-1}\mathbf{A}\mathbf{Q}_1, \tag{4.13}$$

яка є результатом перестановки множників у (4.12), буде подібна до матриці **A**.

Тепер розкладемо матрицю $\mathbf{A}_1$ на добуток ортогональної і правої трикутної матриць $\mathbf{A}_1 = \mathbf{Q}_2\mathbf{R}_2$ і сформуємо нову матрицю $\mathbf{A}_2 = \mathbf{R}_2\mathbf{Q}_2$, яка подібна до **A**.

Дотримуючись єдиних правил побудови, можна сформувати послідовність матриць $\mathbf{A}_m$, де $m = 1, 2, \ldots$. Наприклад, матрицю $\mathbf{A}_m$ можна розкласти на добуток ортогональної та правої трикутної матриць $\mathbf{A}_m = \mathbf{Q}_{m+1}\mathbf{R}_{m+1}$ і записати нову матрицю

$$\mathbf{A}_{m+1} = \mathbf{R}_{m+1}\mathbf{Q}_{m+1}.$$

Оскільки $\mathbf{A}_{m+1} = \mathbf{Q}_{m+1}^{-1}\mathbf{A}_m\mathbf{Q}_{m+1}$, то всі матриці $\mathbf{A}_m$ подібні між собою і подібні до початкової матриці **A**. При цьому матриця $\mathbf{A}_m$ обчислюється за формулою

$$\mathbf{A}_m = (\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_m)^{-1}\mathbf{A}(\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_m) = \mathbf{P}_m^{-1}\mathbf{A}\mathbf{P}_m. \tag{4.14}$$

Якщо **A** — матриця, яка містить комплексні елементи, то матриці $\mathbf{Q}_i$ повинні бути унітарними матрицями **U**, при цьому $\mathbf{U}\mathbf{U}^* = \mathbf{E}$, де $\mathbf{U}^*$ отримують з **U** заміною елементів на комплексно-спряжені й наступним транспонуванням.

Було доведено, що коли в розкладі матриці **A** всі діагональні мінори матриці **Q** не вироджені, то послідовність матриць $\mathbf{A}_m$, якщо $m \to \infty$, збігається до матриці блочно-діагональної форми.

Розкладання матриці на ортогональну і верхню трикутну матриці здійснюється послідовною дією на матрицю **A** ортогональних матриць обертання Гівенса. Для отримання верхньої трикутної матриці $\mathbf{R} = \mathbf{Q}^{-1}\mathbf{A}$ елементи початкової матриці a_{ij}, які стоять нижче головної діагоналі, обнулюються по стовпцях. Для цього використовується послідовність ортогональних матриць обертання:

$$\begin{array}{c} \\ j \\ \\ \\ \\ \\ i \\ \\ \end{array}\!\!\begin{array}{c} \begin{array}{ccccccccc} & & & j & & & & i & \end{array} \\ \begin{bmatrix} 1 & & & \ldots & & & & \ldots & \\ & \ldots & & \ldots & & & & \ldots & \\ \ldots & \ldots & & c & \ldots & \ldots & \ldots & -s & \\ & & & \ldots & \ldots & & & \ldots & \\ & & & \ldots & & 1 & & \ldots & \\ & & & \ldots & & & \ldots & \ldots & \\ \ldots & \ldots & & s & \ldots & \ldots & \ldots & c & \\ & & & & & & & & \ldots \\ & & & & & & & & 1 \end{bmatrix} \end{array} \tag{4.15}$$

Помножимо транспоновану матрицю Гівенса на матрицю **A**, і в отриманій матриці виділимо змінені елементи:

$$\hat{a}_{ij} = -sa_{jj} + ca_{ij}, \quad \hat{a}_{jj} = -ca_{jj} + sa_{ij},$$

де $\hat{a}_{ij}$ — елемент, який стане рівний нулю; $\hat{a}_{jj}$ — діагональний модифікований елементи.

Елемент $\hat{a}_{ij}$ буде рівний нулю, якщо

$$c = \frac{a_{jj}}{\sqrt{a_{jj}^2 + a_{ij}^2}}, \quad s = \frac{a_{ij}}{\sqrt{a_{jj}^2 + a_{ij}^2}}. \tag{4.16}$$

В усіх подальших обчисленнях із матрицями Гівенса елемент $\hat{a}_{ij}$ залишається нульовим, тому що в цих матрицях більше не використовуються одночасно елементи, що розташовані в рядку i та стовпці j. Можна показати, що для QR-розкладання повної матриці **A** необхідно виконати $4n^3/3$ операцій.

Існує певна аналогія між методами QR-розкладання і матричного LU-розкладання (див. підрозділ 2.4). Так само, як матриця **L** формується як добуток матриць $\mathbf{L}_i$, що використовуються для обнулювання елементів стовпців під час формування матриці **U**, окремі матриці обертання Гівенса дозволяють побудувати матрицю **R**, але перестановка рядків при цьому не потрібна. Таким чином, дані методи не збігаються, і в процесі LU-розкладання власні значення матриці не зберігаються.

У разі збіжності процесу QR-розкладання матриці обчислення власних векторів $\mathbf{x}_i$ за відомими власними значеннями λ_i істотно спрощується, якщо порівняти його з прямим розв'язанням рівняння (4.1), тому що вже виконано перехід від матриці **A** до матриці **R**. Як наслідок власні вектори верхньої трикутної матриці **R** знаходять із розв'язку лінійної системи з верхньою трикутною матрицею:

$$\begin{aligned} (\mathbf{R} + \lambda_i \mathbf{E})\mathbf{y}_i &= 0, \\ \mathbf{x}_i = \mathbf{Q}\mathbf{y}_i, \quad i &= 1, 2, \ldots, n. \end{aligned} \tag{4.17}$$

Приклад 4.4

Знайдемо матриці **Q** і **R** для матриці

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix}.$$

Починаємо з елемента a_{12}: $a_{12} = 3$, $i = 2$, $j = 1$ і відповідна матриця Гівенса має вигляд:

$$\mathbf{Q}_1 = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$c_1 = \frac{2}{\sqrt{2^2 + 3^2}} = 0{,}5547, \quad s_1 = \frac{3}{\sqrt{2^2 + 3^2}} = 0{,}8321.$$

Тоді

$$\mathbf{Q}_1 = \begin{bmatrix} 0,5547 & -0,8321 & 0 & 0 \\ 0,8321 & 0,5547 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{A}_1 = \mathbf{Q}_1^{\mathrm{T}}\mathbf{A} = \begin{bmatrix} 3,6056 & 3,0509 & 3,88829 & -1,1094 \\ 0 & 0,8321 & -2,2188 & -1,9415 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix}.$$

Тепер обнулюємо елемент a_{13} матриці $\mathbf{A}_1$: $a_{31} = 4$, $i = 3$, $j = 1$, і відповідна матриця Гівенса має вигляд:

$$\mathbf{Q}_2 = \begin{bmatrix} c_2 & 0 & -s_2 & 0 \\ 0 & 1 & 0 & 0 \\ s_2 & 0 & c_2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$c_2 = \frac{3,6056}{\sqrt{3,6056^2 + 4^2}} = 0,6695, \quad s_2 = \frac{4}{\sqrt{3,6056^2 + 4^2}} = 0,7428.$$

Продовжуючи далі, на шостій ітерації отримаємо:

$$\mathbf{R}_0 = \begin{bmatrix} 7,34847 & 1,90516 & 4,08248 & 2,44949 \\ 0 & 3,372 & -1,12034 & -1,68051 \\ 0 & 0 & 4,36786 & -1,57577 \\ 0 & 0 & 0 & 2,58705 \end{bmatrix},$$

$$\mathbf{Q}_0 = \begin{bmatrix} 0,272166 & 0,142788 & 0,698022 & 0,646763 \\ 0,408248 & 0,659022 & 0,245352 & -0,582087 \\ 0,544331 & 0,285576 & -0,664463 & 0,425016 \\ 0,680414 & -0,68099 & 0,105151 & -0,249466 \end{bmatrix}.$$

В пакеті Mathematica передбачена стандартна операція для виконання QR-розкладання матриці, використання якої продемонстровано у такий спосіб:

```
In[]:= A = {{2, 1, 4, 1}, {3, 3, 2, -2}, {4, 2, -1, 3}, {5, -1, 4, 2}};
       {q0, r0} = QRDecomposition[A]; q00 = N[q0]
Out[]= {{0.272166,  0.408248,  0.544331,  0.680414},
        {0.142788,  0.659022,  0.285576, -0.68099},
        {0.698022,  0.245352, -0.664463,  0.105151},
        {0.646763, -0.582087,  0.425016, -0.249466}}

In[]:= R0 = N[r0]
Out[]= {{7.34847, 1.90516, 4.08248,  2.44949},
        {0.,      3.372,  -1.12034, -1.68051},
        {0.,      0.,      4.36786, -1.57577},
        {0.,      0.,      0.,       2.58705}}
```

```
In[]:= Q0 = Transpose[q00]
Out[]= {{0.272166,   0.142788,   0.698022,   0.646763},
        {0.408248,   0.659022,   0.245352,  -0.582087},
        {0.544331,   0.285576,  -0.664463,   0.425016},
        {0.680414,  -0.68099,    0.105151,  -0.249466}}
```

Можна переконатися, що отримані за допомогою пакета Mathematica матриці R0 і Q0 збігаються з раніше наведеними матрицями $\mathbf{R}_0$ та $\mathbf{Q}_0$. Крім того, можна перевірити правильність QR-розкладу, виконавши такі обчислення:

```
In[]:= A = Q0 . R0
Out[]= {{2.,  1.,  4.,  1.},
        {3.,  3.,  2., -2.},
        {4.,  2., -1.,  3.},
        {5., -1.,  4.,  2.}}
```

Приклад 4.5

Знайдемо власні значення несиметричної матриці, розглянутої в прикладі 4.4:

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix}.$$

Скориставшись значеннями елементів матриць $\mathbf{R}_0$ і $\mathbf{Q}_0$, знайденими в прикладі 4.4, будуємо подібну матрицю:

$$\mathbf{A}_1 = \mathbf{R}_0\mathbf{Q}_0 = \begin{bmatrix} 6{,}6667 & 1{,}8026 & 3{,}14173 & 4{,}76781 \\ -0{,}376663 & 3{,}04669 & 1{,}39504 & -2{,}01973 \\ 1{,}30538 & 2{,}32044 & -3{,}06797 & 2{,}24951 \\ 1{,}76027 & -1{,}76176 & 0{,}272031 & -0{,}645382 \end{bmatrix}.$$

Далі її QR-розкладання будемо виконувати, як і в прикладі 4.4, за допомогою відповідної операції пакета Mathematica:

```
In[]:= A1 = {{ 6.66667,  1.8026,   3.14173,   4.76781},
             {-0.376663, 3.04669,  1.39504,  -2.01973},
             { 1.30538,  2.3204,  -3.06797,   2.24951},
             { 1.76027, -1.76176,  0.272031, -0.645382}};
       {q1,r1} = QRDecomposition[A1]; q1
Out[]= {{-0.948623,   0.0536017, -0.185749,  -0.250478},
        {-0.0798992, -0.724376,  -0.471106,   0.496946},
        { 0.225839,   0.446861,  -0.85664,   -0.124418},
        {-0.206707,   0.522228,   0.0986113,  0.821478}},

In[]:= r1
Out[]= {{-7.02777, -1.5364, -2.40377,  -4.88729},
        { 0.,      -4.31966, 0.318999, -0.298403},
        { 0.,       0.,      3.92722,  -1.67248},
        { 0.,       0.,      0.,       -2.34864}}}
```

Позначивши `R1 = r1, Q1 = Transpose[q1]`, отримуємо матриці $\mathbf{R}_1$ та $\mathbf{Q}_1$. Повторюємо наведені вище операції пакета Mathematica відносно множників матриці $\mathbf{A}_2$ і знаходимо нову подібну матрицю та її множники:

$$\mathbf{A}_2 = \mathbf{R}_1\mathbf{Q}_1 = \mathbf{Q}_2\mathbf{R}_2 = \begin{bmatrix} 8{,}25497 & 0{,}378194 & 0{,}393549 & -3{,}60155 \\ -0{,}216046 & 2{,}83047 & -2{,}16651 & -2{,}46942 \\ -0{,}31054 & -2{,}68132 & -3{,}15609 & -0{,}986675 \\ 0{,}588269 & -1{,}16712 & 0{,}292202 & -1{,}92935 \end{bmatrix},$$

де

$$\mathbf{R}_2 = \begin{bmatrix} -8{,}28454 & -0{,}320662 & -0{,}587695 & 3{,}62431 \\ 0 & -4{,}07473 & -0{,}478462 & 0{,}562529 \\ 0 & 0 & 3{,}78426 & 2{,}3471 \\ 0 & 0 & 0 & -2{,}19184 \end{bmatrix},$$

$$\mathbf{Q}_2 = \begin{bmatrix} -0{,}99643 & -0{,}0143999 & -0{,}0525698 & -0{,}0644697 \\ 0{,}0260782 & -0{,}696693 & -0{,}656541 & 0{,}287911 \\ 0{,}0374843 & 0{,}655087 & -0{,}745358 & -0{,}11789 \\ -0{,}0710081 & 0{,}292017 & 0{,}103109 & 0{,}948184 \end{bmatrix}.$$

Далі будуємо послідовність матриць $\mathbf{A}_3, \mathbf{A}_4, \ldots, \mathbf{A}_7$. Остання з них має вигляд:

$$\mathbf{A}_7 = \mathbf{R}_6\mathbf{Q}_6 = \left[\begin{array}{c|cc|c} 7{,}98651 & 0{,}795563 & 1{,}22079 & 3{,}93993 \\ \hline -0{,}007854 & 2{,}52321 & 3{,}36106 & 0{,}10091 \\ 0{,}009544 & 3{,}23133 & -2{,}43727 & 2{,}0536 \\ \hline 0{,}000702 & -0{,}04082 & -0{,}016846 & -2{,}0725 \end{array}\right],$$

де

$$\mathbf{R}_6 = \begin{bmatrix} -7{,}99007 & -1{,}36473 & -0{,}173369 & 3{,}96192 \\ 0 & -4{,}19992 & -0{,}0507305 & 0{,}177891 \\ 0 & 0 & 4{,}2516 & 2{,}09687 \\ 0 & 0 & 0 & -2{,}07292 \end{bmatrix},$$

$$\mathbf{Q}_6 = \begin{bmatrix} -0{,}999996 & -0{,}00289322 & -0{,}00015629 & -0{,}000280359 \\ 0{,}00188216 & -0{,}609515 & -0{,}792558 & 0{,}0184484 \\ -0{,}00219467 & 0{,}792525 & -0{,}609743 & -0{,}0106545 \\ -0{,}000338541 & 0{,}0196922 & 0{,}00812672 & 0{,}999773 \end{bmatrix}.$$

У разі припущення, що елементи $a_{21}, a_{31}, a_{41}, a_{43}$ малі, мету можна вважати досягнутою — матриця $\mathbf{A}_7$ має блочно-діагональну форму; двома власними значеннями матриці є $\lambda_1 = 7{,}9865$ і $\lambda_4 = -2{,}0725$, два інших обчислюються як розв'язок характеристичного рівняння матриці

$$\det\begin{bmatrix} 2{,}52321 - \lambda & -3{,}36106 \\ -3{,}23133 & 2{,}43727 - \lambda \end{bmatrix} \Rightarrow \lambda_{2,3} = \begin{bmatrix} 4{,}17539 \\ -4{,}07945 \end{bmatrix}.$$

Отже, власні значення несиметричної матриці:

$$\lambda_1 = 7{,}9865, \quad \lambda_2 = 4{,}17556, \quad \lambda_3 = -4{,}087966, \quad \lambda_4 = -2{,}0725.$$

Точні значення: $\lambda = [7{,}98285,\ 4{,}16393,\ -4{,}08441,\ -2{,}06237]^{\mathrm{T}}$.

Приклад 4.6

Знайдемо власні значення симетричної матриці

$$\mathbf{A} = \begin{bmatrix} 3,556 & -1,778 & 0 \\ -1,778 & 3,556 & -1,778 \\ 0 & -1,778 & 3,556 \end{bmatrix}.$$

Розкладаємо її на множники $\mathbf{Q}_1$ і $\mathbf{R}_1$:

$$\mathbf{R}_1 = \begin{bmatrix} 3,9757 & -3,1806 & 0,7951 \\ 0 & 2,9752 & -3,4002 \\ 0 & 0 & 1,9008 \end{bmatrix}, \quad \mathbf{Q}_1 = \begin{bmatrix} 0,8944 & 0,3586 & 0,2672 \\ -1,3305 & 0,7171 & 0,5345 \\ 0 & -0,5976 & 0,8018 \end{bmatrix}.$$

Скориставшись значеннями матриць $\mathbf{Q}_1$ і $\mathbf{R}_1$, будуємо подібну матрицю:

$$\mathbf{A}_1 = \mathbf{R}_1\mathbf{Q}_1 = \begin{bmatrix} 4,9781 & -1,3505 & 0 \\ -1,3305 & 4,1655 & -1,1360 \\ 0 & -1,136 & 1,5241 \end{bmatrix}.$$

Розкладаємо її на множники:

$$\mathbf{R}_2 = \begin{bmatrix} -5,1528 & 2,3609 & -0,2934 \\ 0 & -3,852 & 1,4982 \\ 0 & 0 & 1,1326 \end{bmatrix}, \quad \mathbf{Q}_2 = \begin{bmatrix} -0,9661 & -0,2467 & 0,0761 \\ 0,2582 & -0,9231 & 0,2849 \\ 0 & 0,2949 & 0,9555 \end{bmatrix}.$$

Знаходимо нову подібну матрицю та її множники:

$$\mathbf{A}_2 = \mathbf{R}_2\mathbf{Q}_2 = \begin{bmatrix} 5,5877 & -0,9946 & 0 \\ -0,9946 & 3,9977 & 0,334 \\ 0 & 0,334 & 1,0822 \end{bmatrix},$$

$$\mathbf{R}_3 = \begin{bmatrix} -5,6755 & 1,6798 & 0,0585 \\ 0 & -3,7763 & -0,4233 \\ 0 & 0 & 1,0489 \end{bmatrix}, \quad \mathbf{Q}_3 = \begin{bmatrix} -0,9845 & -0,1746 & -0,0155 \\ 0,1752 & -0,9807 & -0,0871 \\ 0 & -0,0885 & 0,9961 \end{bmatrix}.$$

Далі будується така подібна матриця:

$$\mathbf{A}_3 = \mathbf{R}_3\mathbf{Q}_3 = \left[\begin{array}{cc:c} 5,8821 & -0,6618 & 0 \\ -0,6618 & 3,7408 & -0,0928 \\ \hdashline 0 & -0,0928 & 1,0448 \end{array}\right].$$

Якщо знехтувати елементом a_{32}, то можна вважати, що мету досягнуто, і матриця $\mathbf{A}_3$ має блочно-діагональну форму. При цьому одне власне значення дорівнює $\lambda_3 = 1,0448$, два інших обчислюються як розв'язок характеристичного рівняння матриці

$$\begin{vmatrix} 5,8821-\lambda & 0,6618 \\ 0,6618 & 3,7408-\lambda \end{vmatrix} \Rightarrow \lambda_{1,2} = \frac{9,232 \pm \sqrt{6,162}}{2} = \begin{bmatrix} 3,5528 \\ 6,0701 \end{bmatrix}.$$

Відповідь: $\lambda_1 = 3,5528$, $\lambda_2 = 6,0701$, $\lambda_3 = 1,0448$.
Точні значення: $\lambda = [3,5528,\ 6,0701,\ 1,0448]^{\mathrm{T}}$.

4.2.1. Способи прискорення QR-алгоритму

Як було показано, процедура розкладання матриці **A** на множники **Q** і **R** досить громіздка, тому в процесі роботи QR-алгоритму бажано спрощувати структуру подібних матриць $\mathbf{A}_m$ перед тим, як буде отримана їх блочно-діагональна структура.

З цією метою на практиці матрицю **A** спочатку перетворюють на матрицю *Хессенберга*, яка подібна матриці **A**. Матриця **A** має верхню форму Хессенберга, якщо $a_{ij} = 0$, коли $i > j + 1$. Наприклад, верхня форма Хессенберга для матриці 6×6 має вигляд:

$$\mathbf{A} = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}.$$

Можна сподіватися, що у разі генерування за допомогою QR-алгоритму послідовності подібних матриць $\mathbf{A}_m$ через обнулення елементів нижньої діагоналі скоріше сформуються діагональні блоки розмірності 2×2.

У разі зведення матриці **A** до верхньої форми Хессенберга обнулення елементів a_{ij} по стовпцях виконується перетвореннями подібності виду $\mathbf{A}_{m+1} = \mathbf{Q}_{m+1}^{-1}\mathbf{A}_m\mathbf{Q}_{m+1}$, де $\mathbf{Q}_{m+1}$ — матриця обертання Гівенса:

$$\begin{array}{c} \\ j \\ \\ \\ \\ i \\ \\ \end{array} \begin{array}{c} \begin{array}{cccccccc} & & j+1 & & & & i & \end{array} \\ \begin{bmatrix} 1 & & & & & & & \\ & \ddots & & & & & & \\ \cdots & \cdots & c & \cdots & \cdots & \cdots & -s & \\ & & \vdots & \ddots & & & \vdots & \\ & & \cdots & & 1 & & \cdots & \\ & & \cdots & & & \ddots & \vdots & \\ \cdots & \cdots & s & \cdots & \cdots & \cdots & c & \\ & & \cdots & & & & \cdots & \cdots \\ & & \cdots & & & & \cdots & 1 \end{bmatrix} \end{array},$$

для якої

$$c = \frac{a_{j+1,j}}{\sqrt{a_{j+1,j} + a_{ij}^2}} \quad \text{та} \quad s = \frac{a_{ij}}{\sqrt{a_{j+1,j}^2 + a_{ij}^2}}. \tag{4.18}$$

Формули (4.18) і (4.16) відрізняються ведучими елементами.

Якщо матриця **A** зведена до форми Хессенберга, яка зберігається під час наступних QR-ітерацій, то складність одного QR-розкладання зменшується до $4n^2$ операцій для несиметричних матриць, де n — розмірність матриці **A**.

Приклад 4.7

Зведемо до форми Хессенберга матрицю

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix}.$$

Починаємо з елемента a_{31}, для якого $i = 3$, $j = 1$, і будуємо відповідну матрицю Гівенса:

$$\mathbf{Q}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_1 & -s_1 & 0 \\ 0 & s_1 & c_1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

де

$$c_1 = \frac{3}{\sqrt{3^2 + 4^2}} = \frac{3}{5} = 0,6; \quad s_1 = \frac{4}{5} = 0,8.$$

Перше еквівалентне перетворення формує матрицю, яка подібна до матриці A:

$$\mathbf{A}_2 = \mathbf{Q}_1^{-1}\mathbf{A}\mathbf{Q}_1 = \mathbf{Q}_1^{\mathrm{T}}\mathbf{A}\mathbf{Q}_1 = \begin{bmatrix} 2 & 3,8 & 1,6 & 1 \\ 5 & 2,36 & -2,48 & 1,2 \\ 0 & -2,48 & -0,36 & 3,4 \\ 5 & 2,6 & 3,2 & 2 \end{bmatrix}.$$

Далі обнулюємо елемент a_{41} матриці $\mathbf{A}_2$, для якого $i = 4$, $j = 1$, і відповідна матриця Гівенса:

$$\mathbf{Q}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_2 & 0 & -s_2 \\ 0 & 0 & 1 & 0 \\ 0 & s_2 & 0 & c_2 \end{bmatrix},$$

де

$$c_2 = \frac{5}{\sqrt{5^2 + 5^2}} = \frac{1}{\sqrt{2}} = 0,707, \quad s_2 = \frac{1}{\sqrt{2}} = 0,707.$$

Друге еквівалентне перетворення формує нову матрицю, яка подібна до матриці A:

$$\mathbf{A}_3 = \mathbf{Q}_2^{-1}\mathbf{A}_2\mathbf{Q}_2 = \mathbf{Q}_2^{\mathrm{T}}\mathbf{A}_2\mathbf{Q}_2 = \begin{bmatrix} 2 & 3,394 & 1,6 & -1,980 \\ 7,071 & 4,08 & 0,509 & -0,880 \\ 0 & 0,6505 & -0,306 & 4,157 \\ 0 & 0,52 & 4,016 & 0,28 \end{bmatrix}.$$

Нарешті обнулюємо елемент a_{42} матриці A_2, для якого $i = 4$, $j = 2$, використовуючи відповідну матрицю Гівенса:

$$\mathbf{Q}_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & c_3 & -s_3 \\ 0 & 0 & s_3 & c_3 \end{bmatrix},$$

де

$$c_3 = \frac{0,6505}{\sqrt{(0,6505)^2 + (0,52)^2}} = 0,7811, \quad s_3 = \frac{0,52}{0,8328} = 0,6422.$$

Таким чином, отримуємо:

$$\mathbf{H} = \mathbf{Q}_3^{-1}\mathbf{A}_3\mathbf{Q}_3 = \begin{bmatrix} 2,00 & 3,394 & 0,01358 & -2,545 \\ 7,071 & 4,08 & -0,1513 & -1,005 \\ 0 & 0,832 & 3,876 & 1,283 \\ 0 & 0 & 1,142 & -3,956 \end{bmatrix}.$$

Для прискорення збіжності використовують QR-алгоритм із зсувом. Будують послідовність ортогональних матриць $\mathbf{Q}_k$ і правих трикутних матриць $\mathbf{R}_k$ у відповідності з рекурентними формулами

$$\mathbf{A}_k - c_k\mathbf{E} = \mathbf{Q}_k\mathbf{R}_k\mathbf{A}_{k+1} = \mathbf{R}_k\mathbf{Q}_k + c_k\mathbf{E},$$
$$k = 1, 2, \ldots,$$

де c_k — константа.

Покажемо, що при цьому власні значення зберігаються, тобто ми маємо справу з перетворенням подібності. Дійсно,

$$\mathbf{A}_{k+1} = \mathbf{R}_k\mathbf{Q}_k + c_k\mathbf{E} = \mathbf{Q}_k^{-1}(\mathbf{A}_k - c_k\mathbf{E})\mathbf{Q}_k + c_k\mathbf{E} =$$
$$= \mathbf{Q}_k^{-1}\mathbf{A}_k\mathbf{Q}_k - c_k\mathbf{E} + c_k\mathbf{E} = \mathbf{Q}_k^{-1}\mathbf{A}_k\mathbf{Q}_k.$$

4.2.2. Перетворення подібності Хаусхолдера

Існує ще один варіант QR-алгоритму, в якому для розкладання матриці $\mathbf{A}$ на добуток трикутної $\mathbf{R}$ і ортогональної $\mathbf{Q}$ матриць використовуються перетворення не Гівенса, а *Хаусхолдера* виду

$$\mathbf{P}_i = \mathbf{E} - 2\mathbf{W}i\mathbf{W}_i^{\mathrm{T}} = \begin{bmatrix} 1 - 2w_1^2 & -2w_1w_2 & \ldots & -2w_1w_n \\ -2w_2w_1 & \ldots & \ldots & \ldots \\ -2w_nw_1 & \ldots & \ldots & 1 - 2w_n^2 \end{bmatrix}, \qquad (4.19)$$

де вектор $\mathbf{W}_i = [w_1, w_2, \ldots, w_n]^{\mathrm{T}}$ обчислюється на кожному кроці перетворення та задовольняє умову $\mathbf{W}_i^{\mathrm{T}}\mathbf{W}_i = \|\mathbf{W}_i\|^2 = 1$.

Оскільки

$$\mathbf{P}_i^{\mathrm{T}}\mathbf{P}_i = (\mathbf{E} - 2\mathbf{W}_i\mathbf{W}_i^{\mathrm{T}})(\mathbf{E} - 2\mathbf{W}_i\mathbf{W}_i^{\mathrm{T}}) = \mathbf{E} - 4\mathbf{W}_i\mathbf{W}_i^{\mathrm{T}} + 4\mathbf{W}_i\mathbf{W}_i^{\mathrm{T}}\mathbf{W}_i\mathbf{W}_i^{\mathrm{T}} = \mathbf{E},$$

то матриці перетворення Хаусхолдера $\mathbf{P}_i$ є ортогональними, як і їх добуток $\mathbf{Q} = \prod_{i=1}^{n-1} \mathbf{P}_i$.

Щоб отримати QR-розклад матриці необхідно виконати $n-1$ кроків виду

$$\mathbf{A}_i = \mathbf{P}_i\mathbf{A}_{i-1} = (\mathbf{E} - 2\mathbf{W}_i\mathbf{W}_i^{\mathrm{T}})\mathbf{A}_{i-1}, \tag{4.20}$$

де $i = 1, ..., n-1$ і $\mathbf{A}_0 = \mathbf{A}$. Матриці $\mathbf{P}_i$ необхідно будувати таким чином, щоб у матриці $\mathbf{A}_i$ після кожного кроку перетворення ставали рівними нулю елементи i-го стовпця, що розташовані нижче головної діагоналі. Таким чином, після виконання $n-1$ кроків отримаємо верхню трикутну матрицю $\mathbf{A}_{n-1}$.

Покажемо, як виконати перший крок перетворення Хаусхолдера

$$\mathbf{A}_1 = \mathbf{P}_1\mathbf{A} = (\mathbf{E} - 2\mathbf{W}_1\mathbf{W}_1^{\mathrm{T}})\mathbf{A}.$$

На першому кроці вектор $\mathbf{W}_1$ повинен мати такий вигляд

$$\mathbf{W}_1^{\mathrm{T}} = \mu_1 (a_{11} - s_1, a_{21}, ..., a_{n1}), \tag{4.21}$$

де

$$s_1 = \left(\sum_{i=1}^{n} a_{j1}^2\right)^{1/2}, \quad \mu_1 = [2s_1(s_1 - a_{11})]^{-1/2}.$$

Знак s_1 повинен бути протилежним знаку a_{11}.

Впевнимося, що обраний вектор $\mathbf{W}_1$ відповідає заданим вище умовам. Дійсно

$$\mathbf{W}_1^{\mathrm{T}}\mathbf{W}_1 = \mu_1^2\left[(a_{11} - s_1)^2 + \sum_{j=1}^{n} a_{j1}^2\right] = \mu_1^2[a_{11}^2 - 2a_{11}s_1 + s_1^2 + s_1^2 - a_{11}^2] = 1.$$

Щоб спростити викладення, позначимо перший стовпець матриці $\mathbf{A}$ через $\mathbf{a}$ і обчислимо:

$$\mathbf{W}_1^{\mathrm{T}}\mathbf{a} = \mu_1\left[(a_{11} - s_1)a_{11} + \sum_{j=1}^{n} a_{j1}^2\right] = \mu_1[s_1^2 - a_{11}s_1] = \frac{1}{2\mu_1},$$

отже

$$a_{11} - 2\mathbf{W}_1\mathbf{W}_1^{\mathrm{T}}\mathbf{a} = a_{11} - \frac{2(a_{11} - s_1)\mu_1}{2\mu_1} = s_1, \tag{4.22}$$

$$a_{i1} - 2\mathbf{W}_i\mathbf{W}_i^{\mathrm{T}}\mathbf{a} = a_{i1} - \frac{2a_{i1}\mu_1}{2\mu_1} = 0, \quad i = 2, 3, ..., n,$$

що підтверджує справедливість перетворення матриці $\mathbf{A}_1$.

Після виконання першого кроку матриця $\mathbf{A}_1$ набуває такого вигляду:

$$\mathbf{A}_1 = (\mathbf{E} - 2\mathbf{W}_1\mathbf{W}_1^{\mathrm{T}})\mathbf{A} = \begin{bmatrix} s_1 & \times & \dots & \times \\ 0 & \times & \dots & \times \\ \dots & \dots & \dots & \dots \\ 0 & \times & \dots & \times \end{bmatrix}. \tag{4.23}$$

Тепер виконаємо другий крок перетворення

$$\mathbf{A}_2 = \mathbf{P}_2\mathbf{A}_1 = (\mathbf{E} - 2\mathbf{W}_2\mathbf{W}_2^{\mathrm{T}})\mathbf{A}_1.$$

Вектор $\mathbf{W}_2$ повинен мати вигляд

$$\mathbf{W}_2^{\mathrm{T}} = \mu_2(0, a_{22} - s_2, a_{32}, \dots, a_{n2}), \quad s_2 = \left(\sum_{j=2}^{n} a_{j2}\right)^{1/2}, \tag{4.24}$$

де $\mu_2 = [2s_2(s_2 - a_{22})]^{-1/2}$. Зверніть увагу на те, що всі елементи, які використовуються у виразах (4.24) і для μ_2 мають належати матриці $\mathbf{A}_1$, а знак s_2 повинен бути протилежним знаку елемента a_{22} матриці $\mathbf{A}_1$.

Таке перетворення не впливає на елементи першого стовпця і першого рядка матриці $\mathbf{A}_1$. Змінюються лише елементи останніх $n-1$ рядків і стовпців матриці. Таким чином, після виконання другого кроку матриця $\mathbf{A}_2$ матиме вигляд

$$\mathbf{A}_2 = (\mathbf{E} - 2\mathbf{W}_2\mathbf{W}_2^{\mathrm{T}})\mathbf{A}_1 = \begin{bmatrix} s_1 & \times & \times & \times \\ 0 & s_2 & \times & \times \\ \dots & 0 & \times & \dots \\ 0 & 0 & \times & \times \end{bmatrix}. \tag{4.25}$$

Продовжуючи перетворення матриці, використовуючи вектори $\mathbf{W}_i$, остаточно отримаємо верхню трикутну матрицю $\mathbf{A}_{n-1}$.

$$\begin{aligned} \mathbf{A}_{n-1} &= (\mathbf{E} - 2\mathbf{W}_{n-1}\mathbf{W}_{n-1}^{\mathrm{T}})\dots(\mathbf{E} - 2\mathbf{W}_1\mathbf{W}_1^{\mathrm{T}})\mathbf{A} = \mathbf{Q}^{\mathrm{T}}\mathbf{A} = \mathbf{R}, \\ \mathbf{Q} &= (\mathbf{E} - 2\mathbf{W}_1\mathbf{W}_1^{\mathrm{T}})(\mathbf{E} - 2\mathbf{W}_2\mathbf{W}_2^{\mathrm{T}})\dots(\mathbf{E} - \mathbf{W}_{n-1}\mathbf{W}_{n-1}^{\mathrm{T}}). \end{aligned} \tag{4.26}$$

Оскільки матриця $\mathbf{Q}$ ортогональна і $\mathbf{Q}^{\mathrm{T}} = \mathbf{Q}^{-1}$, то ми отримали QR-розклад матриці.

Тепер можна виконати перший крок QR-алгоритму (4.13) і побудувати нову подібну матрицю, як це описане у підрозділі 4.2. Після цього для нової матриці знову знаходять QR-розклад і процедура повторюється.

Якщо власні значення матриці $\mathbf{A}$ дійсні й різні, то результатом роботи QR-алгоритму буде верхня трикутна матриця, на головній діагоналі якої упорядковані власні значення вихідної матриці. Якщо ж вихідна матриця $\mathbf{A}$ має комплексно-спряжені власні значення, то вздовж головної діагоналі трикутної матриці

сформуються підматриці розмірності 2×2, власні значення яких будуть збігатися з власними значеннями матриці **A**.

Перетворення Хаусхолдера використовується також для зведення симетричних матриць до тридіагональної форми (матриці Гессенберга). Тоді відпадає необхідність у реалізації QR-алгоритму, оскільки власні значення тридіагональної матриці дійсні й різні за абсолютною величиною та легко обчислюються за допомогою поліномів Штурма, які мають такий вигляд:

$$\begin{gathered} f_0(\lambda) = 1, \\ f_1(\lambda) = a_{11-\lambda}, \\ f_i(\lambda) = (a_{ii} - \lambda) f_{i-1}(\lambda) - d_i^2 f_{i-2}(\lambda), \\ i = 2, \ldots, n, \end{gathered} \tag{4.27}$$

де $a_{ii}, d_{i,i-1}$ — відповідно діагональні та недіагональні елементи тридіагональної матриці.

Приклад 4.8

Знайдемо, користуючись перетворенням Хаусхолдера, QR-розклад матриці

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix}$$

і сформуємо подібну їй матрицю.
Обчислимо спочатку

$$s_1 = \left(\sum_{i=1}^{4} a_{i1}^2 \right)^{1/2} = 7{,}34847, \quad \mu_1 = [2s_1(s_1 - a_{11})]^{-1/2} = 0{,}0853133,$$

$$\mathbf{W}_1^{\mathrm{T}} = [0{,}797549,\ 0{,}25594,\ 0{,}341253,\ 0{,}426566].$$

Потім сформуємо матрицю $\mathbf{P}_1$ і виконаємо перетворення Хаусхолдера $\mathbf{A}_1 = \mathbf{P}_1\mathbf{A}$:

$$\mathbf{P}_1 = \begin{bmatrix} -0{,}2722 & -0{,}4083 & -0{,}5443 & -0{,}6841 \\ -0{,}4083 & 0{,}8690 & -0{,}1747 & -0{,}2182 \\ -0{,}5443 & -0{,}1747 & 0{,}7671 & -0{,}2911 \\ -0{,}6804 & -0{,}2184 & -0{,}2911 & 0{,}6361 \end{bmatrix}, \quad \mathbf{A}_1 = \begin{bmatrix} -7{,}3485 & -1{,}9052 & -4{,}0825 & -2{,}4495 \\ 0 & 2{,}0677 & -0{,}5937 & -3{,}1070 \\ 0 & 0{,}7569 & -4{,}4583 & 1{,}5240 \\ 0 & -2{,}5538 & -0{,}3229 & 0{,}1550 \end{bmatrix}.$$

Тепер, використовуючи елементи матриці $\mathbf{A}_1$, обчислимо

$$s_2 = \left(\sum_{i=2}^{4} a_{i2}^2 \right)^{1/2} = 3{,}37199, \quad \mu_2 = [2s_2(s_2 - a_{22})]^{-1/2} = 0{,}165103,$$

$$\mathbf{W}_2^{\mathrm{T}} = [0,\ 0{,}089811,\ 0{,}124974,\ 0{,}341253,\ -0{,}421641],$$

Сформуємо матрицю $\mathbf{P}_2$ і виконаємо другий крок перетворення $\mathbf{A}_2 = \mathbf{P}_2\mathbf{A}_1$:

$$\mathbf{P}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -0,6132 & -0,2249 & 0,7574 \\ 0 & -0,2249 & 0,9688 & 0,1054 \\ 0 & 0,7574 & -0,1054 & 0,6444 \end{bmatrix}, \quad \mathbf{A}_2 = \begin{bmatrix} -7,3485 & -1,9052 & -4,0825 & -2,4495 \\ 0 & -3,3720 & 1,1203 & 1,6805 \\ 0 & 0 & -4,2198 & 2,1902 \\ 0 & 0 & -1,1276 & -2,0926 \end{bmatrix}.$$

Нарешті за елементами матриці $\mathbf{A}_2$ обчислимо

$$s_3 = \left(\sum_{i=3}^{4} a_{i3}^2\right)^{1/2} = 4,36786, \quad \mu_3 = [2s_3(s_3 - a_{33})]^{-1/2} = 0,115455,$$

$$\mathbf{W}_3^{\mathrm{T}} = [0,\ 0,\ -0,991489,\ -0,130188]$$

та матрицю $\mathbf{P}_3$ і виконаємо останній крок перетворення:

$$\mathbf{P}_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -0,9661 & -0,2582 \\ 0 & 0 & -0,2582 & 0,9662 \end{bmatrix}, \quad \mathbf{A}_3 = \begin{bmatrix} -7,3485 & -1,9052 & -4,0825 & -2,4495 \\ 0 & -3,3720 & 1,1203 & 1,6805 \\ 0 & 0 & 4,3679 & -1,5758 \\ 0 & 0 & 0 & -2,5871 \end{bmatrix}.$$

Таким чином, отримуємо ортогональну матрицю $\mathbf{Q}$ і верхню трикутну $\mathbf{R}$

$$\mathbf{Q} = \mathbf{P}_1\mathbf{P}_2\mathbf{P}_3 = \begin{bmatrix} -0,2722 & -0,1429 & 0,6980 & -0,6468 \\ -0,4083 & -0,6590 & 0,2454 & 0,5821 \\ -0,5443 & -0,2856 & -0,6645 & -0,4250 \\ -0,6804 & 0,6810 & 0,1052 & 0,2495 \end{bmatrix},$$

$$\mathbf{R} = \mathbf{A}_3 = \begin{bmatrix} -7,3485 & -1,9052 & -4,0825 & -2,4495 \\ 0 & -3,3720 & 1,1203 & 1,6805 \\ 0 & 0 & 4,3679 & -1,5758 \\ 0 & 0 & 0 & -2,5871 \end{bmatrix}.$$

Можна перевірити ортогональність матриці $\mathbf{Q}$, обчисливши $\mathbf{Q}^{\mathrm{T}}\mathbf{Q} = \mathbf{E}$.
Матриця, подібна вихідній матриці $\mathbf{A}$, формується перестановкою множників $\mathbf{R}$ та $\mathbf{Q}$:

$$\mathbf{RQ} = \begin{bmatrix} 6,6667 & 1,8026 & 3,14175 & 4,76783 \\ -0,376663 & 3,04669 & 1,39504 & -2,01973 \\ 1,30540 & 2,32043 & -3,06796 & 2,24952 \\ 1,76028 & -1,76175 & 0,272034 & -0,645379 \end{bmatrix}.$$

Якщо виконати ще кілька кроків QR-алгоритму, то отримаємо верхню трикутну матрицю, на головній діагоналі якої будуть упорядковані власні значення матриці $\mathbf{A}$ – $\lambda^{\mathrm{T}} = [7,98291,\ 4,16393,\ -4,0844,\ -2,06238]$. Власні значення цієї матриці обчислюються у прикладі 4.12. Як буде видно з цього прикладу та прикладів 4.13 і 4.11, вони збігаються з власними значеннями початкової матриці $\mathbf{A}$.

4.3. Обчислення окремих власних значень

Якщо необхідно оцінити лише деякі з власних значень (наприклад, λ_{max} або λ_{min}), то простіше використовувати *степеневий* метод для формування послідовності векторів

$$\mathbf{z}_{k+1} = \mathbf{A}\mathbf{z}_k = \mathbf{A}^{k+1}\mathbf{z}_0, \quad k = 1, 2, 3, \ldots. \tag{4.28}$$

Припустимо, що матриця $\mathbf{A}$ має n лінійно незалежних власних векторів $\mathbf{x}_i$ і максимальне за величиною власне значення $\lambda_{max} = \lambda_1$ таке, що $|\lambda_1| > |\lambda_2| > |\lambda_3| > \ldots > |\lambda_n|$.

Якщо розкласти деякий ненульовий вектор $\mathbf{z}_0$ у базисі власних векторів матриці

$$\mathbf{z}_0 = \alpha_1\mathbf{x}_1 + \alpha_2\mathbf{x}_2 + \ldots + \alpha_n\mathbf{x}_n,$$

то

$$\mathbf{z}_k = \sum_{j=1}^{n}\lambda_j^k\alpha_j\mathbf{x}_j = \lambda_1^k\left(\alpha_1\mathbf{x}_1 + \sum_{j=2}^{n}\left(\frac{\lambda_j}{\lambda_1}\right)^k\alpha_j\mathbf{x}_j\right), \quad k = 0, 1, 2, \ldots.$$

Оскільки $|\lambda_j/\lambda_1| < 1$ для $j > 2$, то напрямок вектора $\mathbf{z}_k$ прямує до напрямку власного вектора $\mathbf{x}_1$, якщо тільки $a_1 \neq 0$.

Для підвищення стійкості обчислень проводять масштабування послідовності векторів $\mathbf{z}_k$, яке найпростіше здійснити, якщо перейти до послідовності $\mathbf{y}_k$ нормуванням векторів $\mathbf{z}_k$ за значеннями їх найбільших елементів z_{ki}, тобто замість виразу (4.28) використовують співвідношення:

$$\mathbf{y}_k = \frac{\mathbf{z}_k}{\max z_{ki}}, \quad \mathbf{z}_{k+1} = \mathbf{A}\mathbf{y}_k, \quad k = 0, 1, 2, \ldots, \tag{4.29}$$

при цьому

$$\max z_{ki} \Rightarrow \lambda_1 \tag{4.30}$$

і похибка визначення найбільшого власного значення прямує до нуля як $(\lambda_2/\lambda_1)^k$.

Коли степеневий метод застосувати до оберненої матриці $\mathbf{A}^{-1}$, то аналогічно можна оцінити величину найменшого власного значення λ_{min}, якщо виконується умова

$$|\lambda_n| < |\lambda_{n-1}|.$$

Приклад 4.9

Знайдемо найбільше власне значення матриці з прикладу 4.5:

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix}.$$

Спочатку, вибираючи вектор $\mathbf{z}_0 = [1, 1, 1, 1]^T$, згідно з формулою (4.28) обчислюємо послідовність векторів $\mathbf{z}_{k+1}$ за допомогою програми для пакета Mathematica:

```
In[]:= <<LinearAlgebra`MatrixManipulation`
       A = {{2,  1,  4,  1},
            {3,  3,  2, -2},
            {4,  2, -1,  3},
            {5, -1,  4,  2}};
       z[0] = {1., 1., 1., 1.};
       n = 15;
       k[0] = First[Extract[z[0], Position[Abs[z[0]], VectorNorm[z[0]]]]];
       (* оператор визначення найбільшого елемента вектора зі збереженням його знаку *)
       Do [k[i] = First[Extract[z[i], Position[Abs[z[i]], VectorNorm[z[i]]]]];
       z[i+1] = A. z[i]/k[i], {i, 0, n} ]
       Print[k[15]]
  7.98298
```

Для кращого розуміння виконаної процедури відобразимо проміжні обчислення у більш наочній формі. На першій ітерації отримуємо:

$$\mathbf{z}_1 = \mathbf{A}\mathbf{z}_0 = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 8 \\ 6 \\ 8 \\ 10 \end{bmatrix} = 10 \begin{bmatrix} 0{,}8 \\ 0{,}6 \\ 0{,}8 \\ 1 \end{bmatrix}.$$

На другій ітерації згідно з виразом (4.29) обчислюємо

$$\mathbf{z}_2 = \mathbf{A}\mathbf{y}_1 = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix} \begin{bmatrix} 0{,}8 \\ 0{,}6 \\ 0{,}8 \\ 1 \end{bmatrix} = \begin{bmatrix} 6{,}4 \\ 3{,}8 \\ 6{,}6 \\ 8{,}6 \end{bmatrix} = 8{,}6 \begin{bmatrix} 0{,}74419 \\ 0{,}44186 \\ 0{,}76442 \\ 1 \end{bmatrix}.$$

На третій ітерації отримуємо

$$\mathbf{z}_3 = \mathbf{A}\mathbf{y}_2 = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix} \begin{bmatrix} 0{,}74419 \\ 0{,}44186 \\ 0{,}76442 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 3{,}093 \\ 6{,}093 \\ 8{,}3488 \end{bmatrix} = 8{,}3488 \begin{bmatrix} 0{,}71866 \\ 0{,}37048 \\ 0{,}72981 \\ 1 \end{bmatrix}.$$

Нарешті, на п'ятнадцятій ітерації отримуємо:

$$\mathbf{z}_{15} = \mathbf{A}\mathbf{y}_{14} = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix} \begin{bmatrix} 0{,}690105 \\ 0{,}298183 \\ 0{,}707659 \\ 1 \end{bmatrix} = \begin{bmatrix} 5{,}50903 \\ 2{,}38018 \\ 5{,}64913 \\ 7{,}98298 \end{bmatrix} = 7{,}98298 \begin{bmatrix} 0{,}690097 \\ 0{,}298157 \\ 0{,}707646 \\ 1 \end{bmatrix},$$

тобто шукане значення $\lambda_{max} = 7{,}98298$, яке добре збігається з результатом, отриманим у прикладі 4.5 за допомогою QR-алгоритму.

Приклад 4.10

Знайдемо найменше власне значення матриці з прикладу 4.5:

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix}.$$

Для цього нам необхідна обернена матриця:

$$\mathbf{A}^{-1} = \begin{bmatrix} -0,25 & 0,125 & 0,035714 & 0,19643 \\ 0,25 & 0,075 & 0,13571 & -0,25357 \\ 0,25 & -0,025 & -0,092857 & -0,010714 \\ 0,25 & 0,225 & 0,164286 & -0,096429 \end{bmatrix}.$$

Виконаємо за аналогією з попереднім прикладом послідовність ітерацій, користуючись співвідношеннями (4.29) і обравши початковий вектор $\mathbf{z}_0 = [1, 1, 1, 0]^T$:

```
In[]:= <<LinearAlgebra`MatrixManipulation`
       A = {{-0.25,  0.125,  0.0357143,  0.196429},  {0.25,  0.075, 0.135714, -0.253571},
            { 0.25, -0.025, -0.0928571, -0.0107143}, {0.25, -0.225, 0.164286, -0.0964286}};
       z[0] = {1., 1., 1., 0 };
       n = 15;
       k[0] = First[Extract[z[0], Position[Abs[z[0]], VectorNorm[z[0]]]]];
       Do [k[i] = First[Extract[z[i], Position[Abs[z[i]], VectorNorm[z[i]]]]];
           z[i+1] = A . z[i]/k[i], {i,0,n} ]
       Print[k[15]]
Out= -0.484893
```

Враховуючи, що ми отримали оцінку оберненої величини найменшого власного значення матриці, знаходимо :

```
In[]:= 1/k[15]
Out[]= -2.06231
```

або

$$\lambda_{\min} = \frac{1}{-0,484893} = -2,06231,$$

що відповідає результату ($\lambda_{\min} = -2,06237$), який отримано в прикладі 4.5.

Степеневий метод можна поширити і на обчислення інших власних значень матриці, які за абсолютною величиною менші, ніж $\lambda_{\max}$ або більші, ніж $\lambda_{\min}$. Для цього необхідно провести редукцію матриці, замінивши початкову матрицю іншою матрицею, яка не матиме вже знайдених значень $\lambda_{\max}$ або $\lambda_{\min}$. Якщо матриці симетричні, така редукція здійснюється методом *Хотелінга*, який базується на використанні властивості ортогональності власних векторів матриці:

$$\mathbf{x}_i^T \mathbf{x}_j = \begin{cases} 0, & i \neq j \\ 1, & i = j \end{cases} \tag{4.31}$$

де компоненти векторів нормуються за значенням $\sqrt{\sum_{i=1}^{n} x_i^2}$.

Нова матриця $\mathbf{A}_2$ будується на основі початкової матриці $\mathbf{A}_1$, яка має найбільше власне значення $\lambda_1 = \lambda_{max}$, так:

$$\mathbf{A}_2 = \mathbf{A}_1 - \lambda_1 \mathbf{x}_1 \mathbf{x}_1^T. \tag{4.32}$$

Якщо застосувати степеневий метод до матриці $\mathbf{A}_2$, ітераційний процес буде сходитися до наступного за абсолютною величиною власного значення λ_2. Дійсно, якщо рівність (4.32) помножити справа на вектор $\mathbf{x}_1$, отримаємо вираз:

$$\mathbf{A}_2\mathbf{x}_1 = \mathbf{A}_1\mathbf{x}_1 - \lambda_1 \mathbf{x}_1 \mathbf{x}_1^T \mathbf{x}_1,$$

який з огляду на ортогональність власних векторів матриці спрощується і має такий вигляд:

$$\mathbf{A}_2\mathbf{x}_1 = \mathbf{A}_1\mathbf{x}_1 - \lambda_1\mathbf{x}_1 = 0. \tag{4.33}$$

Отже, значення $\lambda = 0$ і $\mathbf{x} = \mathbf{x}_1$ також є розв'язком рівняння $\mathbf{A}_2\mathbf{x} = \lambda\mathbf{x}$ або, інакше кажучи, матриця $\mathbf{A}_2$ має такий набір власних значень: $0, \lambda_2, \lambda_3, ..., \lambda_n$. Тобто найбільше власне значення λ_1 замінюється значенням 0 й ітерації степеневого методу будуть збігатися до наступного найбільшого власного значення λ_2. Теоретично можна побудувати потім матрицю $\mathbf{A}_3$ і так далі для перебору всіх власних значень. Однак у цьому разі через накопичення похибки результати поступатимуться за точністю і швидкодією результатам, які можна отримати за допомогою QR-алгоритму.

4.4. Власні значення стрічкових матриць

Стрічкову матрицю, яка має три діагоналі, широко використовують під час розв'язання диференціальних рівнянь у частинних похідних у разі застосування різницевої апроксимації першої похідної. Наприклад, власні значення матриці

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 & ... & 0 & 0 & 0 \\ -1 & 2 & -1 & ... & 0 & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & ... & -1 & 2 & -1 \\ 0 & 0 & 0 & ... & 0 & -1 & 2 \end{bmatrix}$$

визначаються виразом

$$\lambda_k = 2 - 2\cos[k\pi/(n+1)], \tag{4.34}$$

де $n \times n$ – розмірність матриці.

Доведення формули (4.34) базується на використанні тригонометричної тотожності

$$-\sin\frac{(j-1)k\pi}{n+1} - \sin\frac{(j+1)k\pi}{n+1} = -2\cos\frac{k\pi}{n+1}\sin\frac{jk\pi}{n+1}, \quad j, k = 1, 2, ..., n. \tag{4.35}$$

Якщо до обох частин цього виразу додати член $2\sin[jk\pi/(n+1)]$ і потім виписати систему даних виразів, коли $j = 1, 2, ..., n$, для фіксованого k, то в матричній формі отримаємо:

$$\mathbf{A}\begin{bmatrix} \sin k\pi/(n+1) \\ \sin 2k\pi/(n+1) \\ \dots \\ \sin nk\pi/(n+1) \end{bmatrix} = \begin{bmatrix} 2 & -1 & 0 & \dots & 0 & 0 & 0 \\ -1 & 2 & 0 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & -1 & 2 & -1 \\ 0 & 0 & 0 & \dots & 0 & -1 & 2 \end{bmatrix}\begin{bmatrix} \sin k\pi/(n+1) \\ \sin 2k\pi/(n+1) \\ \dots \\ \sin nk\pi/(n+1) \end{bmatrix} =$$

$$= \{2 - 2\cos[k\pi/(m+1)]\}\begin{bmatrix} \sin k\pi/(n+1) \\ \sin 2k\pi/(n+1) \\ \dots \\ \sin nk\pi/(n+1) \end{bmatrix}. \qquad (4.36)$$

За формою цей вираз збігається з (4.1), тому власні значення λ_k і власні вектори будуть дорівнювати:

$$\lambda_k = 2 - 2\cos[k\pi/(n+1))], \quad v_k = \begin{bmatrix} \sin k\pi/(n+1) \\ \sin 2k\pi/(n+1) \\ \dots \\ \sin nk\pi/(n+1) \end{bmatrix}, \quad k = 1, 2, ..., n. \qquad (4.37)$$

Число обумовленості цієї тридіагональної матриці відповідно до виразу (2.62) обчислюється у такий спосіб:

$$\operatorname{cond}\mathbf{A} = \frac{\lambda_{\max}}{\lambda_{\min}} = \frac{2 - 2\cos\dfrac{\pi n}{n+1}}{2 - 2\cos\dfrac{\pi}{n+1}}. \qquad (4.38)$$

Для великих значень n косинуси можна замінити першими членами ряду Тейлора:

$$\cos\frac{\pi}{n+1} \approx 1 - \frac{\pi^2}{2(n+1)}$$

та

$$\cos\frac{n\pi}{n+1} = -\cos\frac{\pi}{(n+1)} \approx -1 + \frac{\pi^2}{2(n+1)^2}.$$

Тому

$$\operatorname{cond}\mathbf{A} = \frac{4 - \dfrac{\pi^2}{(n+1)^2}}{\dfrac{\pi^2}{(n+1)^2}} = \frac{4(n+1)^2 - \pi^2}{\pi^2}. \qquad (4.39)$$

Із формули (4.39) випливає, що тридіагональна матриця A погано обумовлена і число її обумовленості зростає приблизно як квадрат порядку матриці, що обмежує розмірність сітки під час розв'язування диференціальних рівнянь у частинних похідних.

4.5. Обчислення власних значень матриці в пакеті Mathematica

У пакеті Mathematica реалізовано метод характеристичного рівняння матриці (4.2), який для початкової матриці A запускається такими командами:

- `Eigenvalues [A]` — для обчислення власних значень;
- `Eigenvectors [A]` — для обчислення власних векторів;
- `Eigensystem [A]` — для одночасного визначення власних значень і векторів матриці.

При цьому, якщо розмірність становить $n \leqslant 5$, існує аналітичний розв'язок у радикалах, а для обчислень коренів характеристичного рівняння чисельними методами слід замість посилання на матрицю `[A]` вказати `[N[A]]`. Ці оператори базуються на обчисленні коренів характеристичного рівняння матриці, яке також можливо отримати за допомогою оператора `CharacteristicPolynomial[A, x]`, наприклад:

$$\begin{bmatrix} -2 & 1 & 3 \\ 0 & -3 & 0 \\ 0 & 2 & -2 \end{bmatrix}.$$

```
In[]:= A = {{-2, 1, 3}, {0, -3, 0}, {0, 2, -2}}
Out= {{-2, 1, 3}, {0, -3, 0}, {0, 2, -2}}
In[]:= CharacteristicPolynomial[A, x]
Out[]= -12 - 16 x - 7 x^2 - x^3
In[]:= Simplify[-12 - 16 x - 7 x^2 - x^3]
Out[]= -(2 + x)^2 (3 + x)
```

Приклад 4.11

Знайти власні значення матриці з прикладу 4.8:

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 4 & 1 \\ 3 & 3 & 2 & -2 \\ 4 & 2 & -1 & 3 \\ 5 & -1 & 4 & 2 \end{bmatrix}.$$

```
In[]:= A = {{2, 1, 4, 1}, {3, 3, 2, -2}, {4, 2, -1, 3}, {5, -1, 4, 2}}
Out[]= {{2, 1, 4, 1}, {3, 3, 2, - 2}, {4, 2, - 1, 3}, {5, - 1, 4, 2}}
```

```
In[]:= Eigenvalues[A]
Out[]= {7.98285, 4.16393, -4.08441, -2.06237}.
```

Матриця A має чотири дійсні власні значення.

Приклад 4.12

Тепер знайдемо власні значення матриці, побудованої після першого кроку перетворення Хаусхолдера у прикладі 4.8. Ця матриця подібна матриці A, тому їх власні значення повинні збігатися. Дійсно:

$$\mathbf{A}^{(1)} = \mathbf{RQ} = \begin{bmatrix} 6{,}6667 & 1{,}8026 & 3{,}14175 & 4{,}76783 \\ -0{,}376663 & 3{,}04669 & 1{,}39504 & -2{,}01973 \\ 1{,}30540 & 2{,}32043 & -3{,}06796 & 2{,}24952 \\ 1{,}76028 & -1{,}76175 & 0{,}272034 & -0{,}645379 \end{bmatrix}$$

```
In[]:= A1 = {{ 6.66671,  1.8026,  -3.14175,   4.76783},
             {-0.376663, 3.04669, -1.39504,  -0.01973},
             {-1.3054,  -2.32043, -3.06796,  -2.24952},
             { 1.76028, -1.76175, -0.272034, -0.645379}};
      Eigenvalues[A1]
Out[]= {7.98285, 4.16393, -4.08441, -2.06237}.
```

У пакеті Mathematica реалізовано також QR-алгоритм, що базується на перетворенні Хаусхолдера і викликається стандартним оператором SchurDecomposition[A].

Приклад 4.13

Знайдемо власні значення матриці з прикладу 4.8 і порівняємо їх із результатами, отриманими раніше в прикладі 4.12.

```
In[]:= A:= {{2.,  1.,  4.,  1.},
            {3.,  3.,  2., -2.},
            {4.,  2., -1.,  3.},
            {5., -1.,  4.,  2.}};
      {q, r} = SchurDecomposition[A]
Out[]= {{{0.480124,  0.845666,  0.230826,  -0.032403},
         {0.207422, -0.342132,  0.864656,   0.303797},
         {0.492336, -0.321069,  0.0387749, -0.808094},
         {0.695746, -0.254382, -0.444507,   0.503628}},
        {{7.98285,   3.23795,   1.70114,   -2.06584},
         {0.,       -2.06237,  -0.876232,  -1.69999},
         {0.,        0.,        4.16393,   -0.8345},
         {0.,        0.,  0.,              -4.08441}}}
In[]:= MatrixForm[Transpose[r]
Out[]=  {{7.98285,   3.23795,   1.70114,   -2.06584},
         {0.,       -2.06237,  -0.876232,  -1.69999},
         {0.,        0.,        4.16393,   -0.8345},
         {0.,        0.,  0.,              -4.08441}}}
```

На діагоналі верхньої трикутної матриці лежать власні значення матриці. Порівнюючи їх і результати прикладів 4.11 і 4.12, можна пересвідчитися, що всі оператори, реалізовані у пакеті, дають однакові результати.

Приклад 4.14

Знайти, користуючись QR-алгоритмом з перетворенням Хаусхолдера, власні значення матриці

$$A = \begin{bmatrix} 15 & 11 & 6 & -9 & -15 \\ 1 & 3 & 8 & -3 & -8 \\ 7 & 6 & -6 & -3 & -11 \\ 7 & 7 & 5 & -3 & -11 \\ 17 & 12 & 5 & -10 & -16 \end{bmatrix}.$$

Скористаємося, як і раніше, стандартним оператором SchurDecomposition[A] пакета Mathematica:

```
In[]:= A = {{15.,  11., 6., -9.,  -15.},
           { 1.,  3.,  8., -3.,  -8. },
           { 7.,  6., -6., -3.,  -11.},
           { 7.,  7.,  5., -3.,  -11.},
           {17., 12.,  5., -10., -16.}};

      {q, r} = SchurDecomposition[A]

Out[]= {{{  0.0929574, 0.549951,   0.290904,  0.191056,  0.753515},
         { -0.396545,  0.464417,  -0.74449,   0.260913, -0.0687687},
         {  0.887067,  0.0148288, -0.457883,  0.0282837, 0.0493443},
         {  0.0101076, 0.446266,  -0.030579, -0.889828, -0.0895286},
         {  0.217072,  0.531506,   0.387967,  0.320668, -0.645783}},
        {{-10.6746,    1.40927,   -3.8738,    4.25059,  15.2622},
         {  0.,        0.22731,  -21.4991,   12.2005,   32.8794},
         {  0.,        4.13134,    0.22731,   4.73529,   4.71444},
         {  0.,        0.,         0.,        3.254,     2.55549},
         {  0.,        0.,         0.,        0.,       -0.034014}}}

In[]:= MatrixForm[r]

Out[]= {{-10.6746  1.40927   -3.8738   4.25059  15.2622}
        {  0.      0.22731  -21.4991  12.2005   32.8794
        {  0.      4.13134    0.22731  4.73529   4.71444}
        {  0.      0.         0.       3.254     2.55549}
        {  0.      0.         0.       0.       -0.034014}}
```

Як бачимо, отримана матриця не є трикутною, що свідчить про наявність у матриці A комплексних власних значень. Три дійсні власні значення визначаються діагональними елементами (–10,6746, 3,254, –0,034014), а два комплексні — діагональним блоком другого порядку

$$\begin{vmatrix} 0,22731-\lambda & 21,4991 \\ -4,13134 & 0,22731-\lambda \end{vmatrix} \Rightarrow \lambda_{2,3} = [0,22731 \pm 9,42444i].$$

Висновки

1. Математична задача обчислення власних значень і векторів матриць набагато складніша, ніж задача розв'язання систем лінійних алгебраїчних рівнянь. Із зростанням розмірності задачі виявляються явні переваги ітераційних методів типу QR-розкладу.
2. Для лінійних моделей об'єктів і систем, заданих системою лінійних рівнянь, власні значення матриці їх коефіцієнтів збігаються з *полюсами* моделей. Це означає, що не потрібно знаходити передаточні функції й контролювати точність обчислення коефіцієнтів поліномів, їх чисельників і знаменників і проводити наступне обчислення коренів полінома знаменника. Як відомо, саме через полюси визначаються перехідні процеси в об'єктах, їх смуги пропускання, стійкість та ін., а комплексно-спряжені полюси визначають резонансні явища в поведінці об'єктів.
3. *Погана обумовленість* лінійних систем рівнянь і *жорсткість* диференціальних рівнянь визначаються *різницею* максимального і мінімального власних значень відповідних матриць (коефіцієнтів лінійних рівнянь або перших похідних нелінійних рівнянь). У математичних моделях об'єктів ця різниця визначається конкретними фізичними ефектами і процесами, наприклад ефектом підсилення в електронних схемах, коли сигнали малої потужності на *базі* транзистора керують сигналами значно більшої потужності на *колекторі* транзистора. Крім того, причиною вищезгаданих особливостей математичної моделі об'єкта може бути надмірна ідеалізація характеристик його компонентів.
4. Задача обчислення власних значень симетричних матриць виявилася на практиці краще обумовленою порівняно з аналогічною процедурою для несиметричних матриць, оскільки власні значення довільної матриці є більш чутливими до малих змін елементів матриці.

Контрольні запитання та завдання

1. Для матриці з прикладу 4.2

$$\begin{bmatrix} -2 & 1 & 3 \\ 0 & -3 & 0 \\ 0 & 2 & -2 \end{bmatrix},$$

скориставшись методом Крилова, вибрати довільний вектор $\mathbf{x} = [1, 0, 0]^T$ і довести, що система рівнянь (4.10) для коефіцієнтів характеристичного полінома буде виродженою.

2. Для матриці

$$\begin{bmatrix} 2 & 8 & 10 \\ 8 & 3 & 4 \\ 10 & 4 & 7 \end{bmatrix},$$

скориставшись методом Фадєєва–Левер'є, довести, що вектор коефіцієнтів характеристичного полінома $\mathbf{b} = [1, -12, -139, 98]^{\mathrm{T}}$.

3. Застосуйте метод QR-розкладання для матриці

$$\begin{bmatrix} 5 & 1 & 4 \\ 3 & 3 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

і доведіть, що вектор власних значень матриці $\lambda = [7,509,\ 0,745 \pm i0,4928, 0]^{\mathrm{T}}$.

4. Нехай задано матрицю

$$\mathbf{A} = \frac{1}{4}\begin{bmatrix} 1 & 1 \\ -1 & 2 \end{bmatrix}.$$

Визначте, чи буде розв'язок системи $\dot{\mathbf{X}} = \mathbf{AX}$ асимптотично стійким, тобто дійсні частини комплексно-спряжених значень чи самі дійсні власні значення матриці будуть від'ємними.

5. Безпосередньою підстановкою і застосуванням тригонометричних тотожностей доведіть, що дійсна симетрична матриця порядку $n \times n$

$$\mathbf{A} = \begin{bmatrix} a & b & & & \\ b & a & b & & \\ & \ldots & \ldots & \ldots & \\ & & b & a & b \\ & & & b & a \end{bmatrix}$$

має власні значення $\lambda_k = a + 2b\cos[k\pi/(n+1)]$, $k = 1, 2, \ldots, n$, яким відповідають власні вектори $\mathbf{x}_k = \{\sin[k\pi/(n+1)], \ldots, \sin[nk\pi/(n+1)]\}$, $k = 1, 2, \ldots, n$.

Покажіть, що для одного QR-розкладання повної матриці **A** необхідно виконати $4n^3/3$ операцій. Якщо матриця зведена до форми Хессенберга, яка зберігається під час виконання наступних QR-ітерацій, то складність одного QR-розкладання зменшується до $4n^2$ операцій для несиметричних матриць і до $12n$ — для симетричних.

Розділ 5

Інтерполяція і наближення функцій

- Інтерполяційний поліном Лагранжа
- Інтерполяційні формули Ньютона
- Збіжність процесу інтерполяції
- Сплайни
- Середньоквадратичне наближення функцій
- Наближення функцій ортогональними поліномами

До обчислювальних задач аналізу звичайно відносять методи інтерполяції і наближення функцій, які використовуються, зокрема, в чисельних процедурах для локальної апроксимації експериментальних даних, а також методи чисельного диференціювання та інтегрування функцій і методи розв'язання нелінійних рівнянь. Останні часто використовуються в математичних моделях процесів, об'єктів і явищ навколишнього природного і штучного середовища. Оскільки не існує аналітичних методів розв'язання таких рівнянь (за невеликим винятком), їх розв'язують переважно чисельними методами, для яких характерне послідовне наближення (ітерації) і локальне спрощення нелінійної задачі.

5.1. Постановка задачі наближення функцій

Під час розв'язання багатьох обчислювальних задач на практиці часто доводиться заміняти одну функцію $f(x)$ (відому, невідому або частково відому) близькою до неї деякою функцією $\varphi(x)$ [1–4, 11] яка має визначені властивості. Наприклад, якщо $f(x)$ задана таблицею значень $f(x_0), f(x_1), ..., f(x_n)$ для деякої кінцевої множини аргументів x_i і в процесі розв'язування задачі необхідно використовувати значення $f(x)$ для проміжних значень аргументу, функцію $\varphi(x)$ будують таким чином, щоб у заданих точках $x_0, x_1, ..., x_n$ вона приймала значення, що збігаються зі значеннями $f(x_0), f(x_1), ..., f(x_n)$, а в інших точках відрізка $[a, b]$, що належить області визначення $f(x)$, приблизно зображувала функцію $f(x)$ з тим чи іншим ступенем точності. Тоді під час розв'язування задачі замість функції $f(x)$ використовують функцію $\varphi(x)$. Задача побудови функції $\varphi(x)$ називається задачею наближення. Найчастіше функцію $\varphi(x)$, якою

оперують під час наближення, будують у вигляді многочлена. Такий спосіб наближення базується на теоремі Вейєрштрасса про наближення неперервної функції $f(x)$ на заданому відрізку за допомогою поліномів (функція $f(x)$ може бути досить добре наближена за допомогою полінома деякого порядку m на множині точок $x_0, x_1, ..., x_n$).

У цьому разі функції виду

$$\varphi(x) = c_0\varphi_0(x) + c_1\varphi_1(x) + ... + c_m\varphi_m(x) \tag{5.1}$$

будемо називати узагальненими поліномами (узагальненими многочленами) порядку m, де $c_0, c_1, ..., c_m$ — деякі постійні коефіцієнти. Функцію $f(x)$ і поліном $\varphi(x)$ вважають *близькими*, якщо вони збігаються на заданій системі точок $x_0, x_1, ..., x_n$. Ці точки називають *вузлами інтерполяції*.

Якщо $\varphi(x)$ і $f(x)$ — диференційовані функції, то іноді у постановці задачі наближення вимагають і збігу у вузлах інтерполяції похідних $f(x)$ і $\varphi(x)$ деяких порядків.

На практиці за базисні функції $\{\varphi_i(x)\}$ часто беруть послідовність степеневих функцій відносно x: $1, x^1, x^2, ..., x^m$, тобто $\varphi_0(x) = 1, \varphi_1(x) = x, ..., \varphi_m(x) = x^m$.

Тоді маємо звичайний поліном степеня m:

$$\varphi(x) = c_0 + c_1x + c_2x^2 + ... + c_mx^m. \tag{5.2}$$

Для знаходження коефіцієнтів c_i, $i = 0, 1, ..., m$ (5.1) використаємо умову $\varphi(x_j) = f(x_j)$, $j = 0, 1, ..., n$ і сформуємо систему з $(n+1)$ лінійних алгебраїчних рівнянь на множині точок $x_0, x_1, ..., x_n$:

$$\begin{aligned}
&\varphi_0(x_0)c_0 + \varphi_1(x_0)c_1 + ... + \varphi_m(x_0)c_m = f(x_0),\\
&\varphi_0(x_1)c_0 + \varphi_1(x_1)c_1 + ... + \varphi_m(x_1)c_m = f(x_1),\\
&\qquad ... \quad ... \quad ... \quad ... \quad ...\\
&\varphi_0(x_n)c_0 + \varphi_1(x_n)c_1 + ... + \varphi_m(x_n)c_m = f(x_n).
\end{aligned} \tag{5.3}$$

За умови $n = m$ система рівнянь (5.3) має єдиний розв'язок у випадку, коли вектори $\varphi_i(x_j)$, $i, j = 0, 1, ..., n$ лінійно незалежні. Така задача наближення називається задачею інтерполяції. Якщо $m \ll n$, то система рівнянь може бути розв'язана методом найменших квадратів. Це питання докладно буде розглянуте далі, в підрозділі 5.10.

Коли $m = n$ і базисні функції $\{\varphi_i(x)\}$ мають вигляд поліномів (5.2), коефіцієнти c_i можна обчислити за допомогою такої системи рівнянь:

$$\left.\begin{aligned}
&c_0 + c_1x_0 + c_2x_0^2 + ... + c_nx_0^n = f(x_0),\\
&c_0 + c_1x_1 + c_2x_1^2 + ... + c_nx_1^n = f(x_1),\\
&\qquad ... \quad ... \quad ... \quad ... \quad ...\\
&c_0 + c_1x_n + c_2x_n^2 + ... + c_nx_n^n = f(x_n).
\end{aligned}\right\} \tag{5.4}$$

Визначник цієї системи, який називають визначником Вандермонда,

$$\Delta = \begin{vmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \dots & \dots & \dots & \dots & \dots \\ 1 & x_1 & x_n^2 & \dots & x_n^n \end{vmatrix} \tag{5.5}$$

не перетворюється на нуль, якщо серед сукупності вузлів немає таких, що збігаються, отже, матриця системи (5.4) є невиродженою і система має єдиний розв'язок.

Іноді за $\{\varphi_i(x)\}$ може братися також послідовність показникових функцій:

$$\varphi_0(x) = 1,\ \varphi_1(x) = e^{\alpha_1 x},\ \varphi_2(x) = e^{\alpha_2 x},\ \dots,\ \varphi_m(x) = e^{\alpha_m x},$$

де $\{\alpha_i\}$ — деяка числова послідовність попарно різних дійсних чисел. Тоді

$$\varphi(x) = c_0 + c_1 e^{\alpha_1 x} + c_2 e^{\alpha_2 x} + \dots + c_m e^{\alpha_m x}. \tag{5.6}$$

5.2. Інтерполяційний многочлен Лагранжа

Виходячи з однозначності інтерполяційного многочлена $\varphi(x)$, можна побудувати поліном, коефіцієнти якого визначаються із системи (5.4). Позначимо задані значення $f(x_i) = y_i$. Оскільки шуканий поліном $\varphi(x)$ повинен приймати в заданих вузлах $x_0, x_1, \dots, x_n$ значення, що збігаються зі значеннями $f(x_0)$, $f(x_1), \dots, f(x_n)$, можна записати $\varphi(x)$ у вигляді:

$$\varphi(x) = \sum_{j=0}^{n} y_i \Phi_j(x), \tag{5.7}$$

де $\Phi_j(x)$ — многочлен степеня n, що у вузлах інтерполяції задовольняє такі умови:

$$\Phi_j(x_i) = \begin{cases} 1, & \text{якщо } i = j, \\ 0, & \text{в інших випадках.} \end{cases}$$

Даний варіант запису многочлена $\varphi(x)$ називають інтерполяційним поліномом Лагранжа. Для пошуку $\Phi_j(x)$ знаходять многочлен степеня n, що перетворюється на нуль у вузлах інтерполяції x_i $(i = 0, 1, \dots, j-1, j+1, n)$ і дорівнює одиниці у точці x_j. Многочлен, що задовольняє ці вимоги, може бути записаний у вигляді:

$$\Phi_j(x) = \frac{(x - x_0)(x - x_1)\dots(x - x_{j-1})(x - x_{j+1})\dots(x - x_n)}{(x_j - x_0)(x_j - x_1)\dots(x_j - x_{j-1})(x_j - x_{j+1})\dots(x_j - x_n)}.$$

Тоді, якщо у виразі (5.7) $\Phi_j(x)$ знайдені у вказаний вище спосіб, то інтерполяційний многочлен (5.7) називається інтерполяційним многочленом Ла-

гранжа. Позначимо його як $L_n(x)$, для того щоб відрізняти цю формулу від інших випадків інтерполяції, остаточно:

$$L_n(x) = \sum_{j=0}^{n} y_j \frac{(x-x_0)(x-x_1)...(x-x_{j-1})(x-x_{j+1})...(x-x_n)}{(x_j-x_0)(x_j-x_1)...(x_j-x_{j-1})(x_j-x_{j+1})...(x_j-x_n)}. \quad (5.8)$$

Побудова інтерполяційного многочлена Лагранжа в такому вигляді для кожної конкретної задачі пов'язана зі значними обчислювальними затратами. Скористаємося для побудови і реалізації цієї формули засобами пакета Mathematica, визначимо функцію, що зображує відповідний інтерполяційний многочлен.

Приклад 5.1

Визначимо в Mathematica оператор, який відповідає інтерполяційній формулі Лагранжа, і з його допомогою побудуємо поліном Лагранжа для функції, заданої таблично:

```
In[]:= TA = {{0, 0}, {0.5, 2}, {1, 1.25}, {1.5, 3}, {2, 3.25}};
```

Відносно залежності (5.8) опишемо функцію:

```
In[]:= lgr[TA_, n_, x_]:= Sum[ta[[i,2]]*Product[If[j ≠ i, (x - ta[[j,1]])/(ta[[i,1]] - ta[[j,1]]), 1], {j, 1, n}], {i, 1, n}]
```

В останньому виразі оператора TA — таблиця значень функції, n — число табличних значень функції (n = Length[TA]), x — змінна полінома. Звернувшись до описаної функції

```
In[]:= y[x_]:= lgr[TA, Length[TA], x]; y[x]
```

отримаємо інтерполяційний поліном Лагранжа за вихідною таблицею.

```
Out[]= -5.33333 (-2 + x) (-1.5 + x) (-1 + x) x - 5. (2 - x) (-1.5 + x) (-0.5 + x) x -
       8. (-2 + x) (-1 + x) (-0.5 + x) x + 2.16667 (-1.5 + x) (-1 + x) (-0.5 + x) x
```

Або після спрощення

```
In[]:= Expand[y[x]]
Out[]= 14.875 x - 32.9583 x^2 + 25.5 x^3 - 6.16667 x^4
```

Побудуємо графік (рис. 5.1) цього многочлена за допомогою

```
In[]:= Show[Plot[y[x], {x,0,2}, Frame → True, GridLines → Automatic],
          ListPlot[TA, PlotStyle → PointSize[0.03]]];
```

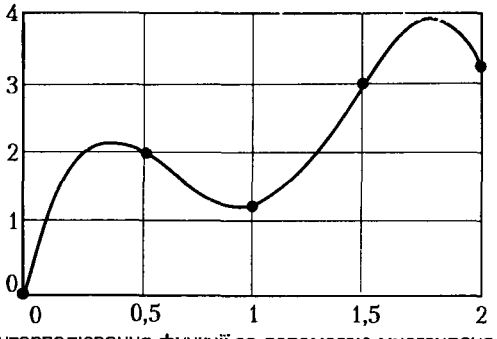


Рис. 5.1. Інтерполювання функції за допомогою многочлена Лагранжа

Отримаємо з тієї ж таблиці TA інтерполяційний поліном за допомогою стандартного оператора Mathematica:

```
In[]:= yp[z_]:= InterpolatingPolynomial[TA, z]
```

і порівняємо його з y[x], що був отриманий вище:

```
In[]:= Expand[yp[z]]
Out[]= 14.875 z - 32.9583 z^2 + 25.5 z^3 - 6.16667 z^4
```

Поліноми збіглися, що свідчить про правильність визначення функції.

Задача інтерполяції значно спрощується, якщо значення x_i є рівновіддаленими, тобто $x_i = x_0 + ih$, $(i = 1, 2, ..., n)$, тоді можна ввести позначення

$$\frac{x - x_0}{h} = t,$$

і інтерполяційний поліном буде мати вигляд:

$$L_n(x) = (-1)^n \frac{t(t-1)...(t-n)}{n!} \sum_{i=0}^{n} (-1)^i \frac{C_n^i y_i}{t-i}. \qquad (5.9)$$

У (5.9) коефіцієнти перед знаком суми

$$(-1)^n \frac{t(t-1)...(t-n)}{n!}$$

не залежать ні від значень функції $f(x)$, ні від відстані між вузлами інтерполяції h. Їх називають коефіцієнтами Лагранжа. Розглянемо, як можна отримати ці коефіцієнти і використати їх під час розрахунків за допомогою Mathematica.

Приклад 5.2

Скористаємося вихідною таблицею TA з прикладу 5.1, оскільки вузли в ній розташовані регулярно. Опишемо в Mathematica функцію з огляду на залежності (5.9):

```
In[]:= <<DiscreteMath`Combinatorica`
       Ta = {{0, 0}, {0.5, 2}, {1, 1.25}, {1.5, 3}, {2, 3.25}}; n = Length[Ta] - 1;
       p = Table[i, {i, 0, n-1}]; q = Table[Length[KSubsets[p, i]], {i, 0, n-1}];
       Lgr[t_, n_, ta]:= (-1)^n*t* Product[(t-i)/i, {i, 1, n}] * Sum[(-1)^j*q[[j+1]]*ta[[j+1, 2]]/(t-j), {j, 0, n}];
```

Звернення до функції та виведення результату здійснимо за допомогою команди:

```
In[]:= y[t_]:= Lgr[t, Length[TA], TA];  y[t]
```

Отримаємо:

$$\text{Out[]}= \frac{1}{24}\left(\frac{3.25}{-4+t} - \frac{12}{-3+t} + \frac{7.5}{-2+t} - \frac{8}{-1+t}\right)(-4+t)(-3+t)(-2+t)(-1+t)t$$

Замінивши в отриманому виразі параметр t на значення $(x - x_0)/h = t$, матимемо:

```
In[]:= x0 = 0; h = 0.5; y1[t] /. t → (x - TA[[1,1]])/(TA[[2,1]] - TA[[1,1]])
Out[]= 0.833333 x (-4 + 2. x) (-3 + 2. x) (-2 + 2. x) (-1 + 2. x) +
       (3.25/(-4 + 2. x) - 12/(-3 + 2. x) + 7.5/(-2 + 2. x) - 8/(-1 + 2. x))
```

Побудуємо для цій функції графік (рис. 5.2), визначивши на ньому ще і значення функції у вузлах:

```
In[]:= Show[Plot[yl[x], {x, 0, 2}, Frame → True, GridLines → Automatic],
          ListPlot[TA, PlotStyle → PointSize[0.03]]]
```

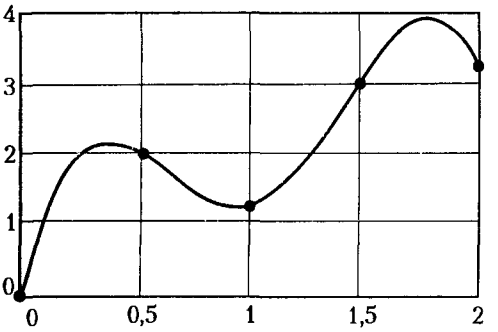


Рис. 5.2. Графік полінома Лагранжа для рівновіддалених вузлів

5.2.1. Похибки інтерполяційної формули Лагранжа

Різницю між функцією $f(x)$ та її інтерполяційним наближенням $L_n(x)$ називають залишковим членом інтерполяційної формули або похибкою інтерполяції [27, 29]:

$$r_n(x) = f(x) - L_n(x). \tag{5.10}$$

Судячи з формули (5.8), зрозуміло, що у вузлах інтерполяції ця похибка дорівнює нулю, тому похибку (5.10) необхідно оцінювати в інших точках відрізка $[a, b]$. Для цього припускають, що інтерпольована функція $f(x)$ має на відрізку $[a, b]$ неперервні похідні до порядку $n+1$. Такі припущення виконуються для більшості випадків, із якими доводиться стикатися на практиці.

Будемо вважати, що $f(x)$ — відома функція. Побудуємо інтерполяційний поліном $f(x) \approx L_n(x)$, причому в $n+1$ вузлах інтерполяції $f(x_i) = L_n(x_i)$, $i = 0, 1, 2, ..., n$. Потрібно оцінити похибку $r_n(x)$ у заданій точці відрізка $[a, b]$, що не є вузлом інтерполяції:

$$r_n(x) = f(x) - L_n(x),$$
$$x \neq x_i, \quad i = 0, 1, 2, ..., n.$$

Введемо допоміжну функцію

$$\varphi(z) = f(z) - L_n(z) - r_n(z), \tag{5.11}$$

у випадку, коли $z = x$, одержуємо $\varphi(x) = 0$ з огляду на співвідношення (5.10). Припустимо, що похибка має вигляд:

$$r_n(x) = C(x - x_0)(x - x_1)...(x - x_n) = C\,\omega(x),$$

де C — деяка постійна, котру можна визначити як

$$C = \frac{f(x) - L_n(x)}{(x - x_0)(x - x_1)...(x - x_n)} = \frac{f(x) - L_n(x)}{\omega(x)}$$
$$\omega(x) = (x - x_0)(x - x_1)...(x - x_n).$$

Із умов $\varphi(z) = 0$ і $\varphi(x_i) = 0 = f(x_i) - L_n f(x_i) - r_n(x_i)$ зрозуміло, що $\varphi(z)$ перетворюється на нуль принаймні в $n+2$ точках $x, x_0, x_1, ..., x_n$. За теоремою Ролля, якщо функція в точках α і β перетворюється на нуль, то знайдеться точка η на відрізку $[\alpha, \beta]$, похідна якої теж перетворюється на нуль. Тобто всередині кожного проміжку $x, x_0, x_1, ..., x_n$ знайдуться точки $\eta_0, \eta_1, ..., \eta_n$, похідні в яких дорівнюють нулю. Можна стверджувати, що:

$$\varphi'(z) = 0 \quad \text{не менш ніж у } n+1 \text{ точках;}$$
$$\varphi''(z) = 0 \quad \text{не менш ніж у } n \text{ точках;}$$
$$\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots$$
$$\varphi^{(n+1)}(z) = 0 \quad \text{принаймні в одній точці } \eta.$$

Можна вважати, що на відрізку $[a, b]$ знайдеться точка ξ, для якої $\varphi^{(n+1)}(\xi) = 0$. Знайдемо значення

$$\varphi^{(n+1)}(\xi) = f^{(n+1)}(\xi) - L_n^{(n+1)}(z) - r_n^{(n+1)}(z).$$

Оскільки $L_n(z)$ — поліном степеня n, то $L_n^{(n+1)}(x) = 0$, $r_n(z)$ — поліном степеня $n+1$, тому $r_n^{(n+1)}(z) = (n+1)!C$. Тоді $\varphi^{(n+1)}(\xi) = f^{(n+1)}(\xi) - (n+1)!C$. Звідси одержимо вираз для константи:

$$C = \frac{f^{(n+1)}(\xi)}{(n+1)!}.$$

Тоді на підставі припущення про вид похибки маємо:

$$r_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\omega(x). \tag{5.12}$$

Похибку інтерполяції полінома Лагранжа можна оцінити в такий спосіб:

$$|f(x) - L_n(x)| \leqslant \frac{M_{n+1}}{(n+1)!}|\omega(x)|, \tag{5.13}$$

де

$$M_{n+1} = \sup_{x \in [a,b]} \left| f^{(n+1)}(x) \right|.$$

Для випадку, коли $f(x)$ є поліномом степеня n, інтерполяція, проведена по будь-яких точках $x_0, x_1, ..., x_n$, здійснюється точно, тобто

$$L_n(x) \equiv f(x).$$

Формула (5.13) дає можливість провести апріорну оцінку похибки, тобто для випадку аналітично заданої функції $f(x)$ провести оцінювання до початку обчислень. Наведемо приклад використання цієї формули.

Приклад 5.3

Побудуємо інтерполяційний поліном для функції

$$f(x) = e^{-0,1x} \sin x$$

на відрізку $0 \leqslant x \leqslant 1$ такий, щоб похибка інтерполяції не перевищувала 0,000005.

На відрізку [0, 1] виберемо сітку з кроком $h = 0,2$. Для заданої функції $f(x)$ на обраній сітці визначимо похибку. Знайдемо M_{n+1} за допомогою операторів Mathematica. Введемо функцію

```
In[]:= f[x_]:= Exp[-x/10]*Sin[x];
```

Визначимо її похідну n-го порядку

```
In[]:= fn[x_,n_]:= Dt[f[x], {x, n}];
```

Оскільки таблиця в нашому прикладі містить шість вузлів інтерполяції, знайдемо похідну шостого порядку:

```
In[]:= f6[x_] = N[fn[x,6]]; f6[x]
Out[]= -0.58006 2.71828^-0.1x Cos[x] - 0.851499 2.71828^-0.1x Sin[x]
```

Побудуємо графік (рис. 5.3) функції даної похідної:

```
In[]:= Plot[f6[x], {x, 0, 1}, AxesLabel → {"x", "f^(VI)(x)"}];
```

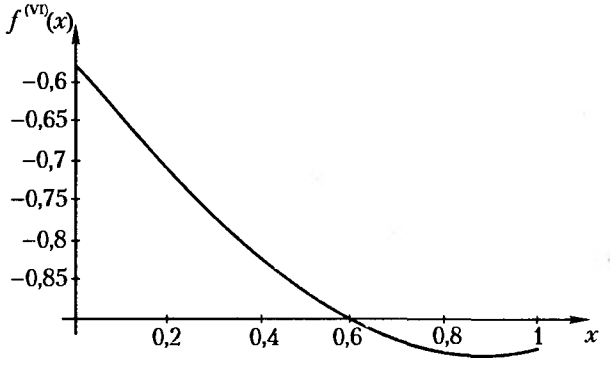


Рис. 5.3. Графік похідної шостого порядку функції $f(x) = e^{-0,1x} \sin x$

Знайдемо її мінімум :

```
In[]:= m = FindMinimum[f6[x], {x, 0.5}]
Out[]= {-0.939473, {x → 0.873116}}
```

Максимум модуля шостої похідної буде дорівнювати $M_6 = 0{,}939473$. Знайдемо максимум модуля функції $\omega(x)$. Спочатку визначимо цю функцію для загального випадку:

```
In[]:= Om[Ta_, n_, x_]:= Product[x - Ta[[i]]];
```

Введемо таблицю вузлів:

```
Ta = {0, 0.2, 0.4, 0.6, 0.8, 1.0};
```

і на її основі побудуємо графік (рис. 5.4) функції $\omega(x)$:

```
Plot[Om [Ta, Length[Ta], x], {x, 0, 1}, AxesLabel → {"x", "ω(x)"}];
```

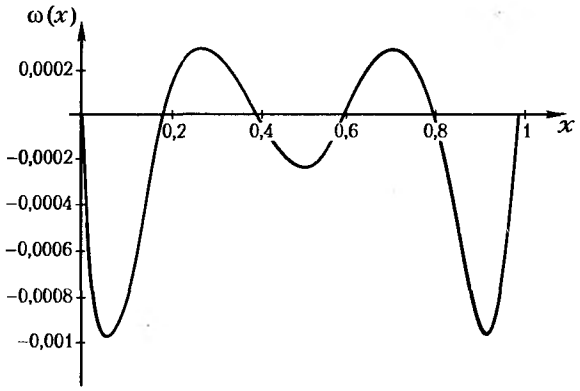


Рис. 5.4. Графік многочлена ω(x), побудованого для $f(x) = e^{-0,1x} \sin x$

Знайдемо мінімум $\omega(x)$:

```
In[]:= mω = FindMinimum[Om [Ta, Length[Ta], x], {x, 0}]
Out[]= {-0.00108166, {x → 0.0673107}}
```

Максимум її модуля дорівнює `Mω = - mω[[1]]`. Таким чином, помилка інтерполяції на обраній сітці складе:

```
In[]:= ri = m[[1]]*mω[[1]]/6!
Out[]= 1.41137*10^-6
```

тобто поліном п'ятого степеня, побудований за таблицею значень функції з обраними вузлами, забезпечить необхідну точність зображення заданої функції. Обчислимо цю таблицю:

```
In[]:= Ty = Table[f[Ta[[i]]], {i,6}]
Out[]= {0, 0.194735, 0.374149, 0.53176, 0.662203, 0.761394}
```

Утворимо з двох векторів `Ta` і `Ty` матрицю `XY`:

```
In[]:= XY = {Ta, Ty}
Out[]= {{0, 0.2, 0.4, 0.6, 0.8, 1.},
        {0, 0.194735, 0.374149, 0.53176, 0.662203, 0.761394}}
```

Транспонуємо останню матрицю:

```
In[]:= Tm = Transpose[XY]
Out[]= {{0, 0}, {0.2, 0.194735}, {0.4, 0.374149},
        {0.6, 0.53176}, {0.8, 0.662203}, {1., 0.761394}}
```

Тепер можемо скористатися оператором Mathematica для одержання інтерполяційного полінома:

```
In[]:= lpol = InterpolatingPolynomial[Tm, x];
```

Знайдемо похибку інтерполяції:

```
r = f[x] - lpol;
```

і побудуємо її графік (рис. 5.5):

```
Plot[r, {x, 0, 1}, AxesLabel → {"x", "r(x)"}]
```

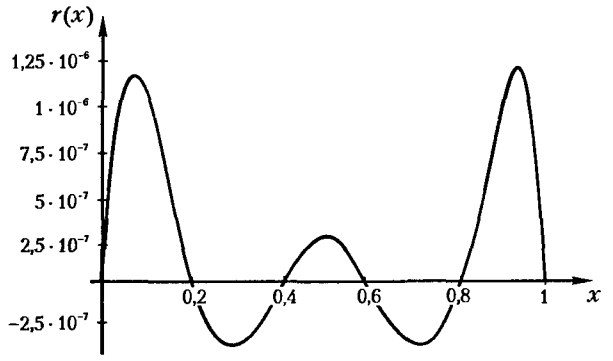


Рис. 5.5. Графік похибки інтерполяції функції $f(x) = e^{-0.1x} \sin x$

5.3. Інтерполяційні формули Ньютона

Для побудови інтерполяційних поліномів Ньютона необхідно спочатку ввести поняття скінченних і розділених різниць. Нехай є система значень заданої функції у вузлах інтерполяції $f(x_0), f(x_1), ..., f(x_n)$. Скінченними різницями першого порядку називають величини, що обчислюються за виразами

$$\begin{aligned} \nabla f(x_0) &= f(x_1) - f(x_0), \\ \nabla f(x_1) &= f(x_2) - f(x_1), \\ &\dots \quad \dots \quad \dots \quad \dots \quad \dots \\ \nabla f(x_{n-1}) &= f(x_n) - f(x_{n-1}), \end{aligned} \tag{5.14}$$

а скінченними різницями другого порядку називають:

$$\begin{aligned} \nabla^2 f(x_0) &= \nabla f(x_1) - \nabla f(x_0), \\ &\dots \quad \dots \quad \dots \quad \dots \quad \dots \\ \nabla^2 f(x_{n-2}) &= \nabla f(x_{n-1}) - \nabla f(x_{n-2}). \end{aligned} \tag{5.15}$$

У загальному вигляді різниця m-го порядку обчислюється за формулою

$$\nabla^m f(x_k) = \nabla^{m-1} f(x_{k+1}) - \nabla^{m-1} f(x_k). \tag{5.16}$$

Розділені різниці першого порядку для $f(x_0)$ у вузлах інтерполяції мають вигляд:

$$\begin{aligned} f(x_0; x_1) &= \frac{f(x_1) - f(x_0)}{x_1 - x_0}, \\ f(x_1; x_2) &= \frac{f(x_2) - f(x_1)}{x_2 - x_1}, \\ &\dots \quad \dots \quad \dots \quad \dots \quad \dots \\ f(x_{n-1}; x_n) &= \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}. \end{aligned} \tag{5.17}$$

Розділені різниці другого порядку записують у вигляді:

$$\begin{aligned} f(x_0; x_1; x_2) &= \frac{f(x_1; x_2) - f(x_0; x_1)}{x_2 - x_0}, \\ f(x_1; x_2; x_3) &= \frac{f(x_2; x_3) - f(x_1; x_2)}{x_3 - x_1}, \\ &\dots \quad \dots \quad \dots \quad \dots \quad \dots \\ f(x_{n-2}; x_{n-1}; x_n) &= \frac{f(x_{n-1}; x_n) - f(x_{n-2}; x_{n-1})}{x_n - x_{n-2}}. \end{aligned} \tag{5.18}$$

Взагалі, якщо розділені різниці k-го порядку вже визначені, то розділені різниці $(k+1)$-го порядку знаходять за допомогою формули:

$$f(x_{i-1}; x_i; \dots; x_{i+k}) = \frac{f(x_i; x_{i+1}; \dots; x_{i+k}) - f(x_{i-1}; x_i, \dots, x_{i+k-1})}{x_{i+k} - x_{i-1}}. \tag{5.19}$$

Використовуючи розділені різниці, можна одержати формулу Ньютона для нерівних проміжків у вигляді:

$$\begin{aligned} H_n(x) = f(x_0) &+ \\ &+ (x - x_0) f(x_0; x_1) + \\ &+ (x - x_0)(x - x_1) f(x_0; x_1; x_2) + \dots \\ &\dots + (x - x_0)(x - x_1) \dots (x - x_{n-1}) f(x_0; x_1; \dots; x_n). \end{aligned} \tag{5.20}$$

Переконаємося, що отриманий вираз дійсно є інтерполяційним поліномом, а саме він є поліномом n-го степеня і приймає у вузлах інтерполяції задані значення:

$$\begin{aligned} H_n(x_0) &= f(x_0), \\ H_n(x_1) &= f(x_0) + (x_1 - x_0) f(x_0; x_1) = f(x_1), \\ &\dots \quad \dots \quad \dots \quad \dots \quad \dots \end{aligned}$$

Вибравши довільну точку x_k; $0 \leqslant k \leqslant n$, можна довести, що

$$H_k(x_k) = f(x_0) + (x_k - x_0)f(x_0; x_1) + ...$$
$$... + (x_k - x_0)(x_k - x_1)...(x_k - x_{k-1})f(x_0; x_1; ...; x_k) = f(x_k),$$

тобто всі вимоги до інтерполяційних многочленів виконуються.

Як приклад розглянемо побудову інтерполяційного полінома Ньютона.

Приклад 5.4

Визначимо функцію, що обчислює на основі значень заданої таблиці TA (для якої раніше ми вже будували поліном Лагранжа) розділену різницю n-го порядку:

```
In[]:= TA = {{0, 0}, {0.5, 2}, {1, 1.25}, {1.5, 3}, {2, 3.25}};

        rd[ta_,n_]:= Sum[ta[[i,2]]/ Product[If [j≠i, (ta[[i,1]] - ta[[j,1]]), 1]
```

Визначимо інтерполяційний поліном Ньютона за такою формулою:

```
New[ta_,n_,x_]:= Sum[rd[ta, k]* Product[(x - ta[[i,1]])
```

Отримаємо інтерполяційний поліном на основі даних таблиці TA:

```
q = New[TA, Length[TA], x]
```

Після зведення подібних і побудови графіка (рис. 5.6) отримаємо шуканий результат:

```
Expand[q];
Out[]= 14.875x - 32.9583x^2 + 25.5x^3 - 6.16667x^4

In[]:= Show[Plot[q, {x,0,2}, Frame → True, GridLines → Automatic],
          ListPlot[TA, PlotStyle → PointSize[0.03]]];
```

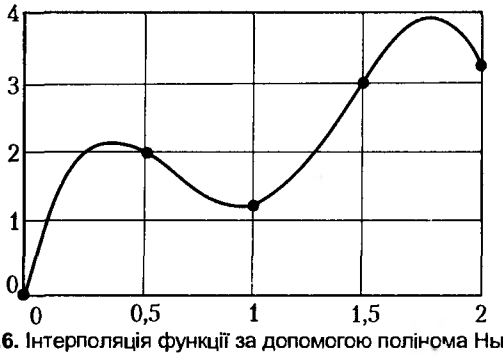


Рис. 5.6. Інтерполяція функції за допомогою полінома Ньютона

Порівнюючи рис. 5.1 і 5.6, можна переконатися, що обидва методи будують один і той самий інтерполяційний поліном. У той же час формула (5.20) більш зручна, порівняно з формулою Лагранжа, тому що для обчислень можуть використовуватися не всі задані точки таблиці, а тільки їх частина. При цьому зрозуміло, що вузли, які лежать ближче до інтерполяційного значення x, впливають на інтерполяційний поліном більше, ніж ті вузли, що лежать далі. Формула Ньютона у вигляді (5.20) використовується для інтерполяції функції на даному відрізку $[a, b]$, якщо шукані точки знаходяться на початку таблиці.

У тому випадку коли шукані точки розташовані ближче до кінця таблиці, використовується формула Ньютона для інтерполяції назад:

$$H_n(x) = f(x_n) + (x - x_n)f(x_n; x_{n-1}) + \\ + (x - x_n)(x - x_{n-1})f(x_n; x_{n-1}; x_{n-2}) + \ldots + \\ + (x - x_n)(x - x_{n-1})\ldots(x - x_1)f(x_n; x_{n-1}; \ldots; x_1; x_0). \tag{5.21}$$

Так само, як і (5.20), ця формула є поліномом n-го степеня й у вузлах інтерполяції приймає задані значення.

Формули (5.20) і (5.21) використовуються для довільно розташованих вузлів. Однак дуже часто ці вузли будуються регулярно. Розглянемо окремий випадок формули Ньютона для інтерполяції на початку таблиці, якщо вузли рівновіддалені. Нехай відстань між сусідніми вузлами $x_i - x_{i-1} = h$, $i = 1, 2, \ldots, n$. Запишемо розділені різниці у формулі (5.20), скориставшись скінченними різницями:

$$f(x_0; x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{\nabla f(x_0)}{h},$$

$$f(x_0; x_1; x_2) = \frac{f(x_1; x_2) - f(x_0; x_1)}{x_2 - x_0} = \frac{\nabla f(x_1) - \nabla f(x_0)}{2h^2} = \frac{\nabla^2 f(x_0)}{2h^2}.$$

Розділену різницю k-го порядку замінимо співвідношенням:

$$f(x_0; x_1; \ldots; x_k) = \frac{\nabla^k f(x_0)}{k! h^k}.$$

Введемо заміну $(x - x_0)/h = t$.

Тоді формула Ньютона для інтерполяції на початку таблиці з рівновіддаленими вузлами буде мати вигляд:

$$H_n(t) = f(x_0) + t\nabla f(x_0) + \frac{t(t-1)}{2!}\nabla^2 f(x_0) + \ldots \\ \ldots + \frac{t(t-1)\ldots(t-(n-1))}{n!}\nabla^n f(x_0). \tag{5.22}$$

Формула Ньютона для інтерполяції наприкінці таблиці з рівновіддаленими вузлами, якщо прийняти, що $t = (x - x_n)/h$, буде мати вигляд

$$H_n(t) = f(x_n) + t\nabla f(x_{n-1}) + \frac{t(t+1)}{2!}\nabla^2 f(x_{n-2}) + \ldots \\ \ldots + \frac{t(t+1)\ldots(t+n-1)}{n!}\nabla^n f(x_0). \tag{5.23}$$

Якщо обчислювані скінченні різниці записувати в таблиці, то для формули (5.22) буде використовуватися верхній рядок різниць, а для формули (5.23) – нижній косий рядок різниць. Необхідно пам'ятати, що кожна з отриманих формул Ньютона є іншою формою запису многочлена Лагранжа, і що різняться ці

формули тільки застосовуваними скінченними різницями (за умови, що в них використані ті ж самі вузли інтерполяції). Обираючи конкретну формулу, потрібно зважати на те, що звичайно буває зручніше вести обчислення, якщо для інтерполяції спочатку використовуються найближчі до x вузли, а потім підключаються більш віддалені. При цьому обчислюване значення здебільшого залежатиме від перших членів інтерполяційних формул. У цьому випадку легше встановити, на якій різниці варто закінчити обчислення.

5.3.1. Похибки інтерполяційних формул Ньютона

Оскільки поліноми Лагранжа і Ньютона, побудовані на основі однієї й тієї ж самої таблиці, різняться лише формою запису, зображення похибки у вигляді (5.13) справедливе як для формули Лагранжа, так і для формул Ньютона.

Можна обчислити залишковий член інтерполяційних формул Ньютона, якщо встановити зв'язок між розділеною різницею порядку $(n+1)$ і $(n+1)$-ю похідною функції $f(x)$. Для цього запишемо розділену різницю $f(x, x_0, x_1, ..., x_n)$ в такому вигляді:

$$f(x; x_0; x_1; ...; x_n) = \frac{f(x)}{(x-x_0)(x-x_1)...(x-x_n)} + \\ + \frac{f(x_0)}{(x-x_0)(x-x_1)...(x-x_n)} + ... + \frac{f(x_n)}{(x-x_0)(x-x_1)...(x-x_n)},$$

порядок її дорівнює n. Звідси можна записати:

$$f(x) = f(x_0)\frac{(x-x_1)(x-x_2)...(x-x_n)}{(x_0-x_1)(x_0-x_2)...(x_0-x_n)} + ... \\ ... + f(x_n)\frac{(x-x_0)(x-x_1)...(x-x_n)}{(x_n-x_0)(x_n-x_1)...(x_n-x_{n-1})} + \\ + (x-x_0)(x-x_1)...(x-x_n)f(x, x_0, x_1, ..., x_n), \\ f(x) = L_n(x) + (x-x_0)(x-x_1)...(x-x_n)f(x, x_0, ..., x_n).$$

Тоді похибку інтерполяційної формули можна зобразити у вигляді:

$$f(x) - L_n = \omega(x)f(x, x_0, ..., x_n). \qquad (5.24)$$

Порівнявши отриманий вираз зі співвідношенням (5.13), можна стверджувати, що існує деяка точка $\xi \in [a, b]$, для якої

$$f^{(n+1)}(\xi) = \frac{f(x, x_0, ..., x_n)}{(n+1)!}.$$

Таким чином, для оцінки похибки залишкового члена інтерполяційних формул Ньютона може бути використана залежність (5.13).

5.4. Інтерполяція в середині таблиці

Позначимо через x_0 внутрішній вузол таблиці. Припустимо, що точка інтерполяції x лежить поблизу x_0, з того чи іншого боку. Будемо залучати для інтерполяції табличні точки в такому порядку: спочатку виберемо x_0, а потім пари точок $(x_0 + h; x_0 - h), (x_0 + 2h; x_0 - 2h), ..., (x_0 + kh; x_0 - kh)$. Число взятих вузлів буде непарним і рівним $2k + 1$. Позначимо, як і раніше,

$$t = (x - x_0)/h,$$
$$f(x_0) = f_0, \quad f(x_1) = f_{-1}, \quad \ldots .$$

Формула Гаусса для випадку, коли $x > x_0$, має такий вигляд:

$$G(x) = f_0 + t\nabla f_0 + \frac{t(t+1)}{2!}\nabla^2 f_{-1} + \frac{(t+1)t(t-1)}{3!}\nabla^3 f_{-1} + \\ + \frac{(t+1)t(t-1)(t-2)}{4!}\nabla^4 f_{-2} + \ldots . \tag{5.25}$$

Якщо шукана точка x знаходиться в середині таблиці і зв'язана з x_0 співвідношенням $x < x_0$, формула Гаусса використовується у вигляді:

$$G(x) = f_0 + t\nabla f_{-1} + \frac{t(t-1)}{2!}\nabla^2 f_{-1} + \frac{(t+1)t(t-1)}{3!}\nabla^3 f_{-2} + \\ + \frac{(t+2)(t+1)t(t-1)}{4!}\nabla^4 f_{-2} + \ldots . \tag{5.26}$$

На основі двох представлених формул Гаусса можна одержати в результаті перетворень формули Стирлінга, Бесселя і Евереттa.

Обчисливши середнє арифметичне першої та другої інтерполяційних формул Гаусса (5.25) і (5.26), одержимо формулу Стирлінга:

$$S(x) = f_0 + t\frac{\nabla f_0 + \nabla f_{-1}}{2} + \frac{t^2}{2}\nabla^2 f_{-1} + \frac{t(t^2-1)}{3!}\frac{\nabla^3 f_{-2} + \nabla^3 f_{-1}}{2} + \\ + \frac{t^2(t^2-1)}{4!}\nabla^4 f_{-2} + \ldots . \tag{5.27}$$

Легко бачити, що $S_n(x_i) = y_i$, коли $i = 0, \pm 1, \pm 2, ..., \pm k$.

Для виведення формули Бесселя використовується друга інтерполяційна формула Гаусса (5.26). Після нескладних перетворень одержимо:

$$B(x) = \frac{f_0 + f_{-1}}{2} + (t - 1/2)\nabla f_0 + \frac{t(t-1)}{2}\frac{\nabla^2 f_{-1} + \nabla^2 f_0}{2} + \\ + \frac{(t-1/2)t(t-1)}{3!}\nabla^3 f_{-1} + \frac{t(t-1)(t+1)(t-2)}{4!}\frac{\nabla^4 f_{-2} + \nabla^4 f_{-1}}{2} + \\ + \frac{(t-1/2)t(t-1)(t+1)(t-2)}{5!}\nabla^5 f_{-2} + \ldots . \tag{5.28}$$

У формулі Бесселя всі члени, які містять різниці непарного порядку, мають множник $t-1/2$, тому, якщо $t=1/2$, формула (5.28) значно спрощується, оскільки всі члени, які містять $t-1/2$ перетворюються на нуль.

За інтерполяції функцій, заданих таблично з постійним кроком аргументу h, слід керуватися такими правилами. Якщо значення x знаходиться близько до початку відрізка, на якому задана таблиця значень функції, то для інтерполяції потрібно використовувати формулу Ньютона для інтерполяції вперед (5.22), а коли x близькі до кінця відрізка — формулу Ньютона для інтерполяції назад (5.23), тому що ці формули допускають застосування правильних різниць до максимального порядку. Якщо аргумент x, для якого потрібно обчислити значення функції, знаходиться в середині таблиці, рекомендується використовувати формулу Стирлінга, якщо $|t| \leqslant 1/4$, і формулу Бесселя — якщо $1/4 \leqslant |t| \leqslant 3/4$. Крім того, в разі використання формули Стирлінга слід враховувати в ній останню, правильну різницю непарного порядку, а у формулі Бесселя — останню, правильну різницю парного порядку.

Залишкові члени інтерполяційних формул для середини таблиці приблизно дорівнюватимуть першому невраховуваному члену.

Запишемо без доведення залишкові члени для формул Стирлінга і Бесселя. Залишковий член інтерполяційної формули Стирлінга, якщо $2k$, — порядок максимальної використовуваної різниці таблиці й $x \in [x_0 - kh, x_0 + kh]$, має вигляд:

$$r_k(x) = \frac{h^{2k+1} f^{(2k+1)}(\xi)}{(2k+1)!} t(t^2-1^2)(t^2-2^2)\ldots(t^2-k^2), \qquad (5.29)$$

де

$$t = \frac{x - x_0}{h} \quad \text{і} \quad \xi \in [x_0 - kh, x_0 + kh].$$

Залишковий член інтерполяційної формули Бесселя, якщо порядок максимальної використаної різниці таблиці дорівнює $2k+1$ і $x \in [x_0 - kh, x_0 + kh]$, має вигляд:

$$r_k(x) = \frac{h^{2k+2}}{(2k+2)!} f^{(2k+2)}(\xi) t(t^2-1^2)(t^2-2^2)\ldots(t^2-k^2)[t-(n+1)], \qquad (5.30)$$

де

$$t = \frac{x - x_0}{h} \quad \text{і} \quad \xi \in [x_0 - kh, x_0 + kh].$$

5.5. Вибір вузлів інтерполяції

Похибки інтерполяційних формул (5.12) дорівнюють добутку двох множників, з яких один, $f^{(n+1)}(\xi)$, залежить від властивостей функції $f(x)$ і не піддається корегуванню, а величина іншого, $\omega(x)$, визначається винятково вибором вузлів інтерполяції.

Задача про раціональний вибір вузлів інтерполяції x_i (коли задана кількість вузлів n) для того, щоб поліном $\omega_{n+1}(x)$ мав найменше максимальне значення за абсолютною величиною на відрізку $[a, b]$, була розв'язана Чебишевим, який довів, що найкращий вибір задається формулою:

$$x_i = \frac{b+a}{2} + \frac{b-a}{2}\xi_i, \tag{5.31}$$

де

$$\xi_i = -\cos\frac{2i+1}{2n+2}\pi, \quad (i = 1, 2, ..., n).$$

У цьому випадку можна стверджувати, що $|\omega_{n+1}(x)| \leqslant ((b-a)/4)^{n+1}$. За допомогою цих залежностей можна зменшити похибку інтерполяції.

Приклад 5.5

Скориставшись даними з прикладу 5.3, у якому розглядалася похибка інтерполяції для функції з регулярно заданими вузлами, побудуємо чебишевську систему вузлів. Використаємо для вибору вузлів інтерполяції співвідношення (5.31), кількість вузлів при цьому залишимо такою ж, як і в прикладі 5.3. За допомогою наведених нижче операторів можна побудувати чебишевську систему вузлів:

```
In[]:= a = 0; b = 1; n = 5;
       f[x_]:= Exp[-x/10]*Sin[x];    fn[x_,n_]:= Dt[f[x], {x, n}];
       f6[x_] = N[fn[x,6]]; m = FindMinimum[f6[x], {x, 0.5}];
       Om[Ta_,n_,x_]:= Product[(x - Ta[[i]]), {i, 1, n}];
       Do [ y[i] = -Cos[(2*(i-1) + 1)/(2*n + 2)*π];
       Uzel[i] = N[(b + a)/2 + (b - a)/2*y[i]], {i, n + 1} ];
       Ta = Table[Uzel[i], {i, n + 1}]
Out[]= {0.0170371, 0.146447, 0.37059, 0.62941, 0.853553, 0.982963}
```

Далі скористаємося всіма вже описаними змінними і функціями й побудуємо графік $\omega(x)$ (рис. 5.7):

```
In[]:= Plot[Om [Ta, Length[Ta], x], {x, Ta[[1]], Ta[[n+1]]}, AxesLabel → {"x", "ω(x)"}]
```

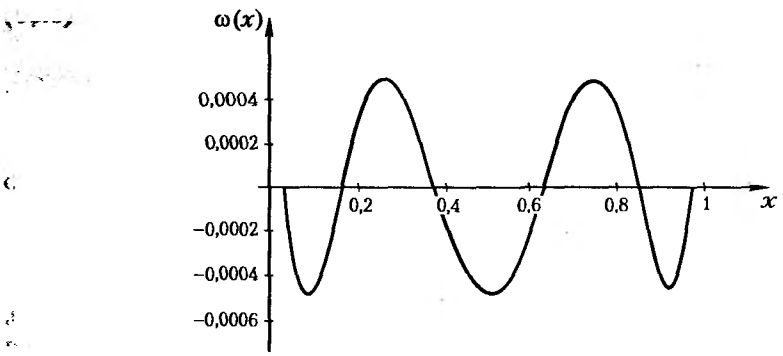


Рис. 5.7. Графік многочлена ω(x), побудованого для функції $f(x) = e^{-0,1x}\sin x$ із чебишевським розташуванням вузлів

Порівнявши отриманий графік із графіком, показаним на рис. 5.4, можна стверджувати, що амплітуда функції $\omega(x)$ зменшилася. Знайдемо мінімум $\omega(x)$:

```
In[]:= mω = FindMinimum[Om [Ta, Length[Ta], x], {x, 0}]
Out[]= {-0.000488281, {x → 0.0669873}}
```

Максимум її модуля $M_\omega = -m\omega[[1]]$. Таким чином, помилка інтерполяції на вибраній сітці складе:

```
In[]:= ri = m[[1]]*mω[[1]]/6!
Out[]= 6.37121*10^-7
```

Точність многочлена п'ятого степеня, побудованого на основі таблиці значень функції з нерегулярними вузлами, буде вище. Виконаємо ту ж послідовність дій, що і раніше:

```
In[]:= Ty = Table[f[Ta[[i]]], {i, 6}]
       XY = {Ta, Ty};
       Tm = Transpose[XY]
```

і одержимо остаточну таблицю:

```
Out[]= {{0.0170371,  0.0170073}, {0.146447,  0.143802},
        {0.37059,    0.34899},   {0.62941,   0.552758},
        {0.853553,   0.691964},  {0.982963,  0.754239}}
```

Тепер можемо скористатися оператором Mathematica для одержання інтерполяційного полінома:

```
In[]:= lpol = InterpolatingPolynomial[Tm, x];
```

Знайдемо похибку інтерполяції:

```
In[]:= r = f[x] - lpol;
```

і побудуємо її графік (рис. 5.8)

```
In[]:= Plot[r, {x, 0, 1}, AxesLabel → {"x", "r(x)"}];
```

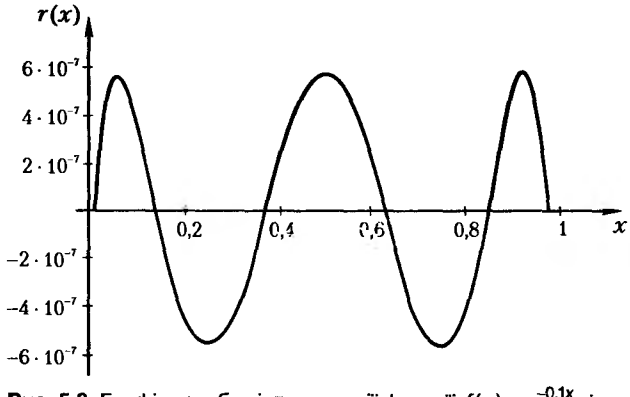


Рис. 5.8. Графік похибки інтерполяції функції $f(x) = e^{-0.1x} \sin x$ із чебишевським розташуванням вузлів

Порівнявши рис. 5.5 і 5.8, можна стверджувати, що похибка зменшилася більш ніж у два рази. Тому у випадку, коли виникає необхідність наближення відомої аналітично функції, вузли рекомендується вибирати за допомогою співвідношень (5.31).

5.6. Збіжність інтерполяційного процесу

Збільшення кількості вузлів інтерполяції з метою зменшення похибки $f(x) - L_n(x)$ не завжди виправдане [15–17]. У зв'язку з цим виникає задача дослідження збіжності інтерполяційного процесу введенням на заданому відрізку $[a, b]$ послідовності сіток зі зростаючою кількістю вузлів, а саме:

$$\begin{aligned} S^0 &= \{x_0^0\}, \\ S^1 &= \{x_0^1, x_1^1\}, \\ &\dots \quad \dots \quad \dots \\ S^n &= \{x_0^n, x_1^n, \dots, x_n^n\}, \end{aligned} \tag{5.32}$$

де $a \leqslant x_0 < x_1 < \dots < x_n \leqslant b$.

Тоді можна задати послідовність інтерполяційних поліномів $L_n[f(x)]$, побудованих для функції $f(x)$ за її значеннями у вузлах сітки S_n, для кожного n-го рядка послідовності (5.32).

Інтерполяційний процес для функції $f(x)$ збігається в точці $\xi \in [a, b]$ (точкова збіжність), якщо

$$\lim_{n\to\infty} L_n(\xi) = f(\xi), \quad \xi \in [a, b].$$

Рівномірна збіжність на відрізку $[a, b]$ означає, що

$$\max_{x\in[a,b]} |f(x) - L_n[f(x)]| \to 0, \; x \in [a, b], \quad \text{коли} \quad n \to \infty.$$

Властивість збіжності інтерполяційного процесу залежить як від гладкості функції $f(x)$, так і від вибору послідовності сіток. Існують функції, для яких послідовність інтерполяційних поліномів, побудованих по будь-якій сітці з послідовності (5.32), не наближається зі зростанням n до $f(x)$ у жодній точці (інтерполяційний процес розходиться).

Більш загальне твердження сформульоване в теоремі Фабера: за будь-якої послідовності сіток S_n знайдеться неперервна на $[a, b]$ функція $f(x)$ така, що послідовність інтерполяційних поліномів $L_n[f(x)]$ не збігатиметься до $f(x)$ рівномірно на відрізку $[a, b]$.

І навпаки, можна навести приклади функцій, скажімо, цілих (тобто представлених у вигляді степеневого ряду), що сходяться рівномірно на будь-якій сітці (5.32). Тобто існують функції, для яких послідовність побудованих по сітках (5.32) інтерполяційних поліномів з елементами, що належать відрізку $[a, b]$, рівномірно збігається до $f(x)$.

На практиці намагаються не використовувати інтерполяційні поліноми високого степеня. Замість них застосовують кусково-поліноміальну інтерполяцію.

5.7. Інтерполяційні сплайни

Збільшуючи число точок інтерполяції, не завжди є можливість знизити похибку. Це має місце в тих випадках, коли проміжок $[a, b]$, на якому потрібно наблизити функцію $f(x)$ функцією $\varphi(x)$, великий і немає підстав вважати дану функцію $f(x)$ достатньо гладкою. Тоді немає сенсу підвищувати точність поліноміальної інтерполяції за рахунок використання поліномів високих степенів. У цих умовах більш перспективним є застосування кусково-поліноміальної інтерполяції.

Ідея даного підходу така: відрізок $[a, b]$ розбивається на кілька ділянок, а потім на кожній ділянці будується інтерполяційний поліном. Через те, що на окремих ділянках кількість вузлів буде меншою, ніж на всьому вихідному відрізку, інтерполювати можна буде за допомогою полінома меншого степеня, що значно спрощує задачу. При цьому звичайно потрібно, щоб у точках з'єднання сусідніх ділянок поліноми мали однакові значення. Як інтерполяційну функцію звичайно вибирають поліноми не вище третього-четвертого степеня. Але в цьому разі у точках з'єднання виникають розриви похідних. Позбутися цих недоліків можна за допомогою сплайнів. Найчастіше використовують кубічні сплайни. Поліном третього степеня будемо називати кубічним сплайном $S(x)$, що відповідає вихідній функції $f(x)$ і заданий на сітці впорядкованих вузлів $a = x_0 < x_1 < ... < x_n = b$, якщо задовольняються такі умови:

- на кожному відрізку $[x_{i-1}, x_i]$, $i = 1, 2, ..., n$ функція $S(x)$ є поліномом третього степеня;
- функція $S(x)$ і її перша і друга похідні неперервні на відрізку $[a, b]$;
- у вузлах інтерполяції $S(x_i) = f(x_i)$, $i = 0, 1, 2, ..., n$.

Доведемо існування і єдиність сплайна, що задовольняє ці умови. Доведення містить також і спосіб побудови кубічного сплайна.

Нехай $\{x_i\}$ — множина вузлів інтерполяції, $i = 1, 2, ..., n$. Між вузлами x_{i-1} і x_i задамо функцію:

$$S(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3,$$

$$x_{i-1} \leqslant x \leqslant x_i.$$

Перш знайдемо a_i. З того, що у вузлах інтерполяції $S(x_i) = f(x_i)$, $i = 1, 2, ..., n$, маємо $a_i = S(x_{i-1}) = f(x_{i-1})$, $i = 1, 2, ..., n$.

Перепишемо формулу для сплайна у вигляді:

$$S(x) = y_{i-1} + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3. \qquad (5.33)$$

Ця система містить $4n - n = 3n$ невідомих.

Будемо виводити формулу для рівновіддалених вузлів, коли: $x_i - x_{i-1} = h$.

Знайдемо значення функції в кінцевій точці як $y_n = y_{n-1} + b_n h + c_n h^2 + d_n h^3$. Це співвідношення дозволить позбутися однієї невідомої. Далі будемо використовувати похідні $S(x \to x_i)$. Запишемо формулу (5.33) для відрізка $[x_i, x_{i+1}]$:

$$S(x) = y_i + b_{i+1}(x - x_i) + c_{i+1}(x - x_i)^2 + d_{i+1}(x - x_i)^3. \tag{5.34}$$

Значення функцій (5.33) і (5.34) на суміжних із вузлом x_i інтервалах повинні збігатися, тобто:

$$y_i = y_{i-1} + b_i h + c_i h^2 + d_i h^3. \tag{5.35}$$

Одержимо n рівнянь. Запишемо значення першої похідної:

$$S'(x) = b_i + 2c_i(x - x_{i-1}) + 3d_i(x - x_{i-1})^2,$$
$$x_{i-1} \leqslant x \leqslant x_i;$$
$$S'(x) = b_{i+1} + 2c_{i+1}(x - x_i) + 3d_{i+1}(x - x_i)^2,$$
$$x_i \leqslant x \leqslant x_{i+1}.$$

Прирівняємо похідні в точці x_i:

$$b_{i+1} = b_i + 2c_i h + 3d_i h^2. \tag{5.36}$$

Таких формул буде $n - 1$.
Обчислимо другу похідну:

$$S''(x) = 2c_i + 6d_i(x - x_{i-1}),$$
$$x_{i-1} \leqslant x \leqslant x_i;$$
$$S''(x) = 2c_{i+1} + 3d_{i+1}(x - x_i),$$
$$x_i \leqslant x \leqslant x_{i+1}.$$

Прирівняємо значення цих похідних у точках x_i:

$$2c_{i+1} = 2c_i + 6d_i h. \tag{5.37}$$

Одержимо ще $n - 1$ рівнянь.
Порахуємо змінні, які ще не визначені:

$$3n - n - 2(n - 1) = 2.$$

Додамо умову:

$$\left.\begin{aligned} S''(x_0) &= 0, \\ S''(x_n) &= 0. \end{aligned}\right\} \tag{5.38}$$

Тепер кількість невідомих дорівнює кількості рівнянь. Перепишемо систему заново, використовуючи отримані співвідношення (5.35–5.38):

$$\left.\begin{aligned} y_i &= y_{i-1} + b_i h + c_i h^2 + d_i h^3, \\ b_{i+1} &= b_i + 2c_i h + 3d_i h^2, \\ c_{i+1} &= c_i + 3d_i h, \\ c_1 &= 0, \\ c_n &= -3d_n h. \end{aligned}\right\}$$

Доведемо, що систему можна перетворити так, щоб її можна було розв'язати методом прогону:

$$d_i = \frac{c_{i+1} - c_i}{3h} \text{ (підставимо в (5.35)),}$$

$$b_i h + c_i h^2 + \frac{c_{i+1} - c_i}{3} h^2 = y_i - y_{i-1},$$

$$b_i = \frac{y_i - y_{i-1}}{h} - \frac{h}{3}(c_{i+1} + 2c_i).$$

Вираз для b_{i+1} буде мати вигляд:

$$b_{i+1} = \frac{y_{i+1} - y_i}{h} - \frac{h}{3}(c_{i+2} + 2c_{i+1}).$$

Підставимо значення b_i, b_{i+1} і d_i і отримаємо:

$$\frac{y_{i+1} - y_i}{h} - \frac{h}{3}(c_{i+2} + 2c_{i+1}) = \frac{y_i - y_{i-1}}{h} - \frac{h}{3}(c_{i+1} + 2c_i) + 2c_i h + c_{i+1} - c_i.$$

Система розв'язується методом прогону, тому що матриця коефіцієнтів є тридіагональною. Відомо, що $c_1 = 0$, $c_n = -3d_n h$.

Приклад 5.6

Складемо програму, яка визначає сплайн-функцію для деякої таблиці TS, що містить більше точок, ніж задано в прикладі 5.1, у таблиці TA:

```
TS = {{0,   0}, {0.5, 2},    {1.,  2.25},
      {1.5, 3}, {2,   3.25}, {2.5, 3},
      {3,   6}, {3.5, 0.75}, {4,   3.75}};
```

Визначимо функцію у вигляді:

```
s[i_, x_, ta_]:= A[[i]] + B[[i]]*(x - ta[[i]]) +
                 G[[i]]*(ta[[i]])^2 + F[[i]]*(x - ta[[i,1]])^3;
```

Невідомі вектори коефіцієнтів визначимо зі співвідношень (5.35)–(5.38):

```
A = Table[TS[[i,2]], {i,1,n}];
B = Table[b[i], {i,1,n}];
G = Table[g[i], {i,1,n}];
F = Table[f[i], {i,1,n}];
```

Знайдемо вираз для лівих частин виведених умов (5.35)–(5.38). Із першої умови отримаємо значення коефіцієнтів $a_i = y_i$, $i = 1, 2, ..., n$. Усі невідомі коефіцієнти сплайн-функцій, що залишилися, можуть бути отримані після формування і розв'язання системи лінійних рівнянь. Будемо визначати сплайн-функцію з вільними кінцями, що задовольняє умови (5.38). З останніх умов випливає, що $f_1 = 0{,}2f_n + 6(x_{n+1} - x_n)g_n = 0$.
У результаті розв'язання побудованої системи отримаємо такі значення коефіцієнтів:

```
{b[1] → 2.45238,      b[2] → 5.54762,   b[3] → -3,          b[4] → 4.04762,
 b[5] → -2.45238,     b[6] → 4.90476,   b[7] → 16.1429,     b[8] → 33.7619,
 f[1] → 2.28719×10^-17, f[2] → 6.19048, f[3] → -7.80952,   f[4] → 2.38095,
 f[5] → -9.42857,     f[6] → -19.0476,  f[7] → 13.5238,     f[8] → -79.5238,
 g[1] → 3.09524,      g[2] → -13.1905,  g[3] → 12.9048,     g[4] → -8.28571,
 g[5] → 4.61905,      g[6] → 35.3333,   g[7] → -60.0476,    g[8] → 119.286
```

Варто врахувати, що формула (5.33) задає значення сплайн-функції тільки на відповідному відрізку $[x_{i-1}, x_i]$, $i = 1, 2, ..., n$. Поза цим відрізком покладемо її значення рівними нулю. Наступна формула визначає сплайн-функцію з урахуванням цієї умови:

```
spl[i_,x_]:= If[TS[[i,1]]<=x&&x<TS[[i+1,1]], s[i,x,TS], 0];
```

Сплайн-функція, що зображує інтерполяційну функцію на відрізку $[x_0, x_n]$, визначається такою формулою:

$$\text{Cubsp}[x\_] := \sum_{i=1}^{n} \text{spl}[i, x];$$

На закінчення наведемо графік функції (рис. 5.9), побудований за останньою формулою.

```
gr1 = Plot[Cubsp[y],{y,0,4}, Frame → True,
      GridLines → Automatic,
      PlotStyle → {Thickness[.01]}]
```

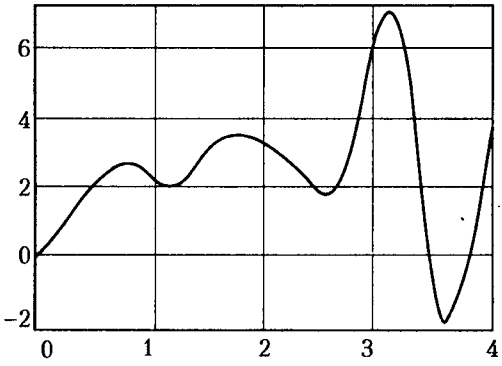


Рис. 5.9. Інтерполяція табличної функції кубічним сплайном

На основі тієї ж таблиці TS можна побудувати еквівалентний інтерполяційний поліном Лагранжа чи Ньютона. Наведемо на одному рисунку (рис. 5.10) графіки інтерполяцій-

ного полінома Лагранжа, і сплайн-функції, отриманої в прикладі 5.6, зобразивши її суцільною лінією. Значення збігаються у вузлах інтерполяції (відмічені на графіку точки) і помітно відрізняються в проміжках між ними через виникнення биття (викидів) у методі Лагранжа. Це ще раз підтверджує, що за великої кількості вихідних точок методи Ньютона, Лагранжа і т. п. стають непридатними, у таких випадках необхідно використовувати інтерполяцію сплайнами.

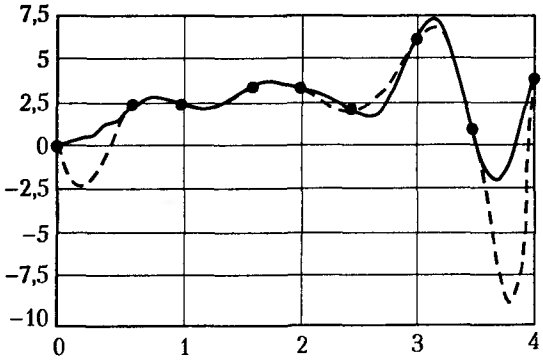


Рис. 5.10. Результати інтерполяції табличної функції поліномом Лагранжа (пунктирна лінія) і кубічним сплайном (суцільна лінія)

Під час побудови сплайн-функцій можна було використовувати й інші умови на кінцях відрізка. Наприклад, можна було б задати значення першої чи другої похідної в точках x_0 і x_n. У пакеті Mathematica табличну функцію можна зобразити за допомогою лінійного або кубічного сплайна. Для побудови сплайн-функції на основі таблиці TA застосовують такий оператор:

```
In[]:= Tp = Interpolation[Ts, InterpolationOrder → 3]
```

Значення параметра 3 вказує на вибір кубічного сплайна. Можна використовувати лінійний сплайн:

```
In[]:= Tpl = Interpolation[Ts, InterpolationOrder → 1]
```

Графіки обох сплайнів наведені на рис. 5.11 (точками позначені вихідні дані).

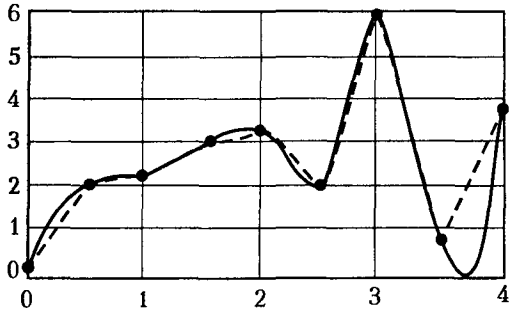


Рис. 5.11. Результати інтерполяції табличної функції лінійним сплайном (пунктирна лінія) і кубічним сплайном (суцільна лінія)

Більшість методів інтерполяції застосовується для наближення однозначних функцій, що задаються таблицею значень аргументу, який змінюється неперервно.

Для інтерполяції однозначних і неоднозначних функцій у Mathematica використовується оператор SplineFit[data, tips], що міститься в стандартній бібліотеці й викликається командою

```
In[]:= <<NumericalMath`SplineFit`
```

Приклад 5.7

Введемо таблицю, що зображує неоднозначну функцію:

```
In[]:= Tb = {{1.15, -0.3}, {1.2, 1},   {1.5, 3.1},
            {1.9,   2.4}, {1.6, 0.9}, {1.3, 0.1}, {0.8, -0.4}};
```

Використаємо оператор для визначення інтерполяційної функції за допомогою кубічних сплайнів:

```
In[]:= Pls = SplineFit[Tb,Cubic]
Out[]= SplineFunction[Cubic,{0.,6},<>]
```

Отримана сплайн-функція задає значення змінних (x, y) параметрично, в інтервалі [0, 6], що заданий у повідомленні. Значення змінних (x, y) можуть бути отримані, якщо задати значення параметра, наприклад:

```
In[]:= Pls[0]{1.15, -0.3} Pls[1.5]{1.3081, 2.19639} Pls[3]{1.9, 2.4}
       Pls[5]{1.3,   0.1} Pls[6]{0.8, -0.4}
```

Графік функції (рис. 5.12) можна побудувати за допомогою таких операторів:

```
In[]:= gs1 = ParametricPlot[Pls[u], {u, 0, 6}, PlotRange → All,
             Compiled → False, DisplayFunction → Identity]
       rg =  ListPlot[Tb, PlotStyle → PointSize[0.03],
             DisplayFunction → Identity]
             Show[gs1, rg, DisplayFunction → $DisplayFunction]
```

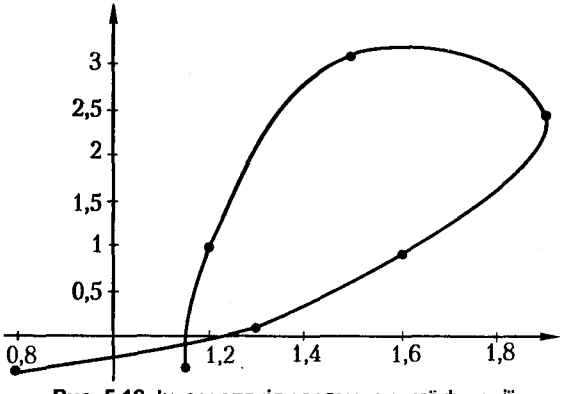


Рис. 5.12. Інтерполяція параметричної функції

На графік нанесені точки інтерполяції (рис. 5.12). Зростанню параметра відповідає обхід кривої за годинниковою стрілкою.

Інтерполяція кубічними сплайнами є процесом, що сходиться, тобто за необмеженого збільшення числа вузлів n відповідна послідовність сплайн-функцій

збігається до функції $f(x)$, що інтерполюється. Оцінка похибки інтерполяції $r(x) = f(x) - S(x)$ залежить від вибору сіток і від гладкості $f(x)$. Якщо зажадати від функції $f(x)$ існування неперервної на відрізку $[a, b]$ четвертої похідної і припустити, що виконано граничної умови $f'(a) = f'(b) = 0$, а також збережені такі ж умови для сплайна, то можна одержати оцінку залишкового члена у вигляді:

$$r(x) = |f(x) - S_h(x)| \leqslant \frac{M_4 h^4}{8}, \tag{5.39}$$

де $M_4 = \max\limits_{x \in [a,b]} |f^{IV}(x)|$.

5.8. Метод найменших квадратів

Характерним для задач, під час розв'язання яких використовується метод найменших квадратів, є те, що початкові дані для виявлення тих чи інших закономірностей завжди наближені через неточність вимірювальних приладів, неповторність умов спостережень, випадкові помилки і т. д. [16, 18]. Метод найменших квадратів застосовується для відновлення функції або заміни складної функції більш простою, а також для обробки експериментальних даних і розв'язування перевизначених систем лінійних алгебраїчних рівнянь.

Розглянемо задачу відновлення деякої функції $f(x)$ методом найменших квадратів. Зажадаємо, щоб міра відхилення експериментальних значень від обраної функції було мінімальною у заданих $n+1$ точках (x_i, y_i), $i = 0, 1, 2, ..., n$:

$$I = \sum_{i=0}^{n} (f(x_i) - \varphi(x_i))^2 \to \min. \tag{5.40}$$

Найчастіше як $\varphi(x)$ обирають поліноми. Тоді говорять, що коли функція $\varphi(x)$ задовольняє рівняння (5.40), то вона є поліномом найкращого середньоквадратичного наближення:

$$\varphi(x) = a_0 + a_1 x + a_2 x^2 + ... + a_k x^k, \quad k < n, \tag{5.41}$$

де $a_0, a_1, ..., a_k$ — довільні дійсні числа (коефіцієнти узагальненого полінома). Таким чином, побудова узагальненого полінома найкращого середньоквадратичного наближення для даної функції $f(x)$ зводиться до знаходження оптимального набору коефіцієнтів для функцій $\varphi(x)$ у рівнянні (5.41) на основі методу найменших квадратів, тобто до розв'язання задачі мінімізації (5.42).

$$I = \sum_{i=0}^{n} [a_0 + a_1 x_i + a_2 x_i^2 + ... + a_k x_i^k - y_i]^2 \to \min. \tag{5.42}$$

Якщо $k = n$, то розглядається задача інтерполяції, при цьому поліном $\varphi(x)$ збігається з інтерполяційним поліномом Лагранжа, побудованим на тих же вузлах.

У нашому формулюванні за фіксованого k міра відхилення $I(a_0, a_1, ..., a_k)$ є функцією $k+1$ аргументів.

Необхідно, щоб

$$\frac{\partial I}{\partial a_j} = 0, \quad \forall j = 0, 1, 2, ..., k.$$

Починаємо обчислювати похідні:

$$\frac{\partial I}{\partial a_0} = 2\sum_{i=0}^{n}[a_0 + a_1 x_i + a_2 x_i^2 + ... + a_k x_i^k - y_i] \cdot 1 = 0.$$

Склавши отримані співвідношення, знаходимо:

$$\sum_{i=0}^{n} a_0 + a_1 \sum_{i=0}^{n} x_i + a_2 \sum_{i=0}^{n} x_i^2 + ... + a_k \sum_{i=0}^{n} x_i^k = \sum_{i=0}^{n} y_i.$$

Введемо заміну змінних:

$$\overline{x} = \frac{\sum_{i=0}^{n} x_i}{n+1}.$$

Тоді попередній вираз можна записати в такому вигляді:

$$(n+1)a_0 + a_1(n+1)\overline{x} + a_2(n+1)\overline{x}^2 + ... + a_k(n+1)\overline{x}^k = (n+1)\overline{y},$$
$$a_0 + a_1\overline{x} + a_2\overline{x}^2 + ... + a_k\overline{x}^k = \overline{y}.$$

Знайдемо такі рівняння:

$$\frac{\partial I}{\partial a_1} = 2\sum_{i=0}^{n}[a_0 + a_1 x_i + a_2 x_i^2 + ... + a_k x_i^k - y_i] x_i = 0.$$

Розкриємо дужки

$$a_0 \sum_{i=0}^{n} x_i + a_1 \sum_{i=0}^{n} x_i^2 + a_2 \sum_{i=0}^{n} x_1^3 + ... + a_k \sum_{i=0}^{n} x_i^{k+1} = \sum_{i=0}^{n} x_i y_i$$

і отримаємо:

$$a_0\overline{x} + a_1\overline{x}^2 + a_2\overline{x}^3 + ... + a_k\overline{x}^{k+1} = \overline{xy},$$

де

$$\overline{xy} = \frac{\sum_{i=0}^{n} x_i - y_i}{n+1}.$$

Послідовно обчислимо всі похідні:

$$\frac{\partial I}{\partial a_j} = 2\sum_{i=1}^{n}[a_0 + a_1 x_i + a_2 x_i^2 + ... + a_j x_i^j + ... + a_k x_i^k - y_i]x_i^j = 0,$$

$$\forall j = 2, 3, ..., k.$$

Після перетворення одержимо систему $k+1$ рівнянь:

$$a_0\overline{x}^j + a_1\overline{x}^{j+1} + a_2\overline{x}^{j+2} + ... + a_k\overline{x}^{j+k} = \overline{x}^j\overline{y}. \qquad (5.43)$$

Якщо єдиний розв'язок існує, то можна обчислити коефіцієнти полінома найкращого середньоквадратичного наближення.

Приклад 5.8

Нехай задана функціональна залежність двома таблицями ta = $\{x_i\}$ і tb = $\{y_i\}$

```
ta = {1.1, 1.2, 1.5, 2,   4,   2.7, 2.4};
tb = {1,   2.5, 3.4, 4.3, 5.1, 4.2, 3.7};
```

Наведемо приклад програми, що визначає по заданих таблицях поліном найкращого середньоквадратичного наближення. Задамо кількість коефіцієнтів системи (степінь полінома буде на одиницю менше) і опишемо допоміжний масив для обчислення коефіцієнтів системи (5.43):

```
m = 4; S = Array[s, {m,m}]; n = Length[ta];
```

Обчислимо допоміжні значення:

```
In[]:= X = Table[Sum[ta[[i]]^j/n, {i, n}], {j, 0, 2*(m-1)}]
Out[]= {1, 2.12857, 5.42143, 15.9916, 52.4174, 184.401, 679.239}
```

Утворимо з них матрицю системи рівнянь:

```
In[]:= TA = Table[s[i,j] = X[[i + j-1]], {i, m}, {j, m}];
```

Наведемо її для ілюстрації:

```
In[]:= TableForm[TA]
Out[]//TableForm=
```

1	2.12857	5.42143	15.9916
2.12857	5.42143	15.9916	52.4174
5.42143	15.9916	52.4174	184.401
15.9916	52.4174	184.401	679.239

Обчислимо праві частини системи рівнянь:

```
In[]:= b = Table[Sum[ta[[i]]^j*tb[[i]]/n, {i, n}], {j, 0, (m-1)}]
Out[]= {3.45714, 8.34571, 23.3129, 73.1062}
```

і отримаємо її розв'язок:

```
In[]:= A = Inverse[TA] . b
Out[]= {-13.4621, 21.5347, -8.4563, 1.05846}
```

Запишемо отриманий поліном:

```
In[]:= p[x_]: = Sum[A[i]]*x^(i-1), {i, m}; p[x]
Out[]= -13.4621 + 21.5347 x - 8.4563 x^2 + 1.05846 x^3
```

Побудуємо графік полінома, на який нанесемо задані табличні значення. Складемо з двох таблиць TA і TB звичайну таблицю $\{x_i, y_i\}$:

```
In[]:= TT = Transpose[{ta, tb}]
Out[]= {{1.1, 1}, {1.2, 2.5}, {1.5, 3.4}, {2, 4.3}, {4, 5.1},
        {2.7, 4.2}, {2.4, 3.7}}
```

Побудуємо графік табличної функції, графік отриманого полінома і об'єднаємо їх (рис. 5.13):

```
In[]:= g1 = ListPlot[TT, PlotStyle → PointSize[0.03], DisplayFunction → Identity]
       g2 = Plot[p[x], {x, 1.1, 4}, DisplayFunction → Identity]
            Show[g1, g2, GridLines → Automatic, Frame → True
            FrameLabel → {"x", "p(x)"}, DisplayFunction → $DisplayFunction]
```

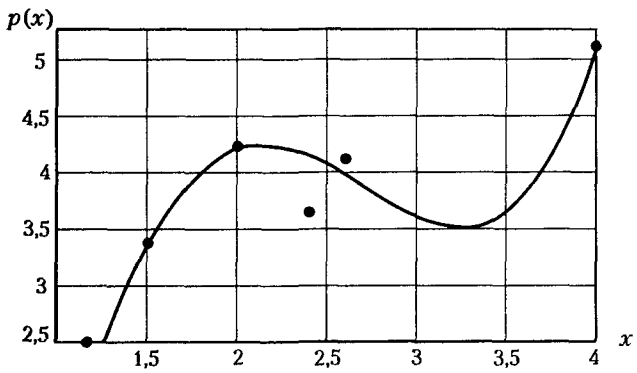


Рис. 5.13. Наближення табличної функції з прикладу 5.8 методом найменших квадратів

Знайдемо відносну середньоквадратичну помилку

```
In[]:= Is = Sum[(p[TT[[i,1]]] - TT[[i,2]])^2/n, {i, 1, n}]
Out[]= 0.0989264
```

Усю послідовність обчислень можна значно скоротити, якщо скористатися стандартним оператором пакета Mathematica PolinomialFit, що також будує поліном найкращого квадратичного наближення. Різниця між ними полягає в тому, що останній оператор зображує поліном по степенях $(x - x_{ср})$ відхилення змінної x від її середнього табличного значення $x_{ср}$. Виклик пакета виконується відповідно до інструкції:

```
In = <<NumericalMath`PolynomialFit`
```

Наведемо приклад його використання. Введемо таблицю:

```
In[]:= TB = {{1.2, 2.1}, {1.4, 3.5}, {1.7, 4.3}, {1.9, 3.2}, {2.3, 2.1}, {2.5, 0.8}};
       n = Length[ta];
```

і одержимо поліном найкращого квадратичного наближення третього степеня:

```
In[]:= p = PolynomialFit[TB, 3]
Out[]= FittingPolynomial[<>, 3]
```

Наведемо його:

```
N[p[x], 6]
```

```
3.78964 - 1.7454 (-1.83333 + x) + 1.08536 (-0.69067 +
2.17072 (-2.32879 + 1.7454 (-1.83333 + x)) (-1.83333 + x)) (-1.83333 + x)
```

Покажемо, що середнім табличним значенням є:

```
xs= Sum[TB[[i,1]]/6, {i, 1, 6}]
```

```
1.83333
```

Отримаємо на основі даних таблиці TB поліном третього степеня за допомогою оператора Fit:

```
f1 = Fit[tb, {1, x, x^2, x^3}, x]
```

$$-35.417 + 59.0875\,x - 28.1038\,x^2 + 4.11221\,x^3$$

Покажемо, що він збігається з поліномом p[x]:

```
p1 = Expand[p[x]]
```

$$-35.417 + 59.0875\,x - 28.1038\,x^2 + 4.11221\,x^3$$

Побудуємо графік полінома, покажемо на ньому табличні дані і порівняємо їх (рис. 5.14):

```
g2 = Plot[p1[x], {x, 1.2, 2.5}, DisplayFunction → Identity]
g1 = ListPlot[TB, PlotStyle → PointSize[0.03], DisplayFunction → Identity]
     Show[g1, g2, GridLines → Automatic, Frame → True,
     FrameLabel → {"x", "p(x)"}, DisplayFunction → $DisplayFunction]
```

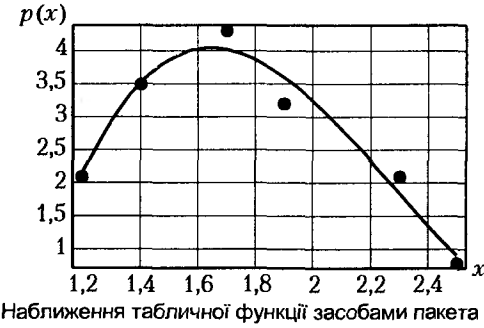


Рис. 5.14. Наближення табличної функції засобами пакета Mathematica

5.9. Метод рівнянь у нормальній формі

Альтернативою методу найменших квадратів може бути метод, що базується на використанні скалярних добутків базисних функцій. Якщо кожне рівняння із системи (5.3) помножити на відповідну базисну функцію $\varphi_j(x)$, $j = 0, ..., m$, отримаємо систему рівнянь у так званій *нормальній* формі:

$$\begin{aligned}
(\varphi_0, \varphi_0)c_0 + (\varphi_0, \varphi_1)c_1 + \ldots + (\varphi_0, \varphi_m)c_m &= (\varphi_0, f),\\
(\varphi_1, \varphi_0)c_0 + (\varphi_1, \varphi_1)c_1 + \ldots + (\varphi_1, \varphi_m)c_m &= (\varphi_1, f),\\
\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots&\\
(\varphi_m, \varphi_0)c_0 + (\varphi_m, \varphi_1)c_1 + \ldots + (\varphi_m, \varphi_m)c_m &= (\varphi_m, f),
\end{aligned} \tag{5.44}$$

яка налічує $m+1$ рівнянь відносно невідомих коефіцієнтів c_j, $j = 0, 1, ..., m$ за значеннями функції f, вибраної на множині точок x_i, $i = 0, 1, 2, ..., n$ для будь-якого $n \geqslant m$. Важливо підкреслити, що збільшення кількості вузлів інтерполяції не впливає на розмірність системи рівнянь (5.44), а позначається тільки на точності визначення скалярних добутків функцій, що входять до рівнянь (5.44) як сталі.

Коли в якості базисних функцій обрані ортогональні поліноми (Лежандра, Чебишева, Ерміта та ін.), для яких виконуються умови $(\varphi_i, \varphi_j) = 0, i \neq j$, $\|\varphi_i\| \neq 0$, обчислення коефіцієнтів c_j істотно спрощується:

$$c_j = \frac{(f, \varphi_j)}{(\varphi_j, \varphi_j)}. \tag{5.45}$$

При цьому похибка ортогональна до всіх φ_j, $j = 0, 1, ..., m$, тобто

$$\left(\sum_{j=0}^{m} c_j \varphi_j - f, \varphi_k\right) = 0, \quad k = 0, 1, 2, ..., m. \tag{5.46}$$

Дані щодо найбільш широко застосовуваних ортогональних поліномів зведені в табл. 5.1.

Таблиця 5.1. Властивості ортогональних поліномів

Властивості	Полином Лежандра	Полином Чебишева	Полином Ерміта
Область визначення	$-1 \leqslant x \leqslant 1$	$-1 \leqslant x \leqslant 1$	$-\infty \leqslant x \leqslant \infty$
φ_0	$P_0(x) = 1$	$T_0(x) = 1$	$H_0(x) = 1$
φ_1	$P_1(x) = x$	$T_1(x) = x$	$H_1(x) = x$
φ_2	$P_2(x) = 1/2(3x^2 - 1)$	$T_2(x) = 2x^2 - 1$	$H_0(x) = 4x^2 - 2$
Рекурентна формула обчислення наступних поліномів	$P_{n+1}(x) = \frac{2n+1}{n+1} x P_n(x) - \frac{n}{n+1} P_{n-1}(x)$	$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$	$H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x)$
Загальна формула опису	$P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n}\left[(x^2-1)^n\right]$	$T_n(x) = \cos(n \arccos(x))$	$H_n(x) = (-1)^n e^{x^2} \cdot \frac{d^n}{dx^n} e^{-x^2}$

Спільною рисою ортогональних поліномів є їх трикутна форма:

$$\begin{cases} \varphi_0(x) = a_{00}, \\ \varphi_1(x) = a_{10} + a_{11}x, \\ \varphi_2(x) = a_{20} + a_{21}x + a_{22}x^2, \\ \quad ... \quad ... \quad ... \quad ... \quad ... \\ \varphi_m(x) = a_{m0} + a_{m1}x + a_{m2}x^2 + ... + a_{mm}x^m. \end{cases}$$

Тому в загальному випадку можна записати співвідношення:

$$\varphi^*(x) = \mathbf{A}\mathbf{x}^*, \tag{5.47}$$

де

$$\varphi^*(x) = [\varphi_0(x), \varphi_1(x), ..., \varphi_m(x)], \quad \mathbf{x}^* = [x^0, x^1, x^2, ..., x^m]^{\mathrm{T}}.$$

$\mathbf{A} = \{a_{ij}\}$, $i, j = 0, 1, ..., m$ — нижня трикутна матриця порядку $(m+1)$.
Розв'язавши обернену задачу, знайдемо:

$$\mathbf{x}^* = \mathbf{A}^{-1}\varphi^*(x) = \mathbf{B}\varphi^*(x), \tag{5.48}$$

звідки

$$\begin{cases} x^0 = b_{00}\varphi_0, \\ x^1 = b_{10}\varphi_0 + b_{11}\varphi_1, \\ x^2 = b_{20}\varphi_0 + b_{21}\varphi_1 + b_{22}\varphi_2, \\ \quad ... \quad ... \quad ... \quad ... \quad ... \\ x^m = b_{m0}\varphi_0 + b_{m1}\varphi_1 + ... + b_{mm}\varphi_m. \end{cases}$$

Тобто, будь-який поліном n-го степеня може бути записаний через сукупність базисних ортогональних поліномів:

$$p(x) = c_0\varphi_0(x) + c_1\varphi_1(x) + ... + c_m\varphi_m(x). \tag{5.49}$$

Це дозволяє проводити наближення функцій ортогональними поліномами за допомогою процедури (5.44), а потім привести це наближення до звичної поліноміальної форми.

У разі застосування поліномів Лежандра і Чебишева слід здійснити перехід від реального інтервалу інтерполяції $[a, b]$ до області визначення ортогональних поліномів за допомогою заміни змінних $x = a_0 + a_1 x_d$, а також переобчислити вибрані значення $f(x)$ у значення $f_1(x_d)$. Враховуємо, що на нижній межі інтервалу $x = a$ і $x_d = -1$, тому $x = a_0 + a_1(-1)$ на верхній межі інтервалу $x = b$ і $x_d = 1$, у результаті $x = a_0 + a_1(1)$.
Отже, отримуємо:

$$a_0 = \frac{b+a}{2} \quad \text{і} \quad a_1 = \frac{b-a}{2},$$

тому

$$x_d = \frac{(b+a) - 2x}{b-a} \quad \text{і} \quad x = \frac{(b+a) - (b-a)x_d}{2}.$$

Приклад 5.9

Апроксимуємо лінійною функцією $f^*(x) = c_0 + c_1 x$ дані таблиці:

x	1	3	4	6	7
$f(x)$	–2,1	–0,9	–0,6	0,6	0,9

У нашому випадку $\varphi_0(x) = 1, \varphi_1(x) = x$, тобто кількість невідомих коефіцієнтів дорівнює двом ($m = 2$), а кількість вузлів інтерполяції дорівнює п'яти ($n = 5$).
Система рівнянь (5.44) у нормальній формі набуває вигляду:

$$\left.\begin{aligned}(\varphi_0, \varphi_0)c_0 + (\varphi_0, \varphi_1)c_1 &= (\varphi_0, f),\\ (\varphi_1, \varphi_0)c_0 + (\varphi_1, \varphi_1)c_1 &= (\varphi_1, f).\end{aligned}\right\}$$

Необхідні скалярні добутки визначаються векторами:

```
In[]:= x = {1, 3, 4, 6, 7};
       f = {-2.1, -0.9, -0.6, 0.6, 0.9};
       φ[0] = {1, 1, 1, 1, 1};
       φ[1] = {1, 3, 4, 6, 7};
```

за допомогою яких послідовно знаходимо потрібні скалярні добутки:

```
In[]:= Do [Do[A[i,j] =φ[i-1] . φ[j-1], {j,2}]; B[i] = f . φ [i-1], {i,2}];
         Array[A, {2,2}]
Out[]= {{5, 21}, {21, 111}}

In[]:= Array[B,2]
Out[]= {-2.1, 2.7}
```

Розв'язок системи лінійних рівнянь буде мати вигляд:

```
In[]:= LinearSolve[Array[A, {2,2}], Array[B,2]]
Out[]= {-2.54211, 0.505263},
```

тобто шукані коефіцієнти апроксимації $c_0 = -2{,}54211$, $c_1 = 0{,}505263$.
Перевірка точності наближення дає похибку:

```
In[]:= -2.54211* φ[0] + 0.505263* φ[1] - f
Out[]= {0.063153, -0.126321, 0.078942, -0.110532, 0.094731},
```

яка свідчить, що пряма наближення майже однаково віддалена від ординат вибраних точок (рис. 5.15).

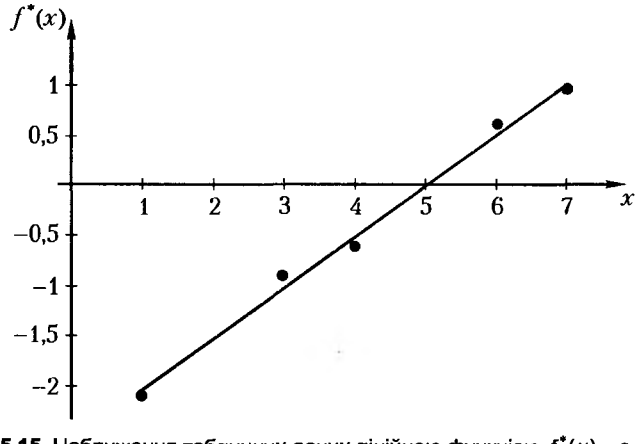


Рис. 5.15. Наближення табличних даних лінійною функцією $f^*(x) = c_0 + c_1x$

Приклад 5.10

Знайдемо апроксимуючу функцію для табличних даних, що відображують коливання рівня води в Північному морі через припливи і відливи:

t, год	0	2	4	6	8	10
h, м	1,0	1,6	1,4	0,6	0,2	0,8

Оскільки мова йде про коливальний процес із періодом коливань $T = 12$ год, то виконаємо апроксимацію через тригонометричні функції:

$$h(t) = a_0 + a_1 \sin\frac{2\pi t}{T} + a_2 \cos\frac{2\pi t}{T}.$$

У нашому випадку $\varphi_0 = 1$; $\varphi_1 = \sin(x)$, $\varphi_2 = \cos(x)$, тобто $m = 3$, де $x = 2\pi t/T$, і значення базисних функцій в окремих точках спостережень ($n = 6$) приведені нижче:

	0 (0 год)	**$\pi/3$ (2 год)**	**$2\pi/3$ (4 год)**	**π (6 год)**	**$4\pi/3$ (8 год)**	**$5\pi/3$ (10 год)**
sin(x)	0	$\sqrt{3}/2$	$\sqrt{3}/2$	0	$-\sqrt{3}/2$	$-\sqrt{3}/2$
cos(x)	1	1/2	−1/2	−1	−1/2	1/2

Реалізуємо цей приклад за допомогою Mathematica, визначивши

```
In[]:= TB = {{0, 1}, {2, 1.6}, {4, 1.4}, {6, 0.6}, {8, 0.2}, {10, 0.B}};
       φ[0] = {1,   1,      1,     1,    1,        1};
       φ[1] = {0,  √3/2,  √3/2,  0,  -√3/2,  -√3/2 };
       φ[2] = {1,  1/2,   -1/2,  -1,  -1/2,   1/2 };
       h = {1,   1.6,    1.4,   0.6,   0.2,    0.8};
```

Система рівнянь (5.44) у нормальній формі набуває вигляду:

$$\begin{cases} (\varphi_0, h) = (\varphi_0, \varphi_0)h + (\varphi_0, \varphi_1)a_1 + (\varphi_0, \varphi_2)a_2, \\ (\varphi_1, h) = (\varphi_1, \varphi_0)h + (\varphi_1, \varphi_1)a_1 + (\varphi_1, \varphi_2)a_2, \\ (\varphi_2, h) = (\varphi_2, \varphi_0)h + (\varphi_2, \varphi_1)a_1 + (\varphi_2, \varphi_2)a_2. \end{cases}$$

Далі послідовно обчислимо потрібні скалярні добутки:

```
In[]:= Do [Do [A[i,j] = φ[i-1] . φ[j-1], {j, 3} ]; B[i] = h . φ [i-1], {i,3}]; Array[A, {3, 3}]
Out[]= {{6, 0, 0}, {0, 3, 0}, {0, 0, 3}}

In[]:= Array[B, 3]
Out[]= {5.6, 1.73205, 0.8}
```

Розв'язок системи лінійних рівнянь для обчислення невідомих коефіцієнтів буде мати вигляд:

```
In[]:= LinearSolve[Array[A, {3, 3}], Array[B, 3]]
Out[]= {0.933333, 0.57735, 0.266667}
```

Таким чином, шуканими коефіцієнтами є:

$$a_0 = 0{,}9333(3), \quad a_1 = 0{,}55735, \quad a_2 = 0{,}266665.$$

На основі отриманих значень побудуємо графік (рис. 5.16) за допомогою функції

```
g[x_]:=0.9333333 + 0.57735*Sin[2*π*x/T] + 0.266667*Cos[2*π*x/T]; T = 12;
```

зобразивши на ньому початкові дані.

```
gl = ListPlot[TB, PlotStyle → PointSize[0.03], DisplayFunction → Identity];
g2 = Plot[g[x], {x, 0, 10}, DisplayFunction → Identity, PlotStyle → {Thickness[.0075]}];
Show[gl, g2, AxesLabel → {"x", "g(x)"}, DisplayFunction → $DisplayFunction]
```

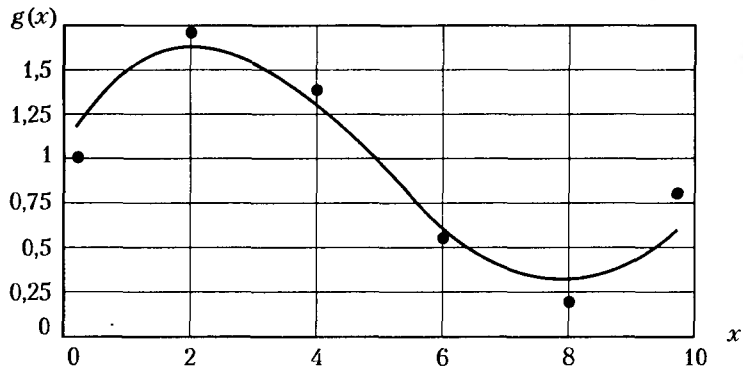


Рис. 5.16. Апроксимація табличних даних за допомогою тригонометричної функції

Похибка апроксимації визначається у такий спосіб

```
0.933333*φ[0] + 0.57735*φ[1] + 0.266665*φ[2] - h
{0.2, -0.0333333, -0.1, 0.0666667, 0.1, -0.233333}
```

Відносна квадратична норма дорівнює 0,0296355.

5.10. Засоби пакета Mathematica для розв'язання задач наближення функцій

У пакеті Mathematica для розв'язання задач наближення функцій передбачено оператор `Fit[data, funs, vars]`, що базується на реалізації методу найменших квадратів. У його форматі `data` — масив вхідних табличних даних у формі $\{\{x_1, f_1\}, \{x_2, f_2\}, ...\}$ чи у формі $\{f_1, f_2\}$, якщо аргумент функції x_i набуває значення 1, 2, 3, ...; `funs` — перелік базисних функцій φ_i, що застосовуються для апроксимації; `var` — перелік змінних за апроксимації. Якщо як базисні функції використовуються функції e^{a+bx}, то оператор наближення змінюється на `Exp[Fit[Log[data], {1,x}, x]]`.

Наприклад, обчислення з прикладу 5.9 можна повторити за допомогою такого запису, в якому базисні функції задаються у вигляді $\varphi_0(x) = 1, \varphi_1(x) = x$:

```
In[]:= <<NumericalMath`Approximations`
In[]:= p = {{1, -2.1}, {3, -0.9}, {4, -0.6}, {6, 0.6}, {7, 0.9}};
 (*таблиця значень аргумента і функції, тобто координат виділених точок*)
In[]:= Fit[p, {1, x}, x]
Out[]= -2.54211 + 0.505263 x
```

Нагадаємо, що в прикладі 5.9 були отримані ті ж самі коефіцієнти $c_0 = -2,54211$, $c_1 = 0,505263$.

Для випадку вживання ортогональних поліномів, розглянутому в прикладі 5.10, що стосується коливання рівня води в Північному морі, обчислення виконуються за таким записом, в якому враховується вибір базисних функцій у вигляді $\varphi_0 = 1$, $\varphi_1 = \sin(x)$, $\varphi_2 = \cos(x)$:

```
In[]:= {0., 1.0472, 2.0944, 3.14159, 4.18879, 5.23599}  (* значення аргумента в радіанах *)
In[]:= pk = {{0., 1}, {1.0472, 1.6}, {2.0944, 1.4}, {3.14159, 0.6},
            {4.18879, 0.2}, {5.23599, 0.8}};          (* значення координат виділених точок *)
In[]:= kkk= Fit[pk, {1, Sin[x], Cos[x]}, x]
Out[]= 0.933333 + 0.577351 Sin[x] + 0.266667 Cos[x]
```

Для порівняння наведемо коефіцієнти, отримані в прикладі 5.10: $a_0 = 0,933333$, $a_1 = 0,55735$, $a_2 = 0,266667$.

Якщо передбачається застосування для апроксимації полінома n-го степеня, то краще перейти до іншого запису оператора `Fit[data, Table[x^i,{i, 0, n}], x]` чи взагалі відмовитися від переліку базисних функцій і використовувати оператор `PolynomialFit [data, n]`. Тоді задачу, наведену в прикладі 5.9, можна ще раз розв'язати таким чином:

```
In[]:= <<NumericalMath`PolynomialFit`
In[]:= p = {{1, -2.1}, {3, -0.9}, {4, -0.6}, {6, 0.6}, {7, 0.9}};
In[]:= q = PolynomialFit[p,1]
Out[]= FittingPolynomial[<>, 1]
In[]:= Expand[q[x]]
Out[]= -2.54211 + 0.505263x
```

Ще один оператор пакета Mathematica заслуговує на особливу увагу. Йдеться про оператор `RationalInterpolation[f, {x,m,n}, {x1,x2,...,xm+ n+1}]`, за допомогою якого деяка функція f апроксимується дробово-раціональним виразом у вигляді частки двох поліномів, при цьому степінь полінома чисельника дорівнює m, а полінома знаменника — n. Під час обчислення використовується інформація в точках $x_1, x_2, ..., x_{m+n+1}$. Наприклад, для функції `Exp[-x/10]` у разі вибору чотирьох точок {0, 0,2, 0,4, 0,6} можлива апроксимація у вигляді частки полінома першого степеня до полінома другого степеня:

```
In[]:= <<NumericalMath`Approximations`
In[]:= RationalInterpolation[Exp[-x/10], {x, 1, 2}, {0, 0.2, 0.4, 0.6}]
                1. - 0.0330022 x
Out[]= ---------------------------------
       1 + 0.0669977 x + 0.00170039 x^2
```

Можливості процедур наближення функцій пакета Mathematica можна проілюструвати на прикладі побудови інтерполяційного полінома для функції $f(x) = 1 + 2e^{-x/3}$, яку спочатку переводимо в табличну форму, а потім поновлюємо через процедуру апроксимації:

```
In[]:= ft = Table[N[1 + 2 Exp[-x/3]], {x, 10}]
Out[]= {2.43306, 2.02683, 1.73576, 1.52719, 1.37775,
        1.27067, 1.19394, 1.13897, 1.09957, 1.07135}
```

```
In[]:= Fit[ ft, {1, Exp[-x/3]}, x]
Out[]= 1. + 2. e^(-x/3)
```

Оператор інтерполяції InterpolatingPolynomial [data, x], який базується на інтерполяційній формулі Ньютона, і SplineFit[data, Cubic], який використовує сплайни, — розглядалися раніше в прикладах 5.4 і 5.6. Слід відзначити: на відміну від випадку застосування оператора Fit[data, Table[x^i, {i, 0, n}], x] степінь полінома в разі використання оператора InterpolatingPolynomial [data, x] не вказується явно, а визначається розміром масиву data, тобто кількістю вибраних вузлів. Наприклад, для трьох вузлів формується поліном другого степеня, який точно відображає координати вузлів:

```
In[]:= <<NumericalMath`Approximations`
In[]:= InterpolatingPolynomial[{{-1, 4}, {0, 2}, {1, 6}}, x]
Out[]= 4 + (1 + x) (-2 + 3 x)
In[]:= Expand[%]
Out[]= 2 + x + 3 x^2
In[]:= % /. x → 0
Out[]= 2
```

Висновки

1. Для наближення функцій з метою спрощення обчислювальних процедур звичайно застосовують поліноміальну інтерполяцію, при цьому визначення коефіцієнтів полінома істотно полегшується, коли як базисні функції застосовують ортогональні поліноми.
2. Локальна апроксимація нелінійних функцій, яка віднесена в розділі 1 до базових принципів побудови чисельних методів, реалізується за допомогою методів наближення, розглянутих у даному розділі. Наприклад, для розв'язання нелінійних рівнянь (розділ 6) у методі Ньютона використовується лінійна локальна апроксимація (або поліном першого порядку), а в методі Мюллера — параболічна апроксимація (або поліном другого порядку).
3. Наближення функцій застосовується також для розв'язання крайової задачі для систем звичайних диференціальних рівнянь (метод колокацій, метод Гальоркіна) і для систем диференціальних рівнянь з частинними похідними (метод скінченних елементів).
4. Існує тільки один поліном n-го порядку, що проходить через всі $n+1$ задані точки (або точно відображає всі $n+1$ задані дискретні значення функції).
5. Процедура інтерполяції дає змогу знаходити проміжні значення для дискретно заданих або ж обчислених функцій і тим самим зменшувати обсяг експериментальних даних під час формування вхідних даних для програм математичного моделювання, а також обсяг обчислень у разі визначення та візуалізації результатів моделювання.

6. Інтерполяційний поліном Ньютона, під час побудови якого використовуються розділені різниці вищих порядків, має аналогічну з формулою Тейлора форму запису й аналогічну оцінку похибки, але застосовується для дискретно заданих функцій. Його зручно використовувати у випадках, коли степінь полінома, що апроксимує розглянуту функцію, апріорі невідомий. Вибираючи різне число членів інтерполяційного полінома, можна перевірити різні гіпотези щодо степеня полінома і, порівнюючи результати й оцінки похибки, вибрати найкращий варіант.
7. Поліном Лагранжа є, по суті, альтернативною формою зображення інтерполяційного полінома Ньютона, в якому не потрібне обчислення і запам'ятовування розділених різниць. Його зручно застосовувати у випадках, коли степінь полінома, що апроксимує розглянуту функцію, апріорі відомий.
8. Якщо число дискретних значень функції n перевищує степінь апроксимуючого полінома n, то великої точності наближення можна досягти, застосовуючи сплайн-функції низьких порядків і «припасовуючи» як їхні значення, так і значення їхніх похідних у суміжних точках.
9. Для того ж випадку, коли число дискретних значень функції n перевищує степінь апроксимуючого полінома m, високу точність наближення можна досягти, застосовуючи метод найменших квадратів. У результаті отримуємо функцію, яка не проходить через задані точки, але відрізняється від заданих значень функції з приблизно однаковою похибкою.
10. У реальних системах математичного моделювання обчислення ведуться зі змінним кроком, тому дискретні значення функцій розміщуються нерівномірно. У цих випадках застосовуються процедури наближення для функцій з нерівномірно розподіленими точками відліку.

Контрольні запитання та завдання

1. У таблиці наведені дані американського бюро перепису населення, що характеризують чисельність населення США:

Рік	1900	1910	1920	1930	1940
Населення	75 994 575	95 972 266	105 710 620	122 775 046	135 669 275
Рік	1950	1960	1970		
Населення	150 697 361	179 323 175	203 235 298		

Виконайте інтерполяцію цих даних за допомогою формул Лагранжа і Ерміта. Використайте сплайн-інтерполяцію. Обчисліть чисельність населення в 1925 і 1969 роках. Визначте за допомогою декількох наближень методу найменших квадратів чисельність населення в 1980 році, що склала 226 547 082. Яке з наближень дає більш точний результат?

2. Побудуйте $n = 11$ точок даних, вибравши

$$\left.\begin{aligned} x_i &= \frac{i-1}{10}, \\ y_i &= \operatorname{erf}(x_i), \end{aligned}\right\} \quad i = 1, 2, ..., 11.$$

Значення функції $\operatorname{erf}(x)$ визначте на основі табличних даних або отримайте за допомогою пакета Mathematica на комп'ютері.

Виконайте інтерполяцію даних за допомогою формули Ньютона і кубічної сплайн-функції. Порівняйте результати в обох випадках зі значенням $\operatorname{erf}(x)$. Яка найбільша помилка?

3. У таблиці наведені так звані монотонні дані. Виберіть кращий, на ваш погляд, варіант інтерполяції. Чи є отримані інтерполянти монотонними?

x_i	0	2	3	5	6	8	9	11	12	14	15
y_i	10	10	10	10	10	10	10,5	15	50	60	85

4. Параметрична інтерполяція виникає під час розв'язання задач пошуку наближуючої функції по точках, що лежать на пласкій кривій. Її форму не можна описати за допомогою будь-якої функції від x, тому що функція була б неоднозначною. Введіть деякий параметр t, що змінюється вздовж кривої, яка описує, наприклад, букву S. Задайте вздовж кривої координати восьми точок, інтерполюйте ці дані й відтворіть отриманий результат. Виберіть як модельну замкнуту криву, наприклад букву O. Повторіть зазначену послідовність дій.

5. Відома залежність

$$\frac{i}{10} = \operatorname{erf}(x_i), \quad 0 < i < 10.$$

Знайдіть шістнадцять значень $\operatorname{erf}(x)$ із кроком $h = 0,1$ для $0 \leqslant x \leqslant 1,5$ у таблицях або порахуйте їх за допомогою пакета Mathematica на комп'ютері. Оцініть значення аргументів у проміжних точках $0,05 + 0,1j$, $j = 1, 2, ..., 15$. Використовуйте для побудови поліноми Стирлінга, Бесселя і сплайн-функції. Порівняйте результати з точними значеннями.

6. По трьох значеннях функції $y = \sin(x)$ на відрізку $[0, \pi/2]$ побудуйте поліном Лагранжа і оцініть залишковий член. Обчисліть наближене значення для точки $\pi/6$.

7. Складіть таблицю скінченних різниць для функції $y = x \ln^2 x$, обчисленої з трьома знаками після коми, в точках $x_i = 0,4 + 0,2i$, $i = 1, 2, ..., 10$. Знайдіть значення функції в точках 0,5; 5,1; 5,5; 2,3, вибираючи для цього відповідний інтерполяційний поліном. Порівняйте отримані результати.

Розділ 6

Чисельні методи розв'язання нелінійних рівнянь

- Метод дихотомії
- Умови збіжності методу простої ітерації
- Метод Ньютона і його модифікації
- Обчислення коренів алгебраїчних рівнянь
- Методи січних і хорд
- Метод Мюллера
- Розширення області розв'язання
- Розв'язання систем нелінійних рівнянь

Математичними моделями багатьох об'єктів і процесів навколишнього світу є нелінійні рівняння і системи нелінійних рівнянь: алгебраїчні і трансцендентні — для сталих станів, диференціальні — для динамічних процесів. У цьому розділі ми розглянемо методи розв'язання нелінійних алгебраїчних і трансцендентних рівнянь.

Розв'язання нелінійних рівнянь виду

$$f(x) = 0 \tag{6.1}$$

виконується переважно чисельними методами, основними властивостями яких є ітераційність рішення і локальність апроксимації [11, 19, 39].

На відміну від лінійних рівнянь не існує прямих методів розв'язання нелінійних рівнянь. Загалом процедура розв'язання нелінійних рівнянь складається з двох етапів: попереднє знаходження інтервалів, що містять лише один корінь (локалізація коренів) і подальше уточнення коренів (розв'язання рівняння).

Процедура розв'язання починається з вибору початкової точки x_0 і обчислення нев'язки рівняння $\varepsilon = f(x_0)$. Якщо $f(x_0) \neq 0$, за певним алгоритмом формується послідовність уточнень x_n з використанням інформації про знак нев'язки, про значення самої нев'язки ε або про швидкість її зміни $\partial\varepsilon/\partial x$.

Вибір початкового значення x_0 є важливим етапом, який істотно впливає на ефективність усієї процедури розв'язання і навіть на можливість одержання розв'язку.

Знайомство з існуючими методами почнемо з прикладу простого одновимірного нелінійного рівняння. Припустимо, що це рівняння має не менше одного дійсного кореня.

6.1. Метод дихотомії

Метод дихотомії (або метод поділу відрізка навпіл) передбачає послідовне обчислення значень функції в ряді точок. Перед використанням методу необхідно визначити відрізок, який містить лише один корінь рівняння. Для пошуку такого відрізка досить визначити точки a_0 і b_0 (рис. 6.1), в яких знаки функції будуть протилежними, тобто $f(a_0)f(b_0) < 0$, наприклад: $f(a_0) < 0$ і $f(b_0) > 0$. Знайти такий відрізок можна, скориставшись графічним або табличним методом.

Далі відрізок починають зменшувати, визначаючи на кожному кроці алгоритму координати його нових граничних точок a_n і b_n за значеннями a_{n-1}, b_{n-1} та координату середньої точки попереднього відрізку:

$$m_n = \frac{1}{2}(a_{n-1} + b_{n-1}).$$

У залежності від знаку функції в точці m_n новий відрізок $[a_n, b_n]$ функції встановлюється згідно з таким правилом:

$$[a_n, b_n] = \begin{cases} [m_n, b_{n-1}], & f(m_n) < 0, \\ [a_{n-1}, m_n], & f(m_n) > 0, \end{cases} \tag{6.2}$$

де $n = 1, 2, \ldots$, m_n – середня точка відрізку $[a_{n-1}, b_{n-1}]$.

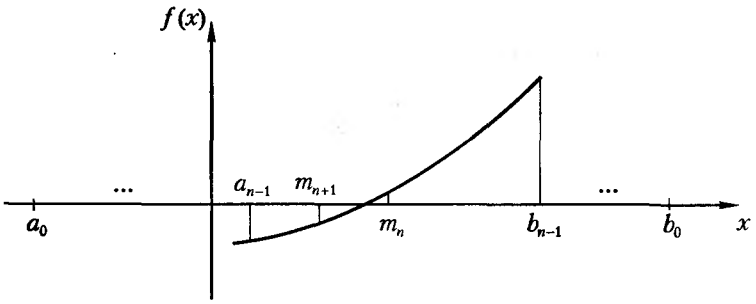


Рис. 6.1. Послідовне стискання інтервалу ізоляції кореня

Довжина інтервалу ізоляції кореня після виконання n кроків зменшується до

$$(b_n - a_n) = 2^{-n}(b_0 - a_0),$$

а значення кореня α, обумовлене координатою середньої точки, і його похибки задаються виразами

$$\alpha = m_{n+1} \pm \xi_n, \quad \xi_n = 2^{-n-1}(b_0 - a_0). \tag{6.3}$$

Із формули (6.3) випливає, що збіжність процесу обчислень дуже повільна, оскільки точність в одному десятковому розряді досягається за 3–4 кроки через те, що $10^{-1} \approx 2^{-3{,}3}$. Цей метод має абсолютну збіжність, тому ніяких вимог до виду і властивостей функції $f(x)$ не існує (вони висуваються в разі використання інших більш швидкодіючих методів).

Приклад 6.1

Обчислимо методом дихотомії корінь рівняння $y = \sin^2(x + \pi/3) - (x/2)^2$ із точністю до 0,01.

Зменшити кількість ітерацій методу дихотомії можна ще на етапі початкового вибору інтервалу ізоляції кореня, якщо використати графічний метод розв'язання рівнянь. Згідно з цим методом досить побудувати графік функції й оцінити наближене значення коренів рівняння за збудованим графіком. Для нашого прикладу графік функції показано на рис. 6.2.

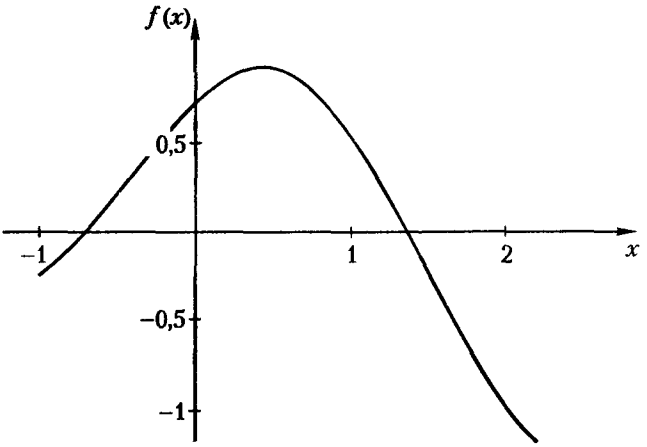


Рис. 6.2. Графік функції $y = \sin^2(x + \pi/3) - (x/2)^2$

Вибираємо початковий інтервал ізоляції першого кореня $I_{01} = [-1, 0]$, тому що $f(-1) = -0{,}247662 < 0$, $f(0) > 0{,}751012 > 0$. Послідовно стискуємо інтервал за допомогою операторів пакета Mathematica.

```
In[]:= f[x_]:= Sin[x + π/3]^2 - (x/2)^2;
       a = -1; b = 0;
       ϵ = 0.01; n = 0;
       While[ Abs[a - b] > ϵ, mid = (a + b)/2;
             Print["n = ", n, " a = ", N[a], " b = ", N[b]];
          If[ f[mid]*f[a] < 0, b = mid, a = mid ];
             Print[" mid = ", N[mid], " f[mid] = ", N[f[mid]];  n++ ];
```

Отримані результати послідовного зменшення інтервалу:

```
n = 0  a = -1.        b =  0        mid = -0.5       f[mid] =  0.208208
n = 1  a = -1         b = -0.5      mid = -0.75      f[mid] = -0.0548687
n = 2  a = -0.75      b = -0.5      mid = -0.625     f[mid] =  0.0702519
n = 3  a = -0.75      b = -0.625    mid = -0.6875    f[mid] =  0.00573371
n = 4  a = -0.75      b = -0.6875   mid = -0.71875   f[mid] = -0.0250964
n = 5  a = -0.71875   b = -0.6875   mid = -0.703125  f[mid] = -0.00980889
n = 6  a = -0.703125  b = -0.6875   mid = -0.695313  f[mid] = -0.00206886
```

Для визначення другого кореня вибираємо початковий інтервал ізоляції $I_{02} = [1, 2]$, виходячи з умов $f(1) = 0{,}5538747 > 0$, $f(2) = -0{,}991334 < 0$. Початковий інтервал змінюється в програмі:

```
In[]:= f[x_]:= Sin[x + π/3]^2 - (x/2)^2;
       a = 1; b = 2; ϵ = 0.01; n = 0;
       While[ Abs[a - b] > ϵ, mid = (a + b)/2; Print["n = ", n," a = ", N[a], " b = ", N[b]]
           If[ f[mid]*f[a] < 0, b = mid, a = mid];
       Print[" mid = ", N[mid], " f[mid] = ", N[f[mid]]];  n++ ];
```

Результати послідовного зменшення інтервалу:

```
n = 0  a = 1.       b = 2        mid = 1.5      f[mid] = -0.248891
n = 1  a = 1.       b = 1.5      mid = 1.25     f[mid] =  0.168235
n = 2  a = 1.25     b = 1.5      mid = 1.375    f[mid] = -0.0384678
n = 3  a = 1.25     b = 1.375    mid = 1.3125   f[mid] =  0.0658329
n = 4  a = 1.3125   b = 1.375    mid = 1.34375  f[mid] =  0.0138589
n = 5  a = 1.34375  b = 1.375    mid = 1.35938  f[mid] = -0.012268
n = 6  a = 1.34375  b = 1.35938  mid = 1.35156  f[mid] =  0.000805537
n = 7  a = 1.35156  b = 1.35938  mid = 1.35547  f[mid] = -0.00572882
```

Слід мати на увазі, що в разі вибору великого інтервалу ізоляції кореня обсяг обчислень для наступного уточнення коренів збільшується; якщо ж відрізок інтервалу занадто малий, значення кореня може опинитися за його межами, оскільки графік будується приблизно.

6.2. Метод простої ітерації

Метод простої ітерації полягає в тому, що рівняння (6.1) попередньо приводиться до канонічного вигляду:

$$x = \varphi(x), \tag{6.4}$$

і ітерації виконуються за правилом

$$x_{n+1} = \varphi(x_n), \quad n = 0, 1, \ldots, \tag{6.5}$$

причому задається початкове наближення x_0.

Якщо процес обчислень збігається до розв'язку α рівняння (6.1), тобто $\alpha = \varphi(\alpha)$, то в разі припущення, що функція $\varphi(x)$ визначена і неперервна на відрізку, де треба знайти корінь, можна встановити взаємозв'язок між похибками обчислень на двох сусідніх ітераціях:

$$\varepsilon_{n+1} = x_{n+1} - \alpha = \varphi(x_n) - \varphi(\alpha) \approx \varphi'(x^*)(x_n - \alpha) = \varphi'(x^*)\varepsilon_n, \quad x^* \in [x_n, \alpha]. \tag{6.6}$$

Із виразу (6.6) випливає, що достатньою умовою збіжності методу простої ітерації, за якої $|\varepsilon_{n+1}|$ буде менше $|\varepsilon_n|$, є умова

$$|\varphi'(x^*)| < 1. \tag{6.7}$$

Умова збіжності (6.7) ітерацій під час розв'язування нелінійних рівнянь є, власне кажучи, узагальненням достатньої умови збіжності (3.6) ітераційного

процесу під час розв'язання систем лінійних рівнянь, отриманої в розділі 3 у вигляді $M < 1$, якщо покласти $|\varphi'(x^*)| = M$. Чим менше значення $|\varphi'(x^*)|$, тим швидше збігається ітераційний процес.

У випадку, коли $\varphi'(x^*) > 0$, похибки ε_{n+1} і ε_n будуть мати однакові знаки і збіжність x_{n+1} до α буде монотонною (рис. 6.3, *а*). Якщо ж $\varphi'(x^*) < 0$, похибки ε_{n+1} і ε_n матимуть різні знаки і наближення x_{n+1} буде збігатися до α, коливаючись біля α (рис. 6.3, *б*).

Коли $\varphi'(x^*) > 1$, похибка ε_{n+1} за абсолютним значенням більше ε_n, і наближення x_{n+1} буде відстояти від розв'язку α далі, ніж x_n. Розв'язок α мов «відштовхує» наближення x_n, близькі до нього, тому не буде збіжності послідовності x_n до α (рис. 6.3, *в*, *г*).

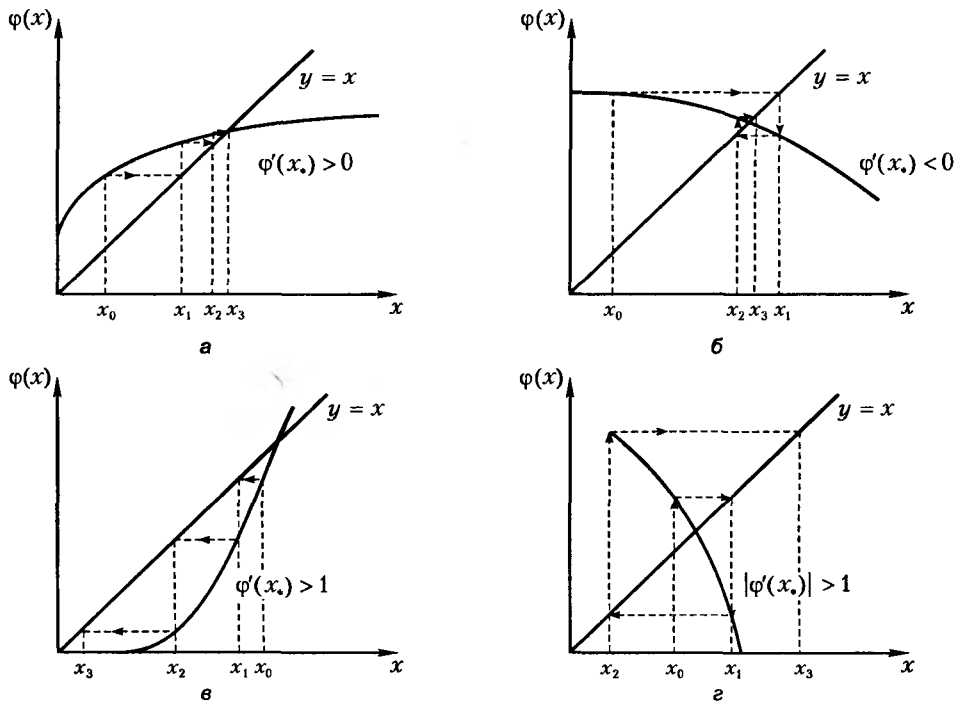


Рис. 6.3. Можливі варіанти збіжності ітерацій: *а* — монотонна збіжність; б — коливальна збіжність; *в* — монотонна розбіжність; *г* — коливальна розбіжність

Особливим випадком є умова, коли $\varphi'(x^*) = 0$, і збіжність ітерацій замість *лінійної*, обумовленої виразом (6.6), стає нелінійною, зокрема квадратичною, як це буде показано в наступному розділі.

Оцінювання глобальної похибки $|x_{n+1} - \alpha|$ зручно виконувати на основі значень локальної похибки, тобто за значеннями наближень, отриманих на сусідніх ітераціях за аналогією з формулою (3.23). Для цього додамо до правої частини виразу (6.6) і віднімемо від неї x_{n+1}:

$$\varepsilon_{n+1} = x_{n+1} - \alpha \approx \varphi'(x^*)[(x_n - x_{n+1}) + (x_{n+1} - \alpha)].$$

Після зведення подібних членів одержимо:

$$|x_{n+1} - \alpha| \approx \frac{|\varphi'(x^*)|}{1 - |\varphi'(x^*)|} |x_n - x_{n+1}| \approx \frac{M}{1 - M} |x_n - x_{n+1}|,$$

де $|\varphi'(x^*)| = M$.

Якщо обчислювати похибку від початкового значення x_0, то для поточної похибки на n-й ітерації згідно з виразом (6.6) справедливе співвідношення

$$|x_{n+1} - \alpha| \leqslant \frac{M^{n+1}}{1 - M} |x_1 - x_0|. \tag{6.8}$$

Приклад 6.2

Використовуючи пакет Mathematica, знайдемо корінь рівняння

$$f(x) = x^{2,5} - e^{-x^{1,2}}.$$

Функція $f(x)$ задається у вигляді

```
In[]:= f[x_]:= x^(2.5) - Exp[-x^(1.2)]
```

Для локалізації дійсних коренів побудуємо графік у тій області, в якій, за нашими оцінками, мають знаходитися потрібні нам корені. Для нашого прикладу $x > 0$. Якщо $x = 0$, то $f(0) = -1$. Зі збільшенням x значення функції зростає і, отже, знайдеться значення x^*, за якого $f(x^*) = 0$ (рис. 6.4). Будуємо графік функції $f(x)$:

```
In:= Plot[f[t], {t,0,2}, AxesLabel → {"x", "f(x)"}]
```

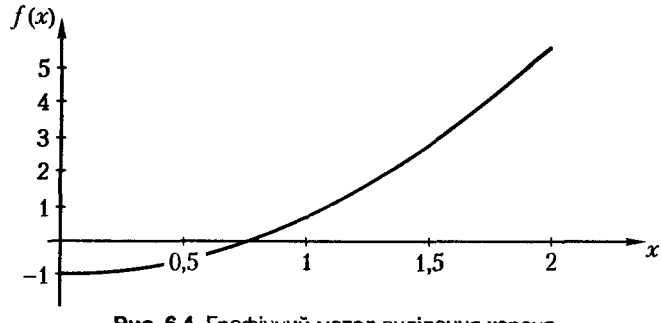


Рис. 6.4. Графічний метод виділення кореня

Початкове рівняння можна замінити еквівалентним йому рівнянням, зведеним до вигляду (6.4):

$$x = e^{-0,4x^{1,2}}.$$

Перевіримо для нього виконання умови (6.7). Визначимо в Mathematica функцію φ[x] = = Exp[-0.4x^1.2] і знайдемо значення її похідної (рис. 6.5) в околі кореня:

```
In[]:= φ[x_]:= Exp[-0.4x^1.2]; φ1[x_] = D[φ[x], x]
       Plot[φ1[x], {x,0,1}, AxesLabel → {"x", "φ'[x]"}]
```

На рисунку видно, що $|\varphi'(x)| < 1$, $x \in [0, 1]$ і, отже, ітераційний процес для всіх $x_0 \in [0, 1]$ збігається.

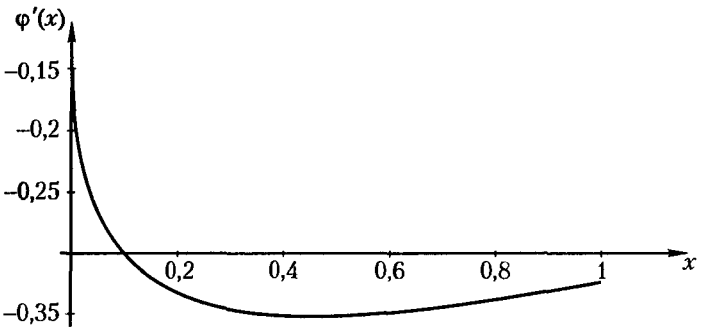


Рис. 6.5. Графік похідної функції $\varphi(x) = e^{-0,4x^{1,2}}$

Оцінимо отриману похибку, використовуючи співвідношення (6.8) і враховуючи в ньому $q = M = \max|\varphi'(x)|$, $x \in [0, 1]$ і $x_1 = \varphi(x_0)$:

$$\left|x_n - x^*\right| \leqslant \frac{q^n}{1-q}|\varphi(x_0) - x_0|, \quad n = 1, 2, \ldots. \tag{6.9}$$

Якщо похибка обчислення кореня рівняння не повинна перевищувати наперед заданого значення ε, то згідно з формулою (6.9) можна знайти необхідну кількість ітерацій n.

$$n^* \geqslant \frac{\lg[\varepsilon(1-q)/|\varphi(x_0) - x_0|]}{\lg q}.$$

Знайдемо значення q для розглянутого прикладу

```
In[]:= Q = -FindMinimum[φ1[x], {x, 0.5}]; q = Q[[1]]
Out[]= 0.351148
```

Тепер можна знайти необхідну кількість ітерацій n_ε, щоб забезпечити обчислення кореня із заданою точністю ε.

```
In[]:= x0 = 1.; ϵ = 0.0001;
       nϵ = Log[ϵ(1 - q)/Abs[φ[x0] - x0]]/Log[q]
Out[]= 8.15372
```

Округлимо отриманий результат до найближчого більшого цілого. Таким чином, для обчислення кореня із заданою точністю необхідно виконати дев'ять ітерацій. У разі послідовного уточнення кореня рівняння одна й та сама формула застосовується кілька разів. У пакеті Mathematica це можна зробити, використовуючи оператор `NestList[f, x, n]`, що дає список n значень аргументу x і функції f, починаючи зі значення x_0. Наприклад:

```
In[]:= n = IntegerPart[nϵ] + 1; L = NestList[φ, x0, n]
Out[]= {1.,        0.67032,  0.78074,  0.742886,  0.75578,
        0.751377,  0.752879, 0.752367 0.752542,  0.752482}
```

Послідовні наближення збігаються, коливаючись біля значення кореня. Це можна використовувати для уточнення похибки обчислення кореня, тому що похибка не буде перевищувати модуль різниці двох послідовних наближених значень. Знайдемо цю похибку:

```
In:= Ep = L[[n + 1]] - L[[n]]
Out= -0.0000596766
```

Збіжність процесу обчислень повністю визначається вибором функції $\varphi(x)$. Відмітимо, що не завжди вдається розв'язати відносно x рівняння (6.1) явно, так щоб привести його до вигляду (6.4) і задовольнити умову збіжності (6.7).

Приклад 6.3

Знайдемо розв'язок рівняння

$$f(x) = x\sin(x^2) - e^{-x/2}.$$

Для локалізації кореня побудуємо графік відповідної функції (рис. 6.6):

```
In[]:= f[x_]:= x*Sin[x^2] - E^(-x/2);
       Plot[f[x], {x, 0.5, 1}, AxesLabel → {"x", "f(x)"}]
```

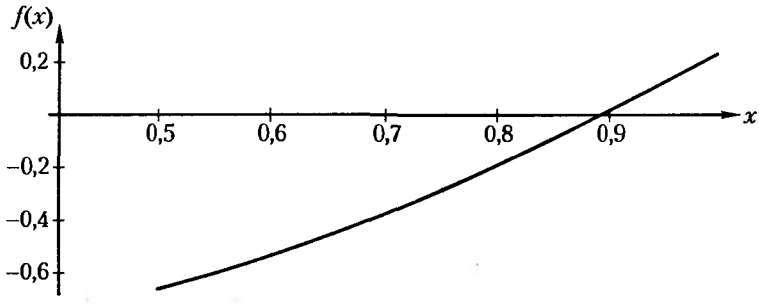


Рис. 6.6. Графік функції $f(x) = x\sin(x^2) - e^{-x/2}$

Розв'яжемо рівняння $x\sin(x^2) - e^{-x/2} = 0$ стосовно x, отримаємо

$$x = \frac{e^{(-x/2)}}{\sin(x^2)}.$$

Перевіримо виконання умови (6.4). Визначимо в пакеті Mathematica функцію φ[x] = = E^(-x/2)/Sin[x^2] і знайдемо значення її похідної (рис. 6.7) в околі кореня, який приблизно дорівнює 0,9:

```
In[]:=φ[x_]:= E^(-x/2)/Sin[x^2]; φ1[x_] = Dt[φ[x], x];
       Plot[φ1[x], {x, 0.8, 1}, AxesLabel → {"x", "φ'(x)"}]
```

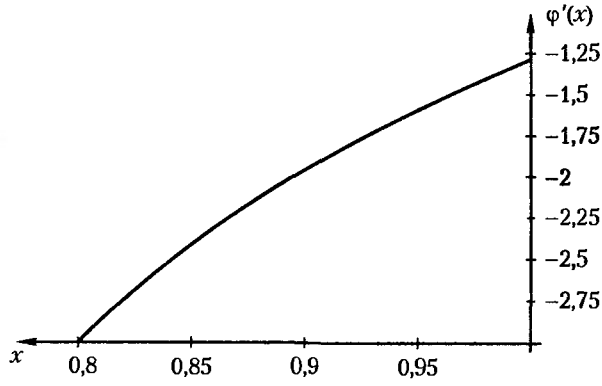


Рис. 6.7. Графік похідної функції $e^{(-x/2)}/\sin(x^2)$

Із наведеного графіка випливає, що в околі кореня значення $|\varphi'(x)| > 1$, тобто умова збіжності тут не справджується. Дійсно, якщо для розглянутого рівняння виконати ітерації за формулою (6.5)

$$x_{n+1} = \varphi(x_n), \quad n = 0, 1, \ldots,$$

то одержимо розбіжний процес

```
In[]:= x0 = 0.85; l1 = NestList[φ, x0, 9]
Out[]= {0.85,     0.98867,    0.735723,  1.34347,     0.525148,
        2.82437,  0.245467,   14.6883,   0.000757322, 1.74291×10^6}
```

У тих випадках, коли не вдається явно розв'язати вихідне рівняння $f(x) = 0$ відносно невідомої x, так щоб у вибраному рівнянні (6.4) функція $\varphi(x)$ задовольняла умову збіжності (6.7), ітерації виконують за іншим правилом:

$$x_{n+1} = x_n + \tau(x_n) f(x_n). \tag{6.10}$$

Тут допоміжна функція $\tau(x_n)$ не повинна змінювати свій знак на відрізку, де шукається корінь. Зокрема, якщо $\tau(x_n) = \tau = const$, одержимо *метод релаксації*:

$$(x_{n+1} - x_n)/\tau = f(x_n), \quad n = 0, 1, 2, \ldots,$$

для якого $\varphi'(x) = 1 + \tau f'(x)$ і метод сходиться за умови

$$-2 < \tau f'(x^*) < 0. \tag{6.11}$$

Якщо в деякому околі кореня виконуються умови

$$f'(x) < 0, \quad 0 < m_1 < |f'(x)| < M_1,$$

то метод релаксації збігається в разі $\tau \in (0, 2/M_1)$. Оптимальне значення параметра в такому випадку вибирається із відношення

$$\tau = 2/(M_1 + m_1). \tag{6.12}$$

Оберемо функцію, що буде задовольняти умову збіжності, у вигляді $\varphi(x) = x - \tau f(x)$. Визначимо початкову функцію та її похідну (рис. 6.8)

```
In[]:= f[x_]:= x*Sin[x^2] - E^(-x/2); fl[x_] = Dt[f[x], x];
       Plot[fl[x], {x, 0.8, 1}, AxesLabel → {"x", "f'(x)"};
```

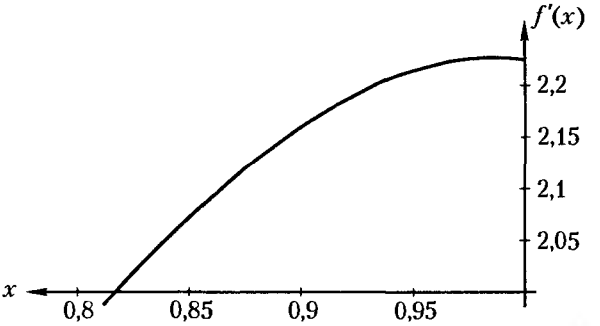


Рис. 6.8. Графік похідної функції $f(x) = x\sin(x^2) - e^{-x/2}$

Виберемо значення τ

```
In[]:= τ = 2/(fl[0.85] + fl[0.95])
Out[]= 0.466591
```

Побудуємо функцію, що буде задовольняти умову збіжності

```
In[]:= φ[x_]:= x - τ*f[x]; x0 = 0.9; Ll = NestList[φ, x0, 2]
Out[]= {0.9, 0.89336, 0.893397}
```

Знадобилося лише два кроки, щоб знайти розв'язок з точністю 10^{-6}. Для порівняння наведемо відповідний розв'язок, отриманий стандартним оператором Mathematica

```
In[]:= FindRoot[f[x] == 0, {x, x0}]
Out[]= {x0 → 0.893397}
```

У порівнянні з методом дихотомії метод простої ітерації є більш швидкодіючим, але він не є абсолютно збіжним, як інші методи, розглянуті нижче. Тому на практиці метод простої ітерації застосовується рідше.

6.3. Метод Ньютона

Для прискорення збіжності ітераційного процесу (рис. 6.9) чергове наближення x_{n+1} може бути визначене за формулою, де враховано як значення самої функції $f(x_n)$, так і швидкість її зміни $f'(x_n)$:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}. \tag{6.13}$$

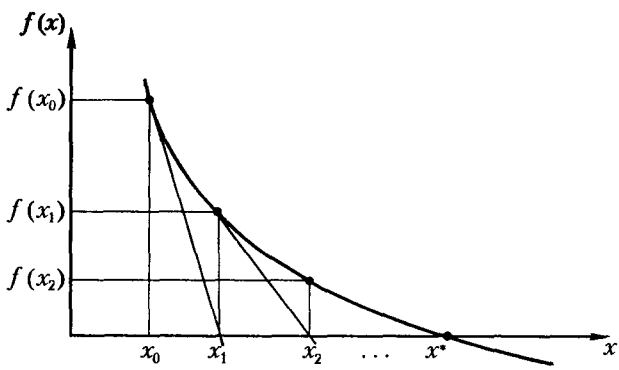


Рис. 6.9. Ітераційний процес методу Ньютона

Цю формулу можна отримати безпосередньо з ряду Тейлора, в якому зберігаються два перших члени розкладу функції:

$$f(x_{n+1}) = f(x_n) + (x_{n+1} - x_n)f'(x_n) = 0,$$

звідки

$$x_{n+1} = x_n + \Delta x_n,$$

де

$$\Delta x_n = -f(x_n)/f'(x_n).$$

Ефективність процедури (6.13) залежить від початкового наближення x_0. Наприклад, воно може збігатися з одним із кінців інтервалу ізоляції кореня $[a, b]$, для якого виконується умова $f(x)^* f''(x) > 0$.

Приклад 6.4

Знайдемо методом Ньютона розв'язок рівняння

$$f(x) = \sin^2(x + \pi/3) - (x/2)^2,$$

розглянутого в прикладі 6.1. Для цього використаємо програму:

```
In[]:= f[x_]:= Sin[x + π/3]^2 - (x/2)^2;
       fl[x_] = Dt[f[x], x];
       x0 = -1; n = 0; ϵ = 0.01; eps = 10; x = x0;
       While[eps > ϵ, x = x - N[f[x]]/N[fl[x]];
             eps = Abs[x0 - x]; x0 = x; n++;
             Print["n = ", n, " x = ", x, " f[x] = ", f[x],
                   " fl[x] = ", fl[x], " Δx = ", eps] ];
```

Вона дозволяє не тільки обчислити потрібну похідну $f'(x) = 2\sin(x + \pi/3)\cos(x + \pi/3) - -(x/2)$, але й отримати такі результати, якщо $x_0 = -1$:

```
n = 0  x = -1          f[x] = -0.247774      fl[x] = 0.594255   Δx = 1
n = 1  x = -0.583051 f[x] =  0.115412      fl[x] = 1.09212    Δx = 0.416949
n = 2  x = -0.688728 f[x] =  0.00450335    fl[x] = 1.00144    Δx = 0.105677
n = 3  x = -0.693225 f[x] =  0.0000102278 fl[x] = 0.996887   Δx = 0.00449685
```

і якщо $x_0 = 2$, де останній стовпець містить значення локальної похибки обчислень:

```
n = 0  x = 2          f[x] = -0.991116      fl[x] = -1.18767   Δx = 1
n = 1  x = 1.1655     f[x] =  0.301944      fl[x] = -1.54184   Δx = 0.834504
n = 2  x = 1.36133 f[x] = -0.0155404     fl[x] = -1.67519   Δx = 0.195833
n = 3  x = 1.35205 f[x] = -0.0000130534 fl[x] = -1.67232   Δx = 0.00927677
```

На прикладі обчислення другого кореня показано, як у загальному випадку здійснюється визначення всіх коренів рівняння «скануванням» області розв'язку за допомогою вибору початкових значень у різних точках цієї області.

У припущенні, що $f'(x) \neq 0$ й $f''(x) \neq 0$, оцінимо збіжність обчислень за формулою (6.13). Для цього в околі кореня рівняння α розкладемо функцію в ряд Тейлора з урахуванням третього члена, що визначає в основному нелінійність апроксимації:

$$f(\alpha) = f(x_n) + (\alpha - x_n)f'(x_n) + \frac{1}{2}(\alpha - x_n)^2 f''(x^*) = 0, \quad x^* \in [x_n, \alpha]. \quad (6.14)$$

Поділивши співвідношення (6.14) на $f'(x)$, з урахуванням формули (6.13) отримаємо

$$\frac{f(x_n)}{f'(x_n)} + \alpha - x_n = \alpha - x_{n+1} = -\frac{1}{2}\frac{(\alpha - x_n)^2 f''(x^*)}{f'(x_n)}.$$

Оскільки $\varepsilon_n = \alpha - x_n$ і $\varepsilon_{n+1} = \alpha - x_{n+1}$, то, якщо $x_n \to \alpha$ одержуємо *квадратичну* залежність похибки на наступних ітераціях:

$$\frac{\varepsilon_{n+1}}{\varepsilon_n^2} \to c = \frac{1}{2}\frac{f''(\alpha)}{f'(\alpha)}. \tag{6.15}$$

Таким чином, метод Ньютона має квадратичну збіжність:

$$|\varepsilon_{n+1}| \leqslant m\varepsilon_n^2. \tag{6.16}$$

Якщо з урахуванням вигляду функції (тобто значень $f''(\alpha)$, $f'(\alpha)$) вибрати початкове значення x_0, так щоб на інтервалі $[\alpha - \varepsilon_0, \alpha + \varepsilon_0]$ виконувалася умова

$$m\varepsilon_0 = m(x_0 - \alpha) < 1,$$

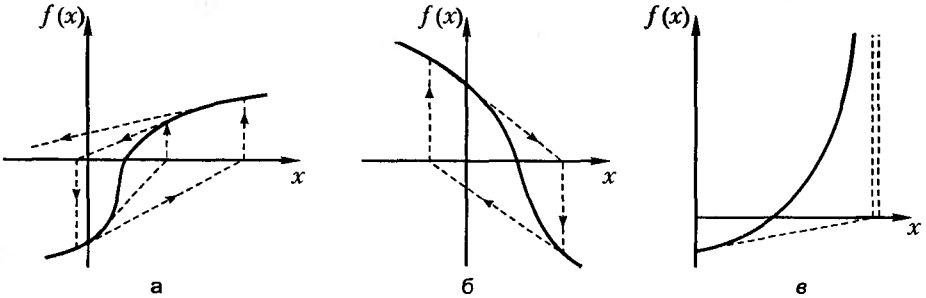
то метод Ньютона завжди буде збігатися квадратично. Про швидкість збіжності можна судити, виходячи зі зміни похибки, обчисленої для $m\varepsilon_0 = 0,9 < 1$:

```
In[]:= Table[0.9^(2^n), {n,7}]
Out[]= {0.81, 0.6561, 0.430467, 0.185302, 0.0343368, 0.00117902, 1.39008×10^-6}
```

На практиці ж вибір значення x_0 набагато складніший. Умови збіжності методу Ньютона, досліджені різними авторами, можна визначити у такий спосіб:

- функція $f(x)$ має бути принаймні двічі диференційована на інтервалі $[a, b]$;
- перша похідна функції не повинна перетворюватися на нуль $f'(x) \neq 0$;
- знаки функції на кінцях інтервалу мають бути різними $\operatorname{sign} f(a) \neq \operatorname{sign} f(b)$;
- друга похідна на інтервалі $[a, b]$ повинна зберегти свій знак: $f''(x) \gg 0$ (опукла) чи $f''(x) < 0$ (увігнута);
- зростання функції (її крутість) має бути обмежене значенням $|f(x)/f'(x)| \leqslant (b - a)$, де $x = a$, коли $|f'(a)| < |f'(b)|$, і $x = b$, коли $|f'(a)| > |f'(b)|$, що ускладнює роботу з експонентними функціями.

Таким чином, збіжність обчислень методу гарантується лише для монотонних і обмежених за крутістю нелінійних функцій. На рис. 6.10 наведені приклади ітераційних процесів, коли ці умови не виконуються, що не дозволяє обрати значення x_0 для побудови збіжного ітераційного процесу.



а б в

Рис. 6.10. Приклади розбіжності методу Ньютона: *а* — функція з перегином; *б* — «зациклення» обчислень; *в* — експонентна функція

Збіжність процедури обчислення кореня рівняння за методом Ньютона можна підвищити, якщо замість виразу (6.13) виконувати ітерації методу за формулою

$$x_{n+1} = x_n \alpha_n \Delta x_n = x_n - \alpha_n \frac{f(x_n)}{f'(x_n)} \tag{6.17}$$

і на кожному кроці обирати коефіцієнт демпфірування α_n з умови зменшення норми функції:

$$\|f(x_{n+1})\| < \|f(x_n)\|. \tag{6.18}$$

У формулі (6.17) співвідношення норм $\|f(x_{n+1})\|$ і $\|f(x_n)\|$ залежить від значення коефіцієнта демпфірування α_n, і таку залежність можна вважати квадратичною (рис. 6.11):

$$K(\alpha_n) = \frac{\|f(x_{n+1})\|}{\|f(x_n)\|} = c_2 \alpha_n^2 + c_1 \alpha_n + c_0. \tag{6.19}$$

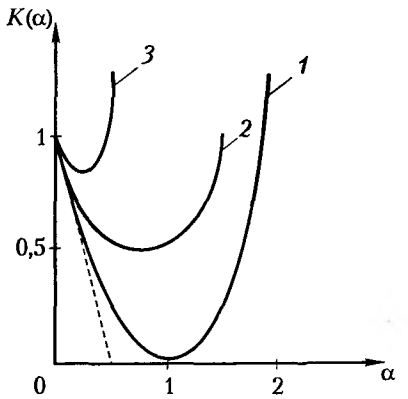


Рис. 6.11. Криві залежності $K(\alpha_n)$: *1* — розв'язання лінійної задачі за одну ітерацію $\alpha_{min} = 1$, $K(1) = 0$; *2* — $0{,}1 \leqslant K(1) \leqslant 100$; *3* — $K(1) > 100$

У цьому можна пересвідчитися, проаналізувавши експериментальні криві Бреніна, наведені на рис. 6.11, — незалежно від значення коефіцієнта відношення норм відхилу $K(\alpha_n)$ всі криві мають спільну дотичну в точці $[0, 1]$ із нахилом, рівним -2, і частки з оптимальними значеннями $\alpha_{n\,\min}$.

Спочатку на кожній ітерації оцінюється коефіцієнт $K(1)$ за повного кроку збільшення Δx_n, тобто $\alpha_n = 1$, і через точки $[0, 1]$ і $[1, K(1)]$ проводиться парабола (6.19), коефіцієнти якої відповідно дорівнюють:

$c_0 = 1$, з умови $K(0) = 1$;

$c_1 = -2$, з умови $\left|\frac{\partial K(\alpha_n)}{\partial \alpha_n}\right|_{\alpha_n = 0} = -2$;

$c_2 = 1 + M(1)$, з умови $|K(\alpha_n)|_{\alpha_n = 1} = K(1)$.

Для параболи (6.19) з урахуванням наведених значень коефіцієнтів обчислюються координати мінімальної точки $\alpha_{\min}$ і $K(\alpha_{n\min})$:

$$\frac{\partial K(\alpha_n)}{\partial \alpha_n} = 2[K(1)+1]\alpha_n - 2 = 0,$$

при цьому

$$\alpha_{n\min} = \frac{1}{K(1)+1} \quad \text{і} \quad K(\alpha_{n\min}) = \frac{K(1)}{K(1)+1} < 1.$$

Крива *2* залежності коефіцієнта відношення норм відхилу $K(\alpha_n)$ (рис. 6.11) відповідає вибору $0 < \alpha_{n\min} < 1$. Якщо ж одержуємо оцінку $K(1) > 100$ (крива *3*), то параболічна апроксимація не виконується і коефіцієнт послідовно зменшується $\alpha_n = 1/2, 1/4, 1/8, \ldots$, поки не виконається умова $K(\alpha_n) < 1$. Якщо $K(1) < 0{,}1$, умова збіжності виконується з запасом, і обчислення проводяться з повним кроком збільшення аргументу

$$\alpha_n = 1.$$

Процедуру послідовного підбору коефіцієнта демпфірування α_n без параболічної апроксимації для розглянутого вище випадку $K(1) > 100$ (рис. 6.11) можна поширити і на всі інші випадки, коли в процесі обчислень не виконується умова (6.18), тобто $\|f(x_{n+1})\| > \|f(x_n)\|$. У цьому разі обирається деякий постійний коефіцієнт α_0, наприклад $\alpha_0 = 0{,}75$, що послідовно використовується на всіх ітераціях, для яких $\|f(x_{n+1})\| > \|f(x_n)\|$, при цьому поточне значення коефіцієнта дорівнює

$$\alpha_n = \alpha_0^k, \tag{6.20}$$

де k — кількість попередніх кроків, для яких не виконувалася умова (6.18).

Починаючи з ітерації, для якої ця умова вже виконується, наявне значення коефіцієнта демпфірування α_n збільшується в $1/\alpha_0$ разів на кожній ітерації аж до відновлення обчислень із повним кроком Δx_n.

Приклад 6.5

Розв'яжемо методом Ньютона рівняння $f(x) = 10\,\mathrm{arctg}(5-x) - 1 = 0$. У разі використання стандартної процедури методу (6.13) навіть за ненульового початкового значення ітерації не збігаються:

```
In[]:= f[x_]:= 10*ArcTan[5 - x] - 1;
       fl[x_] = Dt[f[x], x];
       x0 = 3.5; n = 0; ϵ = 0.01; eps = 10; x = x0;
       While[eps > ϵ,
            x = N[x] - N[f[x]]/N[fl[x]]; eps = Abs[x0 - x]; x0 = x; n++ ];
Out[]= $RecursionLimit::reclim: Recursion depth of 256 exceeded. General::stop
```

Якщо перейти до процедури (6.17) із вибором коефіцієнта $\alpha_n = 0{,}75$ на кожному кроці обчислення за формулою (6.20), то ітерації збігаються досить швидко:

```
In[]:= f[x_]: = 10*ArcTan[5 - x] - 1; fl[x_] = Dt[f[x],x];
       x0 = 3.5; n = 0; ε = 0.0001; eps =10; x = x0; a0 = 0.5; a = 1; k = 0;
```

```
While[eps > ε,
x = N[x] - a0*a*N[f[x]]/N[fl[x]];
eps = Abs[x0 - x];
If [Norm[N[f[x]]] < Norm[N[f[x0]]], k++; a = 0.75^k, k = 0;
If [a = 1, a = 0.75, a/0.75]];
x0 = x; n++;
Print["n = ", n, " x = ", N[x],
      " f[x] = ", N[f[x]], " a = ", a]; a0 = a];
```

```
n = 1  x = 4.93454  f[x] = -0.346331   a = 0.75
n = 2  x = 4.91498  f[x] = -0.151792   a = 0.5625
n = 3  x = 4.91014  f[x] = -0.103784   a = 0.421875
n = 4  x = 4.90828  f[x] = -0.085316   a = 0.316406
n = 5  x = 4.90741  f[x] = -0.0767754  a = 0.237305
n = 6  x = 4.90698  f[x] = -0.0724521  a = 0.177979
n = 7  x = 4.90675  f[x] = -0.0701571  a = 0.133484
n = 8  x = 4.90662  f[x] = -0.0689071  a = 0.100113
n = 9  x = 4.90655  f[x] = -0.0682165  a = 0.0750847
```

Саме такий метод Ньютона з демпфіруванням реалізовано в операторі NSolve пакета Mathematica, що дозволяє застосовувати цей пакет безпосередньо для розв'язання вихідного рівняння:

```
In[]:= NSolve[10.*ArcTan[5. - x] - 1., x]
Out[]= {{x → 4.89967}}
```

6.3.1. Метод Ньютона для кратних коренів

Швидкість збіжності методу Ньютона падає, якщо рівняння $f(x)$ має кратні корені. Метод Ньютона в цьому випадку втрачає квадратичну збіжність, і його збіжність стає лінійною. Швидкість збіжності залежить від кратності кореня q:

$$\frac{\varepsilon_{n+1}}{\varepsilon_n} = \frac{q-1}{q}. \tag{6.21}$$

Наприклад, для $q = 2$ збіжність методу Ньютона буде такою ж, як і збіжність методу дихотомії, тобто

$$\frac{\varepsilon_{n+1}}{\varepsilon_n} = 1/2. \tag{6.22}$$

Однак, якщо кратність кореня q відома заздалегідь, можна зберегти квадратичну збіжність методу Ньютона, модифікувавши основну формулу ітерацій:

$$x_{n+1} = x_n + q\Delta x_n. \tag{6.23}$$

У більшості випадків кратність коренів невідома, тому для збереження квадратичної збіжності на базі заданого рівняння з кратним коренем α формують інше рівняння

$$y(x) = f(x)/f'(x), \tag{6.24}$$

яке має одиничний корінь α незалежно від значення його кратності q у вихідному рівнянні $f(x)=0$. Це можна легко довести, якщо врахувати, що кратний корінь задовольняє як саме рівняння, так і похідні від нього, тобто $f^{(j)}(\alpha)=0$, якщо $j<q$. Записавши для розглянутого випадку розклад у ряд Тейлора функції та її похідної:

$$f(x)=\frac{1}{q!}(x-\alpha)^q f^{(q)}(\xi), \quad f'(x)=\frac{1}{(q-1)!}(x-\alpha)^{q-1} f^{(q)}(\xi_1),$$
$$\xi, \xi_1 \in \text{int}[x, \alpha],$$

переконуємося, що, коли $x \to \infty$, $\lim[y(x)/(x-\alpha)]=1/q$.

Тоді замість виразу (6.13) формула методу Ньютона для кратних коренів набуває такого вигляду:

$$x_{n+1}=x_n-\frac{y(x_n)}{y'(x_n)}; \quad y'(x_n)=1-\frac{f''(x_n)}{f'(x_n)}y(x_n)$$

чи остаточно:

$$x_{n+1}=x_n-\frac{f(x_n)f'(x_n)}{[f'(x_n)]^2-f(x_n)f''(x_n)}. \tag{6.25}$$

Приклад 6.6

Знайдемо методом Ньютона кратні корені рівняння

$$f(x)=x^3-5x^2+7x-3=(x-3)(x-1)(x-1).$$

Спочатку застосуємо стандартну процедуру (6.13):

```
In[]:= f[x_]:= x^3 - 5*x^2 + 7*x - 3;
       f1[x_] = Dt[f[x], x]
       x0 = 0.5; n = 0; ϵ = 0.001; eps=1; x = x0;
       While[eps > ϵ, x = x - f[x]/f1[x]; eps = Abs[x0 - x]; x0 = x; n++;
       Print["n = ", n, " x = ", x, " f[x] = ", f[x], " Δx = ", eps] ];
```

У разі вибору $x_0=0,5$ корінь $x_1=1$ локалізується з заданою точністю після шостої ітерації:

```
n = 0  x = 0.5       f[x] = -0.625        Δx = 1
n = 1  x = 0.727273  f[x] = -0.169046     Δx = 0.227273
n = 2  x = 0.855918  f[x] = -0.0445105    Δx = 0.128645
n = 3  x = 0.925617  f[x] = -0.0114772    Δx = 0.0696993
n = 4  x = 0.962153  f[x] = -0.00291894   Δx = 0.0365365
n = 5  x = 0.980903  f[x] = -0.000736386  Δx = 0.0187492
n = 6  x = 0.990406  f[x] = -0.000184959  Δx = 0.00950375
```

Із застосуванням модифікованої процедури (6.25) за того ж початкового значення $x_0=0,5$:

```
In[]:= f[x_]:= x^3 - 5*x^2 + 7*x - 3;
       f1[x_] = Dt[f[x], x];
       f2[x_] = Dt[f[x], {x,2}];
       x0 = 0.5; n = 0; ϵ = 0.001; eps = 1; x = x0;
   While[eps > ϵ,
         x = x - f[x]*f1[x]/((f1[x])^2 - f[x]*f2[x]); eps = Abs[x0 - x]; x0 = x; n++;
   Print["n = ", n, " x = ", x, " f[x] = ", f[x], " Δx = ", eps] ];
```

невідомий корінь $x_1 = 1$ локалізується після другої ітерації

```
n = 0  x = 0.5       f[x] = -0.625           Δx = 1
n = 1  x = 1.03922   f[x] = -0.00301543      Δx = 0.539216
n = 2  x = 1.0004    f[x] = -3.19808*10-7    Δx = 0.0388158
n = 3  x = 1.        f[x] = -2.66454*10-15   Δx = 0.00039988
```

6.3.2. Обчислення коренів поліномів методом Ньютона

Корені поліномів типу

$$p(z) = a_0 z^m + a_1 z^{m-1} + \ldots + a_m = 0 \tag{6.26}$$

можна знайти за допомогою методу Ньютона

$$z_{n+1} = z_n - \frac{p(z_n)}{p'(z_n)}, \tag{6.27}$$

при цьому для обчислення значень першої й другої похідних від поліноміальних функцій зручно використовувати рекурсивну процедуру Горнера для обчислення поліноміальних коефіцієнтів (див. розділ 1):

$$\begin{aligned} &p(z_n) = b_m; \quad p'(z_n) = c_{m-1}; \quad p''(z_n) = 2!\,d_{m-2};\\ &b_0 = a_0, \quad b_i = b_{i-1} z_k + a_i, \quad i = 1, 2, \ldots, m;\\ &c_0 = b_0, \quad c_i = c_{i-1} z_n + b_i, \quad i = 1, \ldots, m-1;\\ &d_0 = c_0, \quad d_i = d_{i-1} z_n - c_i, \quad i = 1, \ldots, m-2. \end{aligned} \tag{6.28}$$

Знайдений корінь виключається з полінома, і порядок останнього зменшується:

$$g(z) = \frac{p(z) - p(z_n)}{z - z_n} = b_0 z^{m-1} + \ldots + b_{m-2} z + b_{m-1}. \tag{6.29}$$

Описана процедура повторюється n разів, поки не будуть виключені всі корені. Однак часто поліноми мають комплексно-спряжені корені. У цьому випадку початкове значення вибирається також комплексно-спряженим $z_n = x_n + iy_n$, і після знаходження пари таких коренів вони виключаються з полінома одночасно:

$$(z - z_n)(z - \overline{z}_n) = z^2 - 2z\,\mathrm{Re}|z_n| + |z_n|^2 = z^2 - 2x_n z + x_n^2 + y_n^2.$$

Схема Горнера (6.28) при цьому змінюється:

$$\begin{aligned} &p(z_m) = b_{m-1} z_n + b_m;\\ &b_{-1} = 0; \quad b_0 = a_0;\\ &b_i = a_i + q b_{i-1} + s b_{i-2}, \quad i = 1, 2, \ldots, m-1; \quad b_m = a_0 + q b_{m-2};\\ &p'(z_n) = 2iy_n(c_{m-3} z_n + c_{m-2}) + b_{m-1};\\ &c_{-1} = 0; \quad c_0 = b_0;\\ &c_i = b_i + q c_{i-1} + s c_{i-2}, \quad i = 1, 2, \ldots, m-3; \quad c_{m-2} = b_{m-2} + s c_{m-4}. \end{aligned} \tag{6.30}$$

Тут

$$q = 2\,\mathrm{Re}|z_n| = 2x_n, \quad s = -|z_n|^2 = -(x_n^2 + y_n^2).$$

Після виключення комплексно-спряжених коренів продовжується розв'язування поліноміального рівняння, порядок якого на два менше і коефіцієнти якого знайдені за допомогою процедури Горнера:

$$g(z) = \frac{p(z) - p(z_n)}{(z - z_n)(z - \bar{z}_n)} = b_0 z^{m-2} + b_1 z^{m-3} + \ldots + b_{m-2}.$$

Приклад 6.7

Знайдемо комплексно-спряжені корені полінома $x^2 + 1 = 0$ для початкового комплексного значення $x_0 = 1 + i$. Скориставшись формулою методу Ньютона (6.13), проведемо обчислення:

```
In[]:= f[x_]:= x^2 + 1;
       fl[x_] = Dt[f[x], x];
       x0 = 1 + 1*I;
       n = 0; ϵ = 0.001; eps = 1; x = x0;
       While[eps > ϵ, x = N[x - f[x]/fl[x]]; eps = Abs[x0 - x]; x0 = x; n++;
       Print["n = ", n, " x = ", x, " f[x] = ", f[x], " Δx = ", eps] ];
```

```
n = 1  x =  0.25 + 0.75i              f[x] =  0.5 + 0.375i                    Δx = 0.790569
n = 2  x = -0.075 + 0.975i            f[x] =  0.055 - 0.14625i                Δx = 0.395285
n = 3  x =  0.00171569 + 0.997304i    f[x] =  0.00538783 + 0.00342212i        Δx = 0.0798922
n = 4  x = -4.64185×10^-6 + 1. i      f[x] = -4.32096×10^-6 - 9.28371×10^-6 i Δx = 0.0032
n = 5  x = -1.00287×10^-11 + 1. i     f[x] =  1.68789×10^-11 - 2.00574×10^-11 i Δx = 5.12*10^-6
```

Вже після п'ятої ітерації переконуємось, що $\alpha = i$.

6.3.3. Визначення екстремальних точок функції методом Ньютона

Задачу обчислення значень аргументу функції x_e, за яких функція $f(x_e)$ досягає своїх екстремальних значень (максимального чи мінімального), можна звести до задачі розв'язання нелінійних рівнянь, оскільки в даних точках похідна від функції дорівнює нулю, тобто $f'(x_e) = 0$. При цьому формула (6.13) набуває такого вигляду:

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}. \tag{6.31}$$

Приклад 6.8

Знайдемо, користуючись формулою (6.31), координати екстремальної точки функції $f(x) = \sin 2(x + \pi/3) - (x/2)^2$, розглянутої в прикладі 6.1 і відображеної на рис. 6.2. Для цього запишемо і виконаємо програму, яка реалізує формулу (6.31):

```
In[]:= f[x_]:= Sin[x + π/3]^2 - (x/2)^2;
       fl[x_] = Dt[f[x], x]
       x0 = 0; n = 0; ϵ = 0.01; eps = 1; x = x0;
```

```
While[eps > ϵ,
      x = x - N[f1[x]]/N[f2[x]]; eps = Abs[x0 - x]; x0 = x; n++;
Print["n = ", n, "x = ", x, "f[x] = ", N[f[x]], "f1[x] = ", N[f1[x]], "Δx = ", eps] ];
```

Отримаємо результати:

```
n = 0  x = 0         f[x] = 0.75       f1[x] = 0.866025        Δx = 1
n = 1  x = 0.57735   f[x] = 0.91378    f1[x] = -0.395971       Δx = 0.57735
n = 2  x = 0.418227  f[x] = 0.945209   f1[x] = 0.0000737573    Δx = 0.159123
n = 3  x = 0.418257  f[x] = 0.945209   f1[x] = -3.77369×10^-10  Δx = 0.0000300345
```

6.3.4. Точність обчислень у разі застосування методу Ньютона

Оскільки метод Ньютона є базовим у програмних комплексах математичного моделювання об'єктів і систем, розглянемо похибку обчислень даного методу. Для цього скористаємося відповідними співвідношеннями типу (6.8) і (3.8), виведеними раніше для методів простої ітерації (нелінійного й лінійного).

Зведемо формулу (6.13) методу Ньютона до виду, характерного для методу простої ітерації:

$$x_{n+1} = \varphi(x_n) + \delta_n, \tag{6.32}$$

де

$$\varphi(x) = x - \frac{f(x)}{f'(x)},$$

$$\varphi'(x) = 1 + \frac{f''(x)f(x) - (f'(x))^2}{(f'(x))^2} = -\Delta x_n \frac{f''(x)}{f'(x)},$$

$$\Delta x_n = -\frac{f(x)}{f'(x)},$$

δ_n — похибка обчислень.

Тоді за аналогією з виразом (3.23) зв'язок глобальної й локальної похибок визначається співвідношенням

$$|x_{n+1} - \alpha| < \frac{m}{1-m}(x_{n+1} - x_n) + \frac{\delta_n}{1-m}, \tag{6.33}$$

де, як і раніше у формулі (6.16), $|k\Delta x_n| < m < 1$ і $k > |f''(x)/f'(x)|$.

Приклад 6.9

Оцінимо похибку першого кореня після другої ітерації ($m = 2$) із прикладу 6.4 для рівняння

$$f(x) = \sin^2(x + \pi/3) - (x/2)^2,$$

коли $x_2 = -0{,}688728$ і $\Delta x_2 = 0{,}105677$.

Перша і друга похідні від правої і лівої частин розглянутого рівняння мають такий вигляд:

$$f'(x) = 2\sin(x + \pi/3)\cos(x + \pi/3) - (x/2);$$
$$f''(x) = 2\cos^2(x + \pi/3) - 2\sin^2(x + \pi/3) - 1/2 = 2\cos(2x + 2\pi/3) - 1/2.$$

Звичайно, можна було б за допомогою пакета Mathematica оцінити максимально можливі значення цих похідних для обчислення коефіцієнта k у виразі (6.33), як у прикладах 6.2 і 6.3. Але можна зробити це і наближено, приймаючи $x = -0,7$ і вибираючи максимально можливі значення тригонометричних функцій. У результаті отримаємо $|f'(x)| < 1$, $|f''(x)| < 3/2$.
Тоді

$$m = |k\Delta x_2| \leqslant \frac{3}{2} \cdot 1,05677 \cdot 10^{-1} = 0,158516,$$

$$(x_2 - \alpha) < \frac{0,158516 \cdot 0,105677 + 1/2 \cdot 10^{-6}}{1 - 0,158516} < 1,99 \cdot 10^{-2}.$$

6.4. Метод січних

Застосовуючи формулу (6.13) чи (6.25) методу Ньютона, необхідно точно обчислювати похідні від розглянутих функцій, бажано аналітично. На практиці під час обчислень похідні апроксимують розділеними першими різницями:

$$f'(x_n) = \frac{f_n - f_{n-1}}{x_n - x_{n-1}}.$$

Використовуючи апроксимації у формулі методу Ньютона, можна перейти до *методу січних*:

$$x_{n+1} = x_n + \Delta x_n, \quad \Delta x_n = -f_n \frac{x_n - x_{n-1}}{f_n - f_{n-1}}. \tag{6.34}$$

Щоб реалізувати цей метод, необхідні два початкових значення, за допомогою яких здійснюється локальна апроксимація функції січними (рис. 6.12).

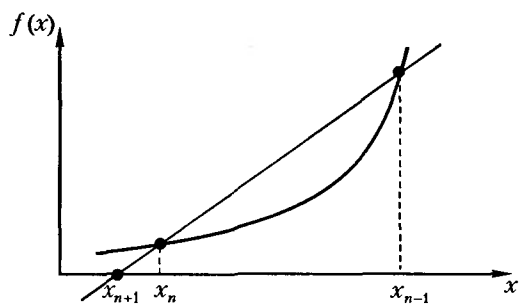


Рис. 6.12. Локальна апроксимація функції в методі січних

Якщо ми маємо справу з кратними коренями, за аналогією з (6.25) можна записати:

$$x_{n+1} = x_n - u'(x_n)\frac{x_n - x_{n-1}}{u(x_n) - u(x_{n-1})},$$

де $u(x) = f(x)/f'(x)$.

Приклад 6.10

Розв'яжемо методом січних рівняння

$$f(x) = \sin^2(x + \pi/3) - (x/2)^2,$$

розглянуте в прикладах 6.1 і 6.4 і розв'язане методами дихотомії і методом Ньютона. Для обчислення першого кореня оберемо два початкових наближення $x_0 = -1{,}1$ і $x_1 = -1$ і скористаємося програмою для пакета Mathematica:

```
In[]:= f[x_]:= Sin[x + π/3]^2 - (x/2)^2;
       x[0] = -1.1; x[1] = -1; n = 0; ϵ = 0.01; eps = 1;
       While[eps > ϵ, x[n+1] = x[n] - N[f[x[n]]*(x[n] - x[n-1])/(f[x[n]] - f[x[n-1]])];
         eps = Abs[x[n+1] - x[n]]; n++;
         Print["n = ", n, "x = ", x[n], "f[x] = ", N[f[x[n]]], "Δx = ", eps] ];
```

Отримаємо результат:

```
n = 0  x = -1.1        f[x] = -0.299714      Δx = 0.1
n = 1  x = -1          f[x] = -0.247774      Δx = 0.1
n = 2  x = -0.522965   f[x] = 0.182176       Δx = 0.477035
n = 3  x = -0.725092   f[x] = -0.0312261     Δx = 0.202126
n = 4  x = -0.695515   f[x] = -0.00227064    Δx = 0.0295762
n = 5  x = -0.693196   f[x] = 0.0000387793   Δx = 0.00231932
```

Для локалізації другого кореня змінимо початкові наближення на $x_0 = 2$ і $x_1 = 2{,}1$.

```
n = 0  x = 2           f[x] = -0.991116      Δx = 0.1
n = 1  x = 2.1         f[x] = -1.10247       Δx = 0.1
n = 2  x = 1.10993     f[x] = 0.385863       Δx = 0.99007
n = 3  x = 1.36661     f[x] = -0.0243988     Δx = 0.256684
n = 4  x = 1.35135     f[x] = 0.0011624      Δx = 0.0152654
n = 5  x = 1.35204     f[x] = 1.56587×10^-6  Δx = 0.000694196
```

Порівнюючи таблиці результатів прикладів 6.1 і 6.4, можна побачити, що збіжність методу хорд трохи повільніша, ніж збіжність методу Ньютона, але набагато швидкіша, ніж збіжність методу дихотомії.

Її можна оцінити, повторивши викладки, зроблені під час виведення формули (6.15). Отримаємо вирази

$$\varepsilon_{n+1} = \varepsilon_n\varepsilon_{n-1}\frac{1}{2}\frac{f''(\xi)}{f'(\xi)}, \quad |\varepsilon_{n+1}| \approx m|\varepsilon_n||\varepsilon_{n-1}|, \tag{6.35}$$

які можна переписати як

$$|\varepsilon_{n+1}| \approx k|\varepsilon_n|^p, \quad |\varepsilon_n| \approx k|\varepsilon_{n-1}|^p, \tag{6.36}$$

при цьому

$$p = \frac{1}{2}(1 + \sqrt{5}) = 1{,}618. \tag{6.37}$$

Таким чином, збіжність методу хорд залишається нелінійною з показником 1,618, а не 2, як у методі Ньютона.

Природно, що були запропоновані методи, в яких не потрібно обчислювати похідну (як у методі хорд), але збіжність яких квадратична (як у методі Ньютона). Наприклад, таким є метод Стефенсона, відповідно до якого

$$x_{n+1} = x_n - \frac{f(x_n)}{g(x_n)}, \quad g(x_n) = \frac{f[x_n + f(x_n)] - f(x_n)}{f(x_n)} \tag{6.38}$$

і для якого

$$\frac{\varepsilon_{n+1}}{\varepsilon_n^2} \to \frac{1}{2}\frac{f''(\alpha)}{f'(\alpha)}(1 + f'(\alpha)). \tag{6.39}$$

6.4.1. Метод хорд

Об'єднання процедур методів січних і дихотомії дозволило побудувати *метод хорд* (або *метод помилкової точки*), в якому передбачено вибір початкових значень для ітерації x_n і x_{n-1} з умови $f(x_n)f(x_{n-1}) < 0$ (рис. 6.13). При цьому кількість ітерацій у порівнянні з методом дихотомії зменшується за рахунок розбиття інтервалу ізоляції кореня не навпіл, а у відношенні $f(a)/f(b)$. Чергове наближення визначається за формулою:

$$x_{n+1} = x_n - \frac{f(x_n)(x_n - c)}{f(x_n) - f(c)}, \tag{6.40}$$

де c – нерухома точка.

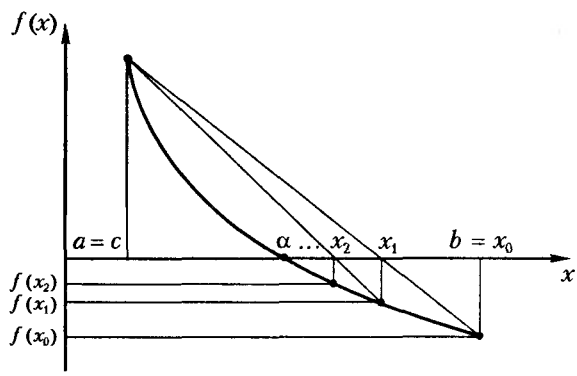


Рис. 6.13. Ітераційний процес методу помилкової точки

За значення c береться той із кінців інтервалу локалізації $[a, b]$, для якого $f(x)f''(x) > 0$. Інший кінець інтервалу приймається за початкове наближення x_0. Ітераційний процес закінчується в разі виконання умови

$$\frac{1}{m}|f(x_{n+1})| < \varepsilon,$$

де $m \leqslant \min\limits_{[a,b]}|f'(x)|$, тому що існує оцінка

$$|x_{n+1} - \alpha| \leqslant \frac{1}{m}|f(x_{n+1})|.$$

На жаль, метод помилкової точки остаточно втрачає властивість нелінійної збіжності, демонструючи таку ж лінійну залежність між похибками двох сусідніх ітерацій, як метод дихотомії та метод простої ітерації. Проте, як і метод дихотомії, він збігається завжди.

6.4.2. Комбінований метод

Оскільки в методах помилкової точки і дотичних наближення кореня обчислюється відповідно з недостачею і з надлишком (залежно від вигляду кривої), був розроблений метод, який об'єднав обидва підходи (рис. 6.14). На кожній ітерації нового методу спочатку використовується формула дотичних (6.13), а потім формула методу хорд (6.40), в якій за c приймається значення, обчислене на даному кроці за формулою (6.13). Процес закінчується, коли

$$\left|x_n^{(\text{н})} - x_n^{(c)}\right| < 2\varepsilon.$$

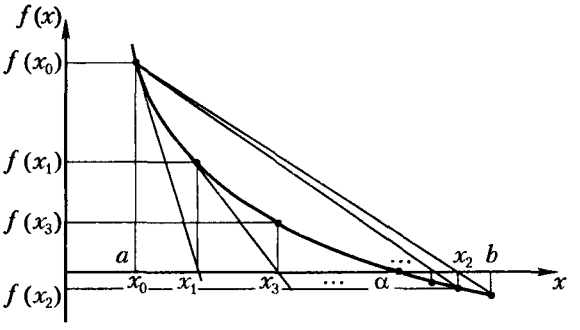


Рис. 6.14. Ітераційний процес комбінованого методу

Кінцеве наближення обчислюється за формулою

$$x_{n+1} = \frac{x_n^{(\text{н})} - x_n^{(c)}}{2}, \qquad (6.41)$$

де $x_n^{(\text{н})}$ і $x_n^{(c)}$ — наближення кореня, отримані методами дотичних (Ньютона) та помилкової точки.

Через те, що ці методи мають різну збіжність (метод дотичних — квадратичну, а метод хорд — лінійну), ефективність комбінованого методу на практиці виявляється нижче очікуваної.

Приклад 6.11

Обчислимо корінь рівняння, заданого в прикладі 6.2, комбінованим методом. Знайдемо початкове наближення для методу дотичних. Нехай початкове наближення $x_0 = 1$. Визначимо першу і другу похідні функції:

```
In[]:= f[x_]:= x^(2.5) - Exp[-x^(1.2)];
       f1[x_] = Dt[f[x], x];
       f2[x_] = Dt[f[x], {x,2}]; f2[1]*f[1]
Out[]= 2.0914
```

Отримане значення додатне, отже, умова збіжності виконана.

Для методу хорд як початкоае наближення необхідно задати два значення x_0 і x_1, таких щоб була виконана умова $f(x_0)f(x_1) < 0$. Виберемо, наприклад, $x_0 = 1$, $x_1 = 0{,}5$, тоді остання умова буде виконаною. Обчислення послідовних наближень виконується за формулами (6.13) і (6.40):

```
In[]:= z1 = 0.5; z0 = 1; x = 1.; n = 0; ϵ = 0.0001; epk = 10;
       While[epk > ϵ, z1 = N[z1 - f[z1]*(z0 - z1)/(f[z0] - f[z1])];
       x = N[x - f[x]/f1[x]]; epk = Abs[z1 - x]; z0 = x; n++;
       Print[" n = ", n, "x = ", x, "z1 = ", z1, "epk = ", epk] ];
n = 0  x =  0.785099  z1 =  0.713306  epk =  0.0717935
n = 1  x =  0.753154  z1 =  0.751687  epk =  0.00146695
n = 2  x =  0.752497  z1 =  0.752497  epk =  6.09456×10^-7
```

Отримані результати показують, що за три ітерації знайдено значення шуканого кореня з похибкою $6{,}09456 \cdot 10^{-7}$.

Розв'язок рівняння можна отримати за допомогою оператора пакета Mathematica, що дає такий самий результат:

```
In[]:= FindRoot[f[x] == 0, {x,1}]
Out[]= {x → 0.752497}
```

6.5. Метод Мюллера

У разі застосування методу Мюллера для поліпшення збіжності ітераційних процедур локальна лінійна апроксимація функції заміняється нелінійною параболічною за допомогою інтерполяційного полінома Ньютона (рис. 6.15):

$$f(x) = f(x_n) + f(x_{n-1}, x_n)(x - x_n) + f(x_{n-2}, x_{n-1}, x_n)(x - x_{n-1})(x - x_n), \qquad (6.42)$$

де застосовані перша і друга розділені різниці

$$f(x_{n-1}, x_n) = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}},$$

$$f(x_{n-2}, x_{n-1}, x_n) = \frac{f(x_n) - 2f(x_{n-1}) + f(x_{n-2})}{(x_n - x_{n-1})(x_{n-1} - x_{n-2})}.$$

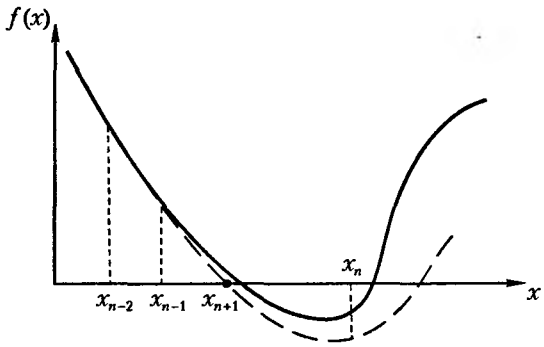


Рис. 6.15. Ітераційний процес методу Мюллера

Ідея методу полягає в тому, що процес пошуку коренів рівняння (6.1) замінюється обчисленням коренів інтерполяційного многочлена (6.42), побудованого для $f(x)$.

Розв'яжемо рівняння $f(x) = 0$. Позначимо різницю наближень на двох сусідніх ітераціях як $z = x_{n+1} - x_n$ і запишемо рівняння (6.42) для випадку $x = x_{n+1}$ у такому вигляді:

$$\begin{aligned} &f(x_n) + f(x_{n-1}, x_n)z + f(x_{n-2}, x_{n-1}, x)z^2 + \\ &+f(x_{n-2}, x_{n-1}, x)z(z + x_n - x_{n-1}) = \\ &= f(x_{n-2}, x_{n-1}, x)z^2 + \omega z + f(x_n) = 0, \end{aligned} \tag{6.43}$$

де

$$\omega = f(x_{n-1}, x_n) + f(x_{n-2}, x_{n-1}, x_n)(x_n - x_{n-1}). \tag{6.44}$$

Якщо продиференціювати рівняння (6.42):

$$f'(x) = f(x_{n-1}, x_n) + f(x_{n-2}, x_{n-1}, x_n)[(x - x_{n-1}) + (x - x_n)]$$

і обчислити похідну для $x = x_n$, можна переконатися, що

$$\omega = f'(x_n) = f(x_{n-1}, x_n) + f(x_{n-2}, x_{n-1}, x_n)(x_n - x_{n-1}). \tag{6.45}$$

Розв'язуючи рівняння (6.43), отримаємо два кореня $z^{(1)}$ і $z^{(2)}$, на основі яких обчислюються корені $x^{(1)}$ і $x^{(2)}$. Як наступне наближення в методі Мюллера вибирається те значення $x^{(1)}$ і $x^{(2)}$, яке ближче до x_n. Зокрема, для рівняння (6.43) знаходимо:

$$x_{n+1} - x_n = \frac{-\omega \pm \sqrt{\omega^2 - 4f(x_n)f(x_{n-2}, x_{n-1}, x_n)}}{2f(x_{n-2}, x_{n-1}, x_n)} =$$

$$= \frac{\omega^2 - [\omega^2 - 4f(x_n)f(x_{n-2}, x_{n-1}, x_n)]}{2f(x_{n-2}, x_{n-1}, x_n)\left[-\omega \pm \sqrt{\omega^2 - 4f(x_n)f(x_{n-2}, x_{n-1}, x_n)}\right]}.$$

Після очевидних спрощень із урахуванням (6.45) остаточно отримуємо:

$$x_{n+1} = x_n - \frac{2f(x_n)}{-\omega \pm \sqrt{\omega^2 - 4f(x_n)f(x_{n-2}, x_{n-1}, x_n)}}. \qquad (6.46)$$

Метод Мюллера зручний тим, що в загальному випадку дозволяє отримати комплексно-спряжені корені рівняння (6.1), використовуючи дійсні початкові наближення x_n, x_{n-1}, x_{n-2}.

Приклад 6.12

Знайдемо методом Мюллера корені рівняння

$$f(x) = \sin x + \cos x - 2 = 0.$$

Для пошуку відрізка локалізації побудуємо графік функції $f(x)$, показаний на рис. 6.16:

```
In[]:= f[x_]:= Sin[x] + Cos[x] - 2;
       Plot[f[x], {x, 0, 1.5}, AxesLabel → {"x", "f(x)"}];
```

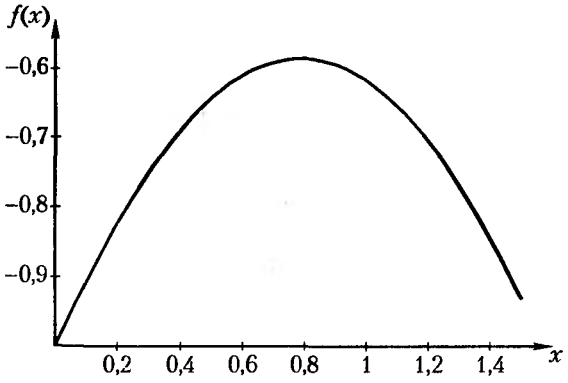


Рис. 6.16. Графік функції $\sin x + \cos x - 2$

На графіку видно, що розглянуте рівняння не має дійсних коренів (графік не перетинає вісь абсцис). Виберемо три довільних значення для аргументу, наприклад $x = 0; 0{,}75$ та 1,5. Складемо таблицю значень функції в цих точках:

```
In[]:= T = Table[{x, f[x]}, {x, 0, 1.5, 0.75}]
Out[]= {{0, -1}, {0.75, -0.586672}, {1.5, -0.931768}}
```

Використовуючи значення функції, отримаємо інтерполяційний многочлен, яким замінимо функцію $f(x)$:

```
In[]:= P = InterpolatingPolynomial[T, x]
Out[]= -1 + (0.551104 - 0.674154 (-0.75 + x))x
```

Знайдемо нулі отриманого многочлена:

```
In[]:= Z = Solve[P == 0, x]
Out[]= {{x → 0.783737 - 0.932254 i}, {x → 0.783737 + 0.932254 i}
```

Таким чином, знайдено наступне, вже комплексне, наближення коренів рівняння. Використаємо їх для подальшого уточнення коренів:

```
In[]:= a = x/.Z[[1]]; b = x/.Z[[2]];
```

Внесемо один з них, наприклад a, у таблицю T. Спочатку виділимо перший елемент списку T, а потім допишемо рядок a:

```
In[]:= T1 = Rest[T]; T = Append[T1, {a, f[a]}]
Out[]= {{0.75, -0.586672}, {1.5, -0.931768},
        {0.783737 - 0.932254i, 0.0745729 - 0.00252109i}}
```

Повторюючи обчислення наближеного значення кореня, отримаємо:

```
Out[]= {{x → 0.786903 - 0.877001i}, {x → 0.835196 + 0.909285i}}
```

Виконавши знову такі ж дії, отримаємо більш точне наближення кореня. Для перевірки обчислимо корені цього ж рівняння вбудованими методами пакета Mathematica і порівняємо їх із отриманими вище значеннями:

```
In[]:= NSolve [f[x] == 0, x]
Out[]= {{x → 0.785398 - 0.881374i}, {x → 0.785398 + 0.881374i}}
```

Отже, корінь знайдений досить точно. Якщо отримана точність недостатня, то потрібно виконати ще кілька ітерацій. Аналогічно знаходять другий корінь рівняння.

6.6. Методи розширення області розв'язку

Одним із недоліків методу Ньютона є його залежність від початкового наближення, і тому процес обчислення збігається лише в незначному околі точного розв'язку рівняння. Існують методи, що дозволяють істотно розширити область вибору початкового значення; до них належать метод *продовження розв'язку за параметром* і метод *пошуку кривої розв'язку*.

6.6.1. Метод продовження розв'язку за параметром (метод Давиденка)

Відповідно до методу продовження розв'язку за параметром початкову задачу $f(x) = 0$, яку розв'язати досить важко, перетворюють на послідовність задач, кожна з яких залежить від параметра, який додатково вводиться у рівняння:

$$F(x, k) = 0. \tag{6.47}$$

Додатковий параметр k вибирається таким чином, щоб виконувались умови

$$F(x, 1) = f(x),$$
$$F(x, 0) = g(x),$$

де $g(x) = 0$ – задача, яку легко розв'язати.

У результаті розв'язок задачі стає функцією від введеного параметра k. Найбільш складним і творчим в цьому методі є саме вибір початкової задачі $g(x) = 0$

і способу введення параметра k. Наскільки швидко і вдало дослідник зможе вибрати цей параметр, залежить від його обізнаності з класом задач, до яких належить початкова задача $f(x) = 0$.

Після формулювання задачі $g(x) = 0$ параметр k у рівняння найпростіше ввести через композицію функцій $f(x)$ і $g(x)$:

$$F(x,k) = kf(x) - (1-k)g(x), \quad 0 \leqslant k \leqslant 1. \tag{6.48}$$

Фіксуючи значення параметра k_i в межах $0 < k_i < 1$, отримуємо проміжну нелінійну задачу $F(x, k_i) = 0$ для поточного значення k_i, яку розв'язуємо за допомогою методів, розглянутих вище. При цьому початкове значення під час розв'язування вибираємо чи з умови припасовування $x_0(k_i) = x(k_{i-1})$, чи на основі інтерполяції двох попередніх рішень

$$x_0(k_i) = x(k_{i-1}) + [x(k_{i-1}) - x(k_{i-2})]\frac{k_i - k_{i-1}}{k_{i-1} - k_{i-2}}. \tag{6.49}$$

Є й інша можливість розв'язати рівняння (6.47), диференціюючи його за параметром k і зводячи його до еквівалентного диференційного рівняння

$$\frac{\partial F}{\partial x}\frac{dx}{dk} + \frac{\partial F}{\partial k} = 0,$$

звідки

$$\frac{dx}{dk} = \left(\frac{\partial F}{\partial x}\right)^{-1}\frac{\partial F}{\partial k}. \tag{6.50}$$

Застосування методів розв'язання звичайних диференціальних рівнянь, розглянутих у розділах 8–10, не гарантує, що розв'язати рівняння (6.50) у разі неперервної зміни параметра k можна більш ефективно, ніж за умови дискретної зміни значень k_i і використання виразу (6.48).

Наприклад, під час моделювання електронних схем дискретний параметр k, що вводиться в рівняння моделей, дозволяє змінювати напругу загального джерела живлення від малих значень до номінального, якщо розв'язок задачі сталих станів за номінального значення напруги не збігається через наявність у моделі експоненціальних рівнянь $p-n$ переходів транзисторів.

Приклад 6.13

Розв'яжемо методом Давиденка експоненціальне рівняння, вибравши $x_0 = 0$,

$$f(x) = x + (e^{x/0.25} - 1) - 50 = 0.$$

Можна показати, що спроба розв'язати це рівняння методом Ньютона для $x_0 = 0$ виявиться невдалою, — навіть після виконання 30 ітерацій, збіжність обчислень не дося-

гається. Проте рівняння $\varphi(x) = x + (e^{x/0,25} - 1) - 5 = 0$ збігається всього за сім ітерацій, якщо $x_{01} = 0$:

```
In[]:= F[x_]:= x + (Exp[x/0.25] - 1) - 5;
       Fl[x_] = Dt[F[x], x];
       x0 = 0; n = 0; ϵ = 0.001; eps = 10; x = x0;
       While[eps > ϵ, x = x - F[x]/Fl[x]; eps = Abs[x0 - x]; x0 = x; n++;
       Print["n = ", n, "x = ", x, "F[x] = ", F[x]]];
```

Отримаємо розв'язок експоненціального рівняння $\varphi(x) = 0$:

```
n = 0  x = 0          F[x] = -5
n = 1  x = 1.         F[x] = 49.5982
n = 2  x = 0.77393    F[x] = 16.8771
n = 3  x = 0.585175   F[x] = 4.97367
n = 4  x = 0.468296   F[x] = 0.977275
n = 5  x = 0.432148   F[x] = 0.0648744
n = 6  x = 0.429391   F[x] = 0.000341258
n = 7  x = 0.429377   F[x] = 9.57346*10^-9
```

Тому початкове рівняння зводиться до (6.47) введенням параметра k:

$$F(x, k) = f(x) = x + (e^{x/0,25} - 1) - 50k - 5(1 - k) = 0.$$

Із останнього виразу видно, що за умови $k = 0$ це рівняння збігається з рівнянням $\varphi(x) = 0$, розв'язок якого наведено вище.
Тепер вибираємо $k = 0,4$ і у разі $x_{02} = 0,429377$, взятого з попереднього розв'язку $\varphi(x) = 0$, розв'язуємо нове рівняння:

```
In[]:= Fk[x_]:= x + (Exp[x/0.25] - 1) - 23;
       Fkl[x_] = Dt[F[x], x];
       x0 = 0.429377; n = 0; ϵ = 0.001; eps = 10; x = x0;
       While[eps > ϵ,
          x = x - Fk[x]/Fkl[x]; eps = Abs[x0 - x]; x0 = x; n++;
          Print["n = ", n, "x = ", x, "F[x] = ", Fk[x]] ];
```

Для $k = 0,4$ обчислення збігаються за шість ітерацій:

```
n = 0  x = 0.429377  F[x] = -18.
n = 1  x = 1.20249   F[x] = 99.9287
n = 2  x = 0.999343  F[x] = 31.4541
n = 3  x = 0.855598  F[x] = 7.4982
n = 4  x = 0.794918  F[x] = 0.833808
n = 5  x = 0.786336  F[x] = 0.0140038
n = 6  x = 0.786187  F[x] = 4.13115*10^-6
```

І, нарешті, якщо $k = 1$, повертаємося до вихідного рівняння (рис. 6.17)

$$F(x, 1) = f(x) = x + (e^{x/0,25} - 1) - 50 = 0$$

і розв'язуємо його для $x_{03} = 0,786187$, взятого з попереднього розв'язку, для $k = 0,4$:

```
In[]:= Fk[x_]:= x + (Exp[x/0.25] - 1) -50;
       Fkl[x_] = Dt[F[x], x];
       x0 = 0.786187; n = 0; ϵ = 0.001; eps = 10; x = x0;
       While[eps > ϵ, x = x - Fk[x]/Fkl[x]; eps = Abs[x0 - x]; x0 = x; n++;
       Print["n = ", n, "x = ", x, "F[x] = ", Fk[x]]];
```

У результаті отримуємо результат усього за п'ять ітерацій:

```
n = 0  x = 0.786187  F[x] = -27.
n = 1  x = 1.07386   F[x] = 23.4395
n = 2  x = 0.994263  F[x] = 3.35374
n = 3  x = 0.978623  F[x] = 0.10227
n = 4  x = 0.978116  F[x] = 0.000103231
```

Таким чином, розв'язок рівняння, що не збігався у разі застосування методу Ньютона, за допомогою методу продовження розв'язку за параметром був знайдений за 19 ітерацій.

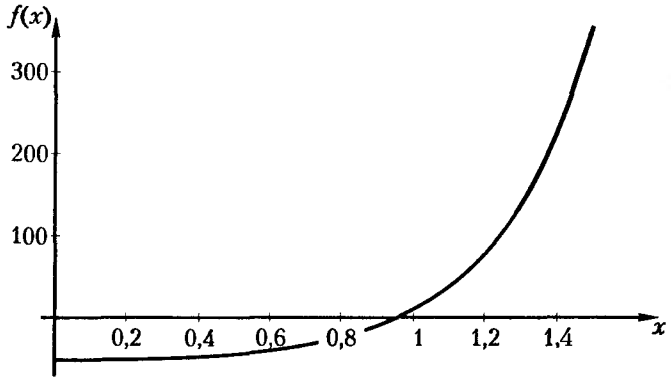


Рис. 6.17. Графік функції $x+(e^{x/0.25}-1)-50$

6.6.2. Метод пошуку кривої розв'язку

Метод пошуку кривої розв'язку істотно збільшує область збіжності розв'язку нелінійної задачі, забезпечуючи швидкий рух від точки початкового наближення в окіл точки розв'язку. Виконувані при цьому операції ілюструються на рис. 6.18.

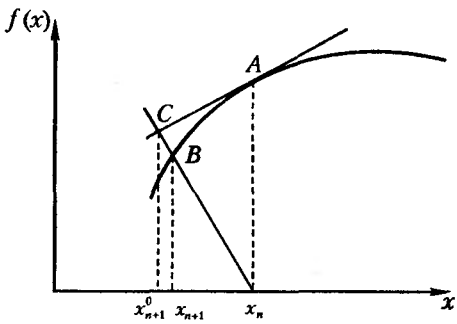


Рис. 6.18. Ітерація в методі пошуку розв'язку

Припустимо, що x_n — точка поточного наближення. У ній будується дотична, і на цю дотичну із точки, що лежить на осі x, проводиться перпендикуляр, який перетинає дотичну в точці C і криву функції в точці B. Координата точки B приймається за наступне наближення x_{n+1}, у ній проводиться нова дотична і т. д.

Щоб знайти значення x_{n+1}, використовуються ітерації за стандартним методом Ньютона, початкове значення x_{n+1}^0 для якого задається координатою точки перетину перпендикуляра з дотичною. Можна ввести додатковий параметр λ, за допомогою якого контролюють кут перетину додаткової прямої з дотичною, змінюючи його від прямого кута ($\lambda_0 = 1$) до збігу цієї лінії з віссю x ($\lambda_0 = 0$), що відповідає звичайному методу Ньютона.

Координата точки C обчислюється, виходячи з умови перетину двох перпендикулярних прямих:

$$f'(x_n)\left(x_{n+1}^0 - x_n\right) + f(x_n) = -\frac{\lambda}{f'(x_n)}\left(x_{n+1}^0 - x_n\right)$$

або

$$\{\lambda + f'^2(x_n)\}(x_{n+1}^0 - x_n) + f(x_n)f'(x_n) = 0,$$

звідки

$$x_{n+1}^0 = x_n - \frac{f(x_n)f'(x_n)}{\lambda + f'^2(x_n)}$$

або

$$x_{n+1}^0 = x_n - \frac{f(x_n)}{f'(x_n)}\frac{1}{1 + \dfrac{\lambda}{f'^2(x_n)}}. \qquad (6.51)$$

Координату точки B обчислюють, виходячи з умови перетину кривої функції з додатковою прямою (перпендикуляром):

$$f(x_{n+1}) = -\frac{\lambda}{f'(x_n)}(x_{n+1} - x_n),$$

за якою будується рівняння

$$p(x) = \lambda(x_{n+1} - x_n) + f(x_{n+1})f'(x_n),$$

що розв'язують методом Ньютона:

$$x_{n+1}^{(k+1)} = x_{n+1}^{(k)} - \frac{p(x_{n+1}^{(k)})}{p'(x_{n+1}^{(k)})}.$$

З урахуванням значення похідної

$$p'(x_{n+1}) = \lambda + f'(x_{n+1})f'(x_n)$$

остаточно отримуємо:

$$x_{n+1}^{(k+1)} = x_{n+1}^{(k)} - \frac{\lambda(x_{n+1}^{(k)} - x_n) + f(x_{n+1}^{(k)})f'(x_n)}{\lambda + f'(x_{n+1}^{(k)})f'(x_n)}. \qquad (6.52)$$

Приклад 6.14

Методом пошуку кривої розв'язку знайдемо корені рівняння

$$f(x) = 3\operatorname{arctg}(3 - x) = 0,$$

яке не можна розв'язати методом Ньютона, вибравши $x_0 = 0$. Графік функції показаний на рис. 6.19.

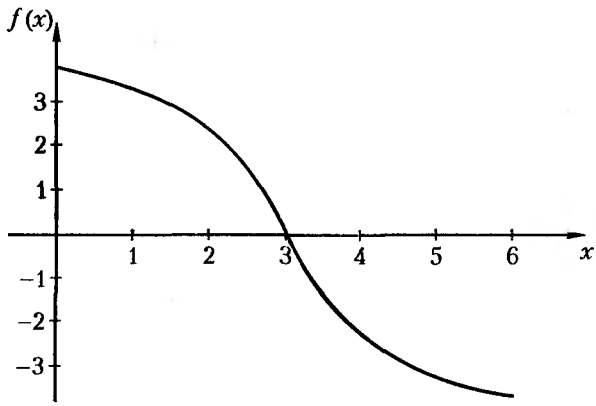


Рис. 6.19. Графік функції $3\operatorname{arctg}(3 - x)$

Якщо розв'язувати задане рівняння, використовуючи формули (6.51) і (6.52), для $\lambda = 1$ і $x_0 = 0$, то отримаємо наближення, які швидко збігаються до сталого значення всього за п'ять ітерацій (рис. 6.20):

```
In[]:= f[x_]:= 3*ArcTan[3. - x];
       fl[x_] = Dt[f[x], x];
       x[0] = 0;
       n = 0;
       ϵ = 0.01; eps = 1;
       While[eps > ϵ,
       u[n+1] = x[n] - f[x[n]]*fl[x[n]]/(1 + (fl[x[n]])^2);
       x[n+1] = u[n+1] - (u[n+1] - x[n] + f[u[n+1]]*fl[x[n]])/(1 + fl[u[n+1]]*fl[x[n]]);
       eps = Abs[x[n+1] - x[n]]; n++;
       Print["n = ", n, "x = ", x[n], "f[x] = ", N[f[x[n]]], "Δx = ", eps]];
```

Результати обчислень методом пошуку кривої розв'язку:

```
n = 0  x = 0         f[x] = 3.74714     Δx = 1
n = 1  x = 0.997051  f[x] = 3.32321     Δx = 0.997051
n = 2  x = 2.22013   f[x] = 1.98704     Δx = 1.22308
n = 3  x = 2.88148   f[x] = 0.353897    Δx = 0.661357
n = 4  x = 2.988     f[x] = 0.0360034   Δx = 0.106514
n = 5  x = 2.9988    f[x] = 0.00360098  Δx = 0.0108014
n = 6  x = 2.99988   f[x] = 0.000360099 Δx = 0.00108029
```

Той самий результат можна отримати за допомогою оператора `NSolve` пакета Mathematica, що дозволяє застосовувати пакет безпосередньо для розв'язання початкового рівняння:

```
In[]:= NSolve[3*ArcTan[3. - x], x]
Out[]= {{x → 3.}}
```

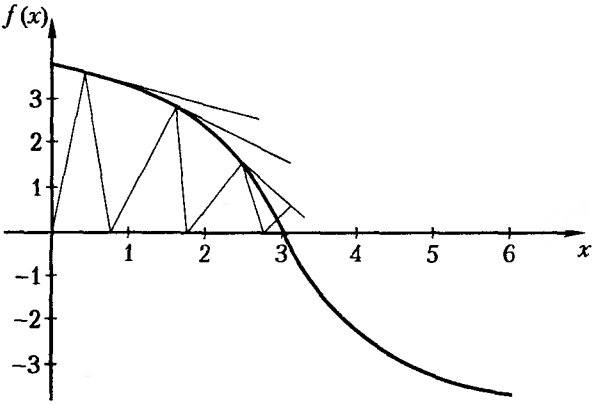


Рис. 6.20. Ітераційний процес методу пошуку кривої розв'язку

Однак, виходячи з наведеного прикладу, не можна стверджувати, що метод пошуку кривої розв'язку є універсальним методом для розв'язання всіх можливих задач. Неважко переконатися, що він значно програє методу Ньютона на лінійних ділянках функції, оскільки потребує виконання кількох ітерацій замість однієї, як це є у разі застосування методу Ньютона. Тому доцільно комбінувати дані методи, передбачивши можливість переходу в процесі розв'язування задачі від одного методу до іншого. Метод пошуку кривої розв'язку дозволяє полегшити обчислення на початковому етапі, знімаючи критичність вибору початкового наближення, а також у складних ситуаціях, коли, наприклад, норма функції починає рости $\|f(x_{n+1})\| > \|f(x_n)\|$.

6.7. Методи розв'язання систем нелінійних рівнянь

У загальному випадку система нелінійних рівнянь записується у вигляді

$$\begin{aligned} f_1(x_1, x_2, ..., x_m) &= 0, \\ f_2(x_1, x_2, ..., x_m) &= 0, \\ \cdots \quad \cdots \quad \cdots \\ f_m(x_1, x_2, ..., x_m) &= 0, \end{aligned} \tag{6.53}$$

де $f_i(x_1, x_2, ..., x_m)$, $i = 1, 2, ..., m$ — функції дійсних змінних $x_1, x_2, ..., x_m$.

Для зручності викладення запишемо систему (6.53) у вигляді операторного рівняння

$$\mathbf{F(x)} = 0, \tag{6.54}$$

де

$$\mathbf{x} = (x_1, x_2, ..., x_m)^T;$$

$$\mathbf{F(x)} = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_m(\mathbf{x}))^T.$$

Для розв'язання нелінійної системи (6.53) використовують ті ж методи, що і для розв'язання рівнянь з однією змінною.

За аналогією з (6.5) для методу простої ітерації можна записати:

$$\mathbf{x}^{(n+1)} = \Psi(\mathbf{x}^{(n)}), \tag{6.55}$$

де

$$\Psi(x) = (\varphi_1(x), \varphi_2(x), ..., \varphi_m(x))^{\mathrm{T}}.$$

На першій ітерації на основі початкового наближення наступне знаходять за формулами:

$$x_i^{(1)} = \varphi_i(x_1^{(0)}, x_2^{(0)}, ..., x_m^{(0)}), \quad i = 1, 2, ..., m.$$

У загальному вигляді, якщо знайдене n-е наближення $x_1^{(n)}, x_2^{(n)}, ..., x_m^{(n)}$, то $n+1$ наближення знаходять за формулами

$$x_i^{(n+1)} = \varphi_i(x_1^{(n)}, x_2^{(n)}, ..., x_m^{(n)}), \quad i = 1, 2, ..., m.$$

У разі, коли $n \to \infty$, $x_i^{(n)} \to \alpha_i$, $i = 1, 2, ..., m$ кажуть, що метод збігається до деякого розв'язку. Для того щоб отримати розв'язок із потрібною точністю ε, процес продовжують доти, доки два послідовних наближення будуть відрізнятися не більш ніж на задану величину $\|\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}\| < \varepsilon$.

Збіжність ітерацій забезпечується, якщо виконується умова, аналогічна (6.7)

$$\|\mathbf{J}(\mathbf{x})\| \leqslant M < 1, \tag{6.56}$$

де $\mathbf{J}(\mathbf{x}) = \Psi'$ — матриця Якобі, чи матриця частинних похідних системи функцій $\varphi_1, \varphi_2, ..., \varphi_m$:

$$\Psi'(\mathbf{x}) = \mathbf{J}(\mathbf{x}) = \begin{bmatrix} \frac{\partial \varphi_1}{\partial x_1} & \frac{\partial \varphi_1}{\partial x_2} & ... & \frac{\partial \varphi_1}{\partial x_m} \\ \frac{\partial \varphi_2}{\partial x_1} & \frac{\partial \varphi_2}{\partial x_2} & ... & \frac{\partial \varphi_2}{\partial x_m} \\ ... & ... & ... & ... \\ \frac{\partial \varphi_m}{\partial x_1} & \frac{\partial \varphi_m}{\partial x_2} & ... & \frac{\partial \varphi_m}{\partial x_m} \end{bmatrix}.$$

Швидкість збіжності методу простих ітерацій, як і раніше, лінійна:

$$\|\mathbf{x}^{(n+1)} - \boldsymbol{\alpha}\| = \|\varphi(\mathbf{x}^{(n)}) - \varphi(\boldsymbol{\alpha})\| \leqslant M \|\mathbf{x}^{(n)} - \boldsymbol{\alpha}\|, \quad n = 0, 1, 2, ...,$$

де $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, ..., \alpha_m)^{\mathrm{T}}$ — точний розв'язок системи (6.53).

Одним із серйозних недоліків методу простих ітерацій є складність вибору функцій φ_i, $i = 1, 2, ..., m$, які б задовольняли достатню умову збіжності (6.56). Тому в більшості випадків для розв'язання системи нелінійних рівнянь застосовують узагальнений метод Ньютона, для якого за аналогією з (6.13) записується матрично-векторна процедура:

$$\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)} - \frac{\mathbf{F}(\mathbf{x}^{(n)})}{\mathbf{W}(\mathbf{x}^{(n)})}, \quad n = 0, 1, 2, ..., \tag{6.57}$$

де використовується інша матриця Якобі, обумовлена системою (6.53):

$$\mathbf{F}'(\mathbf{x}) = \mathbf{W}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \dots & \frac{\partial f_1}{\partial x_m} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \dots & \frac{\partial f_2}{\partial x_m} \\ \dots & \dots & \dots & \dots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \dots & \frac{\partial f_m}{\partial x_m} \end{bmatrix}. \qquad (6.58)$$

Щоб уникнути процедури обернення матриці Якобі, матричне рівняння (6.56) переписують у вигляді системи лінійних рівнянь

$$\mathbf{A}(\mathbf{x}^{(n)})\mathbf{x}^{(n+1)} = \mathbf{b}(\mathbf{x}^{(n)}), \qquad (6.59)$$

де

$$\mathbf{A}(\mathbf{x}^{(n)}) = \mathbf{W}(\mathbf{x}^{(n)}) \quad \text{і} \quad \mathbf{b}(\mathbf{x}^{(n)}) = \mathbf{W}(\mathbf{x}^{(n)})\mathbf{x}^{(n)} - \mathbf{F}(\mathbf{x}^{(n)}). \qquad (6.60)$$

Таким чином, розв'язання системи нелінійних рівнянь (6.54) за методом Ньютона зводиться до багаторазового розв'язання системи лінійних рівнянь (6.59) із перерахуванням значень елементів її матриці і вектора правої частини за результатами наближення за черговими ітераціями. У разі виконання умов збіжності метода Ньютона (див. підрозділ 6.3) швидкість збіжності ітерацій залишається квадратичною:

$$\|\mathbf{x}^{(n+1)} - \alpha\| \leqslant c\|\mathbf{x}^{(n)} - \alpha\|^2.$$

Вважаючи, що матриця $\mathbf{A}(\mathbf{x}^{(n)})$ є невиродженою, розв'язують ітераційно систему лінійних рівнянь (6.59), починаючи з заданого наближення $\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_m^{(0)})^T$, і знаходять вектори $x^{(n)}$, $n = 0, 1, 2, \dots$. Для того щоб отримати розв'язок із потрібною точністю, процес продовжують доти, поки два послідовних наближення будуть відрізнятися один від одного не більш ніж на задану похибку.

На практиці виникають труднощі під час обчислення матриці Якобі (6.58) на кожній ітерації, тому її елементи доцільно оновлювати лише через задану кількість ітерацій:

$$\mathbf{W}(\mathbf{x}^{(p)})(\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}) + \mathbf{F}(\mathbf{x}^{(n)}) = 0, \quad n = p, \dots, p + k.$$

Це зменшує обсяг обчислень, але приводить до збільшення кількості ітерацій, необхідних для досягнення заданої точності.

Звичайно для обчислення елементів матриці Якобі використовується різницева апроксимація похідних:

$$\frac{\partial f_i(\mathbf{x})}{\partial x_j} = \Delta_{ij}(\mathbf{x}, h) = \frac{f_i(\mathbf{x} + h_j e_j) - f_i(\mathbf{x})}{h},$$

де e_j — координатний вектор: $h_j = x_j^{(n+1)} - x_j^{(n)}$.

Застосування різницевої апроксимації сповільнює процес розв'язування, тому що збіжність процедури від квадратичної зміщується до нелінійної з $p = 1{,}618$, як у методі січних. У разі такої апроксимації початкова еквівалентна система лінійних рівнянь (6.53) перетвориться в систему рівнянь, що залежить від параметра h:

$$\mathbf{W}(\mathbf{x}^{(n)}, h)(\mathbf{x}^{(n+1)} - \mathbf{x}^{(n)}) + \mathbf{F}(\mathbf{x}^{(n)}) = 0. \tag{6.61}$$

Якщо ввести $h_j = f_j(\mathbf{x}^{(n)})$, то за аналогією з (6.38) можна отримати узагальнений метод Стефенсона. Коли системи нелінійних рівнянь мають дуже великі розмірності, можна також застосовувати нелінійний ітераційний метод Зейделя (див. розділ 3), який передбачає незалежне розв'язання кожного рівняння системи (6.47) відносно тільки однієї змінної

$$f_i\left(x_1^{(n+1)}, ..., x_{i-1}^{(n+1)}, x_i^{(n)}, x_{i+1}^{(n)}, ..., x_m^{(n)}\right) = 0. \tag{6.62}$$

Приклад 6.15

Розв'яжемо методом Ньютона систему нелінійних рівнянь

$$\begin{cases} f_1(x_1, x_2) = x_1^2 + x_1 x_2 - 10 = 0, \\ f_2(x_1, x_2) = x_2 + 3x_1 x_2^2 - 57 = 0 \end{cases}$$

за початкових умов $x_1^{(0)} = 1{,}5$, $x_2^{(0)} = 3{,}5$.

Скористаємося операторами пакета Mathematica для виконання основних проміжних обчислень. Визначимо систему нелінійних рівнянь і побудуємо матрицю Якобі:

```
In[]:= f[x1_, x2_]:= {x1^2 + x1*x2 - 10, x2 + 3*x1*x2^2 - 57};
       m1[x1_, x2_] = D[f[x1, x2], x1]
Out[]= {2 x1 + x2, 3 x2^2}
In[]:= m2[x1_, x2_] = D[f[x1, x2], x2]
Out[]= {x1, 1 + 6 x1 x2}
In[]:= J[x1_, x2_] = {m1[x1, x2], m2[x1, x2]}
Out[]= {{2x1 + x2, 3x2^2}, {x1, 1 + 6 x1 x2}}
```

Тобто як результат розрахунків отримаємо:

$$\frac{\partial f_1}{\partial x_1} = 2x_1 + x_2 = 2 \cdot 1{,}5 + 3{,}5 = 6{,}5; \quad \frac{\partial f_1}{\partial x_2} = x_1 = 1{,}5;$$

$$\frac{\partial f_2}{\partial x_1} = 3x_2^2 = 3 \cdot (3{,}5)^2 = 36{,}75; \quad \frac{\partial f_2}{\partial x_2} = 1 + 6x_1 x_2 = 1 + 6 \cdot 1{,}5 \cdot 3{,}5 = 32{,}5.$$

Задамо початкове наближення

```
In[]:= n = 0; x[1, n] = 1.5; x[2, n] = 3.5;
```

і оцінимо значення нев'язок на початковій ітерації:

```
In[]:= bp[n] = f[x[1, n], x[2, n]]
Out[]= {-2.5, 1.625}
```

або

$$f_1(0) = (1,5)^2 + 1,5 \cdot 3,5 - 10 = -2,5;$$
$$f_2(0) = 3,5 + 3 \cdot 1,5 \cdot (3,5)^2 - 57 = 1,625.$$

Обчислимо також матрицю Якобі для заданих початкових умов:

```
In:= B = Transpose[J[x[1, n], x[2, n]]]
Out[]= {{6.5, 1.5}, {36.75, 32.5}}
```

Побудуємо ітераційний процес:

```
In[]:= eps = 0.01; epk = 10;
       While[epk > eps, b[n+1] = B. {x[1, n], x[2, n]} - bp[n];
          {x[1, n+1], x[2, n+1]} = LinearSolve[B, b[n+1]];
          bp[n+1] = f[x[1, n+1], x[2, n+1]];
          epk = Max[Abs[x[1, n+1] - x[1, n]], Abs[x[2, n+1] - x[2, n]]]; n++;
          Print["n = ", n, "x1 = ", x[1, n], "x2 = ", x[2, n], "Δx = ", epk] ];
```

Для кращого розуміння самої процедури розв'язання системи нелінійних рівнянь запишемо в матрично-векторній формі результати виконання першої ітерації:

$$\begin{bmatrix} 6,5 & 1,5 \\ 36,75 & 32,5 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = -\begin{bmatrix} -2,5 \\ 1,625 \end{bmatrix} + \begin{bmatrix} 6,5 & 1,5 \\ 36,75 & 32,5 \end{bmatrix}\begin{bmatrix} 1,5 \\ 3,5 \end{bmatrix},$$

звідки

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 2,03603 \\ 2,84388 \end{bmatrix}.$$

Переконаємося, що на наступних ітераціях наближення сходяться до точного розв'язку $x_1 \to 2$, $x_2 \to 3$ дуже швидко:

```
n = 1   x1 = 2.03603   x2 = 2.84388   Δx = 0.656125
n = 2   x1 = 2.00373   x2 = 3.02674   Δx = 0.182864
n = 3   x1 = 1.99767   x2 = 2.99987   Δx = 0.0268733
n = 4   x1 = 2.00047   x2 = 2.99879   Δx = 0.00279836
```

Приклад 6.16

Знайдемо, використовуючи пакет Mathematica, розв'язок системи нелінійних рівнянь методом Ньютона:

$$\begin{cases} \operatorname{tg}(x_1 - x_2) - x_1 x_2 = 0, \\ 0,5x_1^2 + 2x_2^2 - 1 = 0. \end{cases}$$

Визначимо Якобіан вихідної матриці:

```
In[]:= Ja[x1_, x2_] = Jac[{f1[x1, x2], f2[x1, x2]}, {x1, x2}]
Out[]= {{-x2 + Sec[x1 - x2]^2, -x1 - Sec[x1 - x2]^2}, {1. x1, 4 x2}}

In[]:= Det[Ja[1, 0.5]]]
Out[]= 3.89534
```

Якобіан не дорівнює нулю, визначимо вектор-функцію. Спочатку знайдемо матрицю, обернену до матриці Якобі, і визначимо її у початковій точці:

```
In[]:= Jin = Inverse[Ja[1, 0.5]]]
```

Визначимо вектор функції $\mathbf{F(x)}$ і $\Psi(\mathbf{x})$:

```
In[]:= F[X_List]:= [f1[x1,x2], f2[x1,x2]} /.(x1 → X[[1]], x2 → X[[2]]};
       Z[X_]:= {X[[1]], X[[2]]} - Jin . F[X]
```

Розв'яжемо систему рівнянь, застосувавши модифікований метод Ньютона, тобто використаємо обернену матрицю Якобі для нульового наближення. Виконаємо ітерації відповідно до формули (6.59):

```
In[]:= NestList[Z, {1., 0.5}, 6]
Out=[] {{1, 0.5}, {0.976227, 0.511887}, {0.975304, 0.512066}, {0.97525, 0.512077},
        {0.975247, 0.512078}, {0.75247, 0.512078}, {0.975247, 0.512078}}
```

Розв'яжемо розглянутий приклад за допомогою стандартного оператора пакета Mathematica:

```
In[]:= FindRoot[{f1[x1,x2] == 0, {f2[x1,x2] == 0, {x1, 1.}, {x2, 0.5}]
Out[]= {x1 → 0.975247, x2 → 0.512078}
```

і переконаємося, що результати збігаються.

6.8. Розв'язання систем нелінійних рівнянь засобами пакета Mathematica

Для розв'язання нелінійних алгебраїчних рівнянь в пакеті Mathematica передбачено два оператори — FindRoot і NSolve. Перший з них реалізує метод Ньютона з коефіцієнтом демпфірування α_n (6.17, 6.18, 6.20). При цьому можна задати початкове значення x_0, а значення похідної (чи матриці Якобі для системи рівнянь) обчислити в символьному вигляді.

Розглянуте в прикладі 6.13 рівняння

$$f(x) = x + (e^{(x/0,25)} - 1) - 50 = 0$$

розв'язують за допомогою пакета Mathematica в такий спосіб:

```
In[]:= FindRoot[x + (Exp[x/0.25] - 1)- 50 == 0, {x,0}]
Out[]= {x → 0.978115}
```

Для доволі складних функцій, коли програма повідомляє про неможливість обчислення похідних у символьній формі, можна продовжити розв'язування одним із двох способів. По-перше, користувач може вказати в операторі FindRoot не тільки функції, але й їхні похідні та знову скористатися методом Ньютона:

```
In[]:= FindRoot[e^Abs[x] == 2, {x, 1}, Jacobian → {{Sign[x] e^Abs[x]}}]
Out[]= {x → 0.693148}
```

По-друге, користувач може задати дві початкові точки, переключивши в такий спосіб програму з методу Ньютона на метод січних:

```
In[]:= FindRoot[e^Abs[x] == 2, {x, 0, 1}]
Out[]= {x → 0.693147}
```

і

```
In[]:= FindRoot[x + (Exp[x/0.25] - 1) - 50 == 0, {x, 0, 1}]
Out[]= {x → 0.978115}
```

В операторі FindRoot можна задати кілька параметрів, які контролюють ефективність його дії: MaxIterations — для задавання кількості необхідних ітерацій (за замовчуванням цей параметр дорівнює 15); AccuracyGoal — для задавання глобальної похибки обчислень; WorkingPrecision — для задавання локальної похибки (параметр AccuracyGoal значно менший за WorkingPrecision); DampingFactor — задає значення коефіцієнта α_n.

За допомогою оператора FindRoot можна розв'язати всі рівняння, що розглядалися в цьому розділі. Наприклад, щоб розв'язати рівняння з прикладу (6.14)

$$f(x) = 3\operatorname{arctg}(3-x) = 0,$$

досить одного оператора

```
In[]:= FindRoot[3*ArcTan[3. - x], {x, 0}]
Out[]= {x → 3.}
```

У пакеті Mathematica передбачена зручна можливість переглянути проміжні значення на успішних ітераціях за допомогою оператора StepMonitor:

```
In[]:= FindRoot[3*ArcTan[3. - x], {x, 0}, StepMonitor :> Print[x]]
4.33141
1.76227
4.01885
2.39913
3.13553
2.99835
3.
3.
Out[]= {x → 3.}
```

Більш зручний вигляд проміжних значень забезпечується завдяки використанню ще двох операторів — Reap і Sow:

```
In[]:= Reap[FindRoot[3*ArcTan[3. - x], {x, 0}, StepMonitor :> Sow[x]]]
Out[]= {{x → 3.}, {{5.25424, 2.32969, 3.18554, 2.99577, 3., 3.}}}
```

Наведемо ще один приклад розв'язання рівняння $y(x) = 10\operatorname{arctg}(5-x) - 1 = 0$ (приклад 6.5) за допомогою вбудованих засобів пакета Mathematica.

```
In[]:= FindRoot[10 ArcTan[5 - x] - 1 == 0, {x, 3.5}]
Out[]= {x → 4.89967}
```

Знову, як і раніше, можна переглянути проміжні значення, але тепер на всіх виконаних ітераціях, а не тільки успішних, якщо скористатися оператором EvaluationMonitor:

```
In[]:= Reap[FindRoot[10 ArcTan[5 - x] - 1 == 0, {x, 3.5}, EvaluationMonitor :> Sow[x]]]
Out[]= {{x → 4.89967}, {{3.5, 6.36908, 4.7016, 4.90851, 4.89967, 4.89967, 4.89967}}}
```

У порівнянні з прикладом 6.5 певне зменшення кількості ітерацій пояснюється тим, що в пакеті Mathematica додатково до методу Ньютона з демпфіруванням застосовується також метод ітераційного покращення розв'язку на зразок (2.57), розглянутий у розділі 2.

Інший існуючий у пакеті Mathematica оператор NSolve призначений здебільшого для розв'язування алгебраїчних рівнянь. Він реалізує алгоритм Дженкінса–Трауба, що базується на методі Ньютона та співвідношеннях (6.28) і (6.30), і не потребує задавання початкового значення. Наведемо декілька прикладів його застосування:

$$x^{10} - x^9 - 3x^8 + 3x^7 + 2x^4 - x^3 - 7x^2 + 3x + 3 = 0,$$

```
In[]:= NSolve[3 + 3 x - 7 x^2 - x^3 + 2 x^4 + 3 x^7 - 3 x^8 - x^9 + x^10 == 0, x]
Out[]= {{x → -1.73205}, {x → -0.868688 - 0.585282i}, {x → -0.868688 + 0.585282i},
        {x → -0.496292}, {x → 0.0763556 - 1.14095i}, {x → 0.0763556 + 1.14095i},
        {x → 1.}, {x → 1.04048 - 0.56735i}, {x → 1.04048 + 0.56735i}, {x → 1.73205}}
```

Інше рівняння:

$$-(1+x) + 2\sqrt{1+x^2} - 3(1+x^3)^{1/3} + 5(1+x^5)^{1/5} - 4 = 0,$$

```
In[]:= NSolve[(-(1 + x))+ 2 Sqrt[1+x^2]-
            3 (1 + x^3)^(1/3) + 5 (1 + x^5)^(1/5) - 4 == 0, x]
Out[]= {{x → 1.54208}, {x → -0.999478},
        {x → 0.437296 + 0.838713i}},
```

Система:

$$\begin{cases} f_1(x_1, x_2) = x_1^2 + x_1 x_2 - 10 = 0, \\ f_2(x_1, x_2) = x_2 + 3x_1 x_2^2 - 57 = 0, \end{cases}$$

```
In[]:= P = NSolve[{f1 = x1^2 + x1*x2 - 10 == 0,
                   f2 = x2 + 3 x1*x2^2 - 57 == 0},
                   {x1, x2}]
Out[]= {{x1 → 4.39374, x2 → -2.11778},
        {x1 → -3.19687 + 1.24063i, x2 → 0.478245 - 2.29566i},
        {x1 → -3.19687 - 1.24063i, x2 → 0.478245 + 2.29566i},
        {x1 → 2., x2 → 3.}}
```

Серед чотирьох пар значень коренів розглянутої системи рівнянь є і комплексно-спряжені, тому що код, який реалізує оператор NSolve, самостійно вибирає та змінює початкові значення змінних x_1 та x_2, які можуть бути не лише дійсними числами. Слід відмітити, що розв'язок {x2 → 3., x1 → 2.} збігається з наближенням, яке отримане для тієї самої системи рівнянь у прикладі 6.15.

Про складність оператора NSolve свідчить обсяг програмного коду, що його реалізує, — кілька сотень сторінок.

Висновки

1. Під час математичного моделювання об'єктів і систем розв'язання нелінійних рівнянь лежить в основі процедури визначення їх сталих станів. Це одна з основних обов'язкових процедур комплексів моделювання та оптимізації, яка визначає загальний успіх моделювання.
2. Метод Ньютона, що побудований на основі лінійної локальної апроксимації нелінійної функції, є базовим під час розв'язання нелінійних рівнянь. Головна його перевага полягає в тому, що він забезпечує квадратичну збіжність ітерацій в околі точки розв'язку і для нього порівняно просто формується рекурентна формула наближень. Проте ефективність методу залежить від вибраного початкового значення; він накладає також обмеження на вид нелінійних функцій (монотонність і крутість зміни).
3. Швидкий спуск у зону розв'язку забезпечує метод пошуку кривої розв'язку за рахунок спеціальної процедури перевизначення початкового значення.
4. Поступове зміщення наближень до околу точки розв'язку забезпечує метод Давиденка (або метод продовження розв'язку за параметром) за рахунок багаторазового розв'язання задачі, що модифікується в процесі обчислення.
5. Метод дихотомії (поділу відрізка навпіл) не обмежує вид нелінійних функцій, але він має повільну (лінійну) збіжність.
6. Для реалізації методу Ньютона необхідно знати точне значення функції та її похідної у точці наближення. Якщо похідна обчислюється чисельно, метод Ньютона переходить у метод січних, збіжність якого залишається нелінійною, але трохи меншою, ніж для методу Ньютона.
7. У разі застосування методу Мюллера використовується параболічна локальна апроксимація нелінійної функції, завдяки чому прискорюється розв'язання, проте потрібне виконання великого обсягу обчислень. Тим часом цей метод забезпечує обчислення комплексно-спряжених коренів за дійсних початкових умов на відміну від методу Ньютона, що вимагає в таких випадках задавання комплексно-спряжених початкових значень.
8. Розв'язання системи нелінійних рівнянь методом Ньютона зводиться до багаторазового розв'язання лінійної системи рівнянь, матриця коефіцієнтів і вектор правої частини якої уточнюються на кожній ітерації. Розв'язання лінійної системи здійснюється методами, розглянутими в розділах 2 і 3.
9. Якщо матриця коефіцієнтів сформованої лінійної системи рівнянь оновлюється не на кожній ітерації, загальний обсяг обчислень зменшується, але одночасно росте кількість ітерацій.
10. У програмах математичного моделювання рекомендується застосовувати різні комбінації розглянутих вище методів, при цьому потрібні методи, що визначаються користувачем, або вибираються під час розв'язання відповідно до деяких формалізованих критеріїв.

Контрольні запитання та завдання

1. Використовуючи метод Ньютона, знайдіть значення $\sqrt[7]{3}$.
2. Деякі комп'ютери не мають спеціального пристрою для ділення чисел. Щоб обчислити $1/b$, $b > 0$ формується наближення оберненого числа, яке потім використовують як нульову ітерацію. Нехай $f(x) = (1/x) - b$. Показати, що формула методу Ньютона для розв'язання цього рівняння має такий вигляд:

$$x_{k+1} = x_k(2 - bx_k).$$

3. За допомогою цієї формули обчисліть $1/b$ для $b = 1, 2, ..., 10$, вважаючи, що $x_0 = 0,01$.
4. Використовуючи ту ж функцію $f(x)$, повторіть обчислення методом дихотомії. Виберіть початковий інтервал $[0,001, 2]$.
5. Розв'яжіть нелінійне рівняння $(x+1)e^{x-1} - 1 = 0$ методом Ньютона.
6. Розв'яжіть нелінійне рівняння $e^x - x^2 - 2x - 2 = 0$, використовуючи метод Ньютона за початкового значення $x^{(0)} = 2,0$.
7. Розв'яжіть нелінійне рівняння $1 - x - e^{-2x} = 0$ методом Ньютона. Доведіть, що рівняння має два дійсних корені.
8. Розв'яжіть нелінійне рівняння $e^{-2x} - 1 + x = 0$ методом дихотомії. Знайдіть інтервал ізоляції кореня.
9. Знайдіть 10 найменших додатних значень x, для яких пряма $y = x$ перетинає графік $y = \operatorname{tg} x$. Розв'язок цієї задачі використовується у разі визначення максимального навантаження, що може нести стрижень без зміни форми.
10. Знайдіть дійсні корені рівняння $x^4 - 3x^2 + 75x - 10000 = 0$.
11. Застосуйте метод Ньютона до рівняння $f(x) = (x-1)^2$, що має корінь кратності 2.
12. Рівняння Кеплера для обчислення орбіти має вигляд $M = E - e\sin E$. Символи M, E і e позначають середню аномалію, ексцентричну аномалію та ексцентриситет орбіти. Знайдіть E, якщо $M = 24,85109$ і $e = 0,1$.
13. Розв'яжіть систему рівнянь:

$$\begin{cases} (1 - \sin x_1)^2 + (1,5 - \sin x_2)^2 + (\cos x_1 - \cos x_2)^2 - (x_3 + 0,25)^2 = 0, \\ (x_3 + 0,25)\sin x_1 + 2x_3(\cos x_2 \sin x_1 - \cos x_1) = 0, \\ (x_3 + 0,25)\sin x_2 + 2x_3(\cos x_1 \sin x_2 - 1.5\cos x_2) = 0 \end{cases}$$

за умови, що $x_1 \geqslant 0$, $x_2 \leqslant 90°$, $x_3 > 0$.

Розділ 7

Чисельне диференціювання та інтегрування функцій

- Формули чисельного диференціювання на основі інтерполяційних поліномів
- Похибки формул чисельного диференціювання
- Квадратурні формули Ньютона–Котеса
- Оцінка похибки інтегрування
- Квадратурні формули Гаусса

Даний розділ присвячений питанням обчислення похідних заданих порядків та інтегрування функцій. Якщо аналітичне зображення вихідної функції невідоме або доволі складне, похідні будь-якого порядку можуть бути обчислені на основі наведених у розділі формул диференціювання. Більшість формул чисельного диференціювання можуть бути отримані на основі інтерполяційних поліномів. Для цього достатньо замінити початкову функцію її інтерполяційнім поліномом, а потім обчислити похідні від нього [4, 11]. Якщо поліном із достатньою точністю наближає вихідну функцію, то можна стверджувати, що похідні будь-якого порядку від полінома мало відрізнятимуться від похідних функції.

Задача чисельного інтегрування тісно пов'язана із задачею розв'язання диференціальних рівнянь [21]. Оскільки для розв'язання більшості практичних задач треба не тільки знаходити значення інтеграла, а ще й оцінювали похибку, в розділі розглядаються способи обчислення похибки за залишковим членом, за правилом Рунге та із застосуванням екстраполяції за Річардсоном.

7.1. Чисельне диференціювання функцій

Задача чисельного диференціювання полягає у знаходженні значень похідних функції $y = f(x)$ у заданих точках у випадку, коли аналітичний запис функції $f(x)$ невідомий або дуже складний чи функція задана таблично. Привабливість чисельного підходу здебільшого пояснюється наявністю простих залежностей, за допомогою яких похідні в заданих точках можна апроксимувати декількома значеннями функції в цих і близьких до них точках.

Конструювання формул наближеного диференціювання полягає в тому, що функцію $f(x)$ на заданому відрізку $[a, b]$ замінюють відповідною апроксимуючою

функцією $\varphi(x)$, а потім вважають, що похідні від функцій $f(x)$ і $\varphi(x)$ збігаються, наприклад:

$$f'(x) \approx \varphi'(x),$$

де

$$a \leqslant x \leqslant b.$$

Аналогічно знаходять похідні вищих порядків від функції $f(x)$. При цьому апроксимуюча функція $\varphi(x)$ найчастіше задається у вигляді полінома.

Природно, що тоді похідні $f(x)$ обчислюються з деякою похибкою. Якщо через $R(x)$ позначити залишковий член

$$R(x) = f(x) - \varphi(x), \tag{7.1}$$

то похибку $R'(x)$ можна записати як

$$R'(x) = f'(x) - \varphi'(x). \tag{7.2}$$

Такі ж формули диференціювання можна побудувати і для знаходження похідних вищих порядків, тобто

$$\phi^{(k)}(x) = f^{(k)}(x) - R^{(k)}(x).$$

Слід зазначити, що малість залишкового члена $R(x)$ не свідчить про малість залишкових членів похідних, бо похідні від малих функцій можуть бути досить великими. У загальному випадку наближене диференціювання є менш точною операцією, ніж інтерполяція, тому що для як завгодно близьких функцій $f(x)$ і $\varphi(x)$ різниця між їх похідними може бути як завгодно великою. Наявність великої похибки під час обчислення значень похідної пояснюється ще й тим, що значення функцій, які входять до формул диференціювання, здебільшого мають деяку похибку. Однак, застосовуючи виважений підхід до вибору інтерполяційної формули та її порядку, можна отримати результат із необхідною точністю.

7.1.2. Формули чисельного диференціювання на основі першої інтерполяційної формули Ньютона

Нехай функція $f(x)$ задана в рівновіддалених точках $x_i = a + ih$, $i = 0, 1, ..., n$ відрізка $[a, b]$ значеннями $f_i = f(x_i)$.

Щоб обчислити похідні $f'(x)$, $f''(x)$ і т. д., замінимо $f(x)$ інтерполяційним поліномом Ньютона (5.22) для інтерполяції вперед, тобто

$$f(x) = f_0 + t\nabla f_0 + \frac{t(t-1)}{2!}\nabla^2 f_0 + \frac{t(t-1)(t-2)}{3!}\nabla^3 f_0 + ... + \frac{t(t-1)...(t-n+1)}{n!}\nabla^n f_0,$$

де

$$t = \frac{x - x_0}{h}.$$

Для отримання скінченно-різницевої формули чисельного диференціювання візьмемо похідні від обох частин тотожності, скориставшись співвідношеннями

$$f'(x) = \frac{df}{dx} = \frac{df}{dt}\frac{dt}{dx} = \frac{1}{h}\frac{df}{dt},$$

тому що

$$\frac{dt}{dx} = \frac{1}{h}.$$

Тоді

$$f'(x) = \frac{1}{h}\left[\nabla f_0 + \frac{2t-1}{2!}\nabla^2 f_0 + \frac{3t^2-6t+2}{3!}\nabla^3 f_0 + \frac{4t^3-18t^2+22t-6}{4!}\nabla^4 f_0 + \ldots\right]. \quad (7.3)$$

Для обчислення другої похідної використаємо співвідношення

$$f''(x) = \frac{d(f'(x))}{dx} = \frac{d(f'(x))}{dt}\frac{dt}{dx} = \frac{1}{h}\frac{d(f'(x))}{dt}.$$

Тоді формула чисельного диференціювання матиме такий вигляд:

$$f''(x) = \frac{1}{h^2}\left[\nabla^2 f_0 + (t-1)\nabla^3 f_0 + \frac{6t^2-18t+11}{12}\nabla^4 f_0 + \ldots\right]. \quad (7.4)$$

У разі потреби на основі різних інтерполяційних формул, розглянутих у розділі 5, можна обчислити похідні функції будь-якого порядку.

Користуючись виразом (7.3), можна отримати формули чисельного диференціювання для інтерполяційних поліномів різних степенів. Для $n = 1$ маємо формулу чисельного диференціювання на основі лінійного інтерполяційного полінома:

$$f'(x) = \frac{\nabla f_0}{h} \quad \text{для} \quad x \in [x_0; x_1]. \quad (7.5)$$

Для $n = 2$ маємо формулу чисельного диференціювання на основі квадратичного полінома

$$f'(x) = \frac{1}{h}\left[\nabla f_0 + \frac{2t-1}{2!}\nabla^2 f_0\right] \quad \text{для} \quad x \in [x_0; x_2]. \quad (7.6)$$

Формулу на основі кубічної інтерполяції отримуємо, приймаючи, що $n = 3$:

$$f'(x) = \frac{1}{h}\left[\nabla f_0 + \frac{2t-1}{2!}\nabla^2 f_0 + \frac{3t^2-6t+2}{3!}\nabla^3 f_0\right] \quad \text{для} \quad x \in [x_0; x_3] \quad (7.7)$$

і т. д.

Слід відмітити, що в разі обчислення похідних за формулами (7.5)–(7.7) у фіксованій точці x_i як x_0 варто вибирати найближче табличне значення аргументу.

Інколи виникає потреба обчислити значення похідної від функції $f(x)$ безпосередньо у вузлах інтерполяції, тобто для $t = 0$. У цьому випадку формули чисельного диференціювання спрощуються, наприклад з виразу (7.3) маємо:

$$f'(x) = \frac{1}{h}\left(\nabla f_0 - \frac{\nabla^2 f_0}{2} + \frac{\nabla^3 f_0}{3} - \frac{\nabla^4 f_0}{4} + \ldots\right). \tag{7.8}$$

Формула ще більш спроститься, якщо мова буде йти про окремі випадки інтерполяції – лінійної, квадратичної, кубічної (7.5)–(7.7). Необхідні скінченні різниці обчислюються за схемою побудови скінченних різниць:

$$\begin{array}{lllll} f_0 & & & & \\ & \nabla f_0 & & & \\ f_1 & & \nabla^2 f_0 & & \\ & \nabla f_1 & & \nabla^3 f_0 & \\ f_2 & & \nabla^2 f_1 & & \nabla^4 f_0 \cdot \\ & \nabla f_2 & & \nabla^3 f_1 & \\ f_3 & & \nabla^2 f_2 & & \\ & \nabla f_3 & & & \\ f_4 & & & & \end{array}$$

7.1.3. Формули чисельного диференціювання на основі інтерполяційного полінома Лагранжа

Часом у формулах чисельного диференціювання зручніше застосувати не скінченні різниці функції, а її значення. Для отримання таких формул використовують поліном Лагранжа. Розглянемо випадок, коли функція задана таблично в рівновіддалених точках $x_0, x_1, x_2, \ldots, x_n$, тобто відомі $x_i = x_0 + ih$ $(i = 0, 1, \ldots, n)$ і значення $f(x_i)$, $i = 0, 1, \ldots, n$. Для заданої системи вузлів x_i побудуємо інтерполяційний поліном Лагранжа (5.8) у вигляді:

$$L_n(x) = \sum_{i=0}^{n} \frac{\omega_{n+1}(x) f(x_i)}{(x - x_i)\omega'_{n+1}(x_i)}, \tag{7.9}$$

де $\omega_{n+1}(x) = (x - x_0)(x - x_1)\ldots(x - x_n)$.

Скористаємося заміною:

$$t = \frac{x - x_0}{h},$$

тоді отримаємо

$$\omega_{n+1}(x) = (x - x_0)(x - x_1)\ldots(x - x_n) = h^{n+1} t(t-1)\ldots(t-n) = h^{n+1} t^{[n+1]}, \tag{7.10}$$

де $t^{[n+1]} = t(t-1)\ldots(t-n)$.

Многочлен $\omega'_{n+1}(x_i)$ перетворимо в такий спосіб:

$$\omega'_{n+1}(x_i) = (x_i - x_0)(x_i - x_1)...(x_i - x_{i-1})(x_i - x_{i+1})...(x_i - x_n) = \\ = h^n i(i-1)...1(-1)...[-(n-i)] = (-1)^{n-i} h^n i!(n-i)! \tag{7.11}$$

Тобто для випадку рівновіддалених вузлів поліном Лагранжа можемо записати в такому вигляді:

$$L_n(x) = \sum_{i=0}^{n} \frac{(-1)^{n-i} y_i}{i!\,(n-1)!} \frac{t^{[n+1]}}{t-i}. \tag{7.12}$$

Беручі до уваги, що $dx/dt = h$, отримаємо

$$f'(x) = L'(x) = \frac{1}{h} \sum_{i=0}^{n} \frac{(-1)^{n-i} f(x_i)}{i!\,(n-i)!} \frac{d}{dt}\left(\frac{t^{[n+1]}}{t-i}\right). \tag{7.13}$$

Похідні вищих порядків знаходять аналогічно.

У випадку довільного розташування вузлів іноді використовують метод невизначених коефіцієнтів. Цей метод дозволяє уникнути громіздких виразів, поява яких пов'язана з перетвореннями полінома Лагранжа, і обчислити значення похідних будь-якого порядку у вузлах інтерполяції. Для цього шукану формулу записують у вигляді:

$$f^{(k)}(x_i) = \sum_{i=0}^{n} c_i f(x_i) + R(x) \tag{7.14}$$

і підбирають коефіцієнти з умови $R(x) = 0$, якщо $f(x_i) = 1, x_i, x_i^2, ..., x_i^n$. Тоді одержують систему для обчислення невідомих коефіцієнтів c_i:

$$\begin{gathered} c_0 + c_1 + ... + c_n = 0, \\ c_0 x_0 + c_1 x_1 + ... + c_n x_n = 0, \\ c_0 x_0^{k-1} + c_1 x_1^{k-1} + ... + c_n x_n^{k-1} = 0, \\ c_0 x + c_1 x_1^k + ... + c_n x_n^k = k!, \\ c_0 x + c_1 x_1^{k+1} + ... + c_n x_n^{k+1} = (k+1)!\, x_i, \\ c_0 x + c_1 x_1^n + ... + c_n x_n^n = n(n-1)...(n-k+1) x_i^{n-k}. \end{gathered} \tag{7.15}$$

Методи розв'язання таких систем докладно розглянуті в розділі 5.

7.1.4. Залишкові члени формул чисельного диференціювання

Для виведення формул залишкових членів вважатимемо, що функція $f(x)$ достатньо гладка. Нехай вираз (7.1) визначає похибку, яка виникає через заміну функції $f(x)$ інтерполяційним поліномом Ньютона (5.22), а вираз (7.2) —

похибку похідної. Скористаємося отриманою в розділі 5 оцінкою залишкового члена (5.13):

$$R(x) = h^{k+1}\frac{t(t-1)...(t-k)}{(k+1)!}f^{(k+1)}(\xi),$$

де $x_0 < \xi < x_k$.

Припускаючи, що функція $f(x)$ неперервно диференційована $k+2$ разів, отримаємо:

$$R'(x) =$$

$$= \frac{h^k}{(k+1)!}\left[f^{(k+1)}(\xi)\frac{d}{dt}[t(t-1)...(t-k)] + t(t-1)...(t-n)\frac{d}{dt}[f^{(k+1)}(\xi)]\right]. \quad (7.16)$$

Якщо похідну необхідно обчислити у вузлі інтерполяції, то, як було зазначено раніше, можна прийняти, що $x = x_0$ і, отже, $t = 0$. У цьому випадку вираз для залишкового члена спрощується:

$$R'(x = x_0) = (-1)^k\frac{h^k}{k+1}f^{(k+1)}(\xi). \quad (7.17)$$

Оцінки (7.16)–(7.17) є апріорними і залежать від значення $f^{(k+1)}(\xi)$, яке в загальному випадку невідоме. За малих h ці значення наближено можуть бути замінені скінченними різницями:

$$f^{(k+1)}(\xi) \approx \frac{\nabla^{k+1}f_0}{h^{k+1}}.$$

Тоді залишковий член матиме такий вигляд:

$$R'(x_0) \approx \frac{(-1)^k\nabla^{k+1}f_0}{h(k+1)}. \quad (7.18)$$

Аналогічно знаходять залишкові члени для похідних вищих порядків.

Оскільки формула Ньютона стає такою ж, як і формула Лагранжа, коли в розрахунку присутні всі вузли інтерполяції, то в разі чисельного диференціювання за формулою Лагранжа для оцінки залишкового члена можуть бути використані всі вирази (7.16)–(7.18), якщо в них k замінити на n.

Можна отримати й оцінку залишкового члена, що враховує специфіку формули Лагранжа. Для оцінки похибки

$$R_n'(x) = f'(x) - L_n'(x)$$

скористаємося формулою похибки інтерполяційного полінома Лагранжа (5.13):

$$R_n(x) = f(x) - L_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\omega_{n+1}(x).$$

Враховуючи неперервну диференційованість $f(x)$ до $(n+2)$-го порядку включно, знаходимо:

$$R_n'(x) = \frac{1}{(n+1)!}\left[f^{(n+1)}(\xi)\omega_{n+1}'(x) + \omega_{n+1}(x)\frac{d}{dx}\left[f^{(n+1)}(\xi)\right]\right]. \qquad (7.19)$$

Звідси, враховуючи формулу (7.11), отримуємо похибку похідної у вузлах:

$$R_n'(x) = (-1)^{n-i}h^n\frac{i!(n-i)!}{(n+1)!}f^{(n+1)}(\xi). \qquad (7.20)$$

Аналогічні оцінки для похибки можна отримати й у разі використання інших інтерполяційних формул, наведених у розділі 5. Існують також інші вирази формул чисельного диференціювання, які відмінні від (7.16)–(7.20). Проте такі вирази малопридатні для практичного оцінювання похибок, оскільки в них використовуються похідні більш високих порядків, ніж ті, що обчислюються. Однак слід відзначити їх важливість для якісного порівняння різних апроксимацій похідних. Цими формулами можна скористатися і для оцінювання точності результатів чисельного диференціювання, заміняючи значення $\sup_{[a,b]}(f^{(n+1)}(\xi))$ скінченними різницями $\max|\nabla^{n+1}f(x_i)|$, як це зроблено у (7.18).

7.1.5. Формули диференціювання для практичних обчислень

На основі виразів (7.5)–(7.7) та (7.14) можна записати співвідношення, які на практиці застосовуються для апроксимації похідних перших трьох порядків від функцій, заданих таблично (табл. 7.1–7.3).

Таблиця 7.1. Формули для оцінок перших похідних

Тип формули	Формула	Порядок точності
Несиметричні обернені, чи формули диференціювання назад	$f'(x_i) = \frac{f(x_i) - f(x_{i-1})}{h}$	$0(h)$
	$f'(x_i) = \frac{3f(x_i) - 4f(x_{i-1}) + f(x_{i-2})}{2h}$	$0(h^2)$
Несиметричні прямі, чи формули диференціювання уперед	$f'(x_i) = \frac{f(x_{i+1}) - f(x_i)}{h}$	$0(h)$
	$f'(x_i) = \frac{-f(x_{i+2}) + 4f(x_{i+1}) - 3f(x_i)}{2h}$	$0(h^2)$
Симетричні	$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{2h}$	$0(h^2)$
	$f'(x_i) = \frac{-f(x_{i+2}) + 8f(x_{i+1}) - 8f(x_{i-1}) + f(x_{i-2})}{12h}$	$0(h^4)$

Таблиця 7.2. Формули для оцінок других похідних

Тип формули	Формула	Порядок точності
Несиметричні обернені	$f''(x_i) = \frac{f(x_i) - 2f(x_{i-1}) + f(x_{i-2})}{h^2}$	$0(h)$
	$f''(x_i) = \frac{2f(x_i) - 5f(x_{i-1}) + 4f(x_{i-2}) - f(x_{i-3})}{h^2}$	$0(h^2)$
Несиметричні прямі	$f''(x_i) = \frac{f(x_{i+2}) - 2f(x_{i+1}) + f(x_i)}{h^2}$	$0(h)$
	$f''(x_i) = \frac{f(x_{i+3}) + 4f(x_{i+2}) - 5f(x_{i+1}) + f(x_i)}{h^2}$	$0(h^2)$
Симетричні	$f''(x_i) = \frac{f(x_{i+1}) - 2f(x_i) + f(x_{i-1})}{h^2}$	$0(h^2)$
	$f''(x_i) = \frac{-f(x_{i+2}) + 16f(x_{i+1}) - 30f(x_i) + 16f(x_{i-1}) - f(x_{i-2})}{12h^2}$	$0(h^4)$

Таблиця 7.3. Формули для оцінок третіх похідних

Тип формули	Формула	Порядок точності
Несиметричні обернені	$f'''(x_i) = \frac{f(x_i) - 3f(x_{i-1}) + 3f(x_{i-2}) - f(x_i)}{h^3}$	$0(h)$
	$f''' = \frac{5f(x_i) - 18f(x_{i-1}) + 24f(x_{i-2}) - 14f(x_{i-3}) + 3f(x_{i-4})}{2h^3}$	$0(h^2)$
Несиметричні прямі	$f''(x_i) = \frac{f(x_{i+3}) - 3f(x_{i+2}) + 3f(x_{i+1}) - f(x_i)}{h^2}$	$0(h)$
	$f'''(x_i) = \frac{-3f(x_{i+4}) + 14f(x_{i+3}) - 24f(x_{i+2}) + 18f(x_{i+1}) - 5f(x_i)}{2h^2}$	$0(h^2)$
Симетричні	$f'''(x_i) = \frac{f(x_{i+2}) - 2f(x_{i+1}) + 2f(x_{i-1}) - f(x_{i-2})}{2h^3}$	$0(h^2)$
	$f'''(x_i) = \frac{f(x_{i+3}) + 8f(x_{i+2}) - 13f(x_{i+1}) +}{8h^3}$ $\frac{+13f(x_{i-1}) - 8f(x_{i-2}) - f(x_{i-3})}{8h^3}$	$0(h^4)$

Різниця між окремими формулами найбільш чітко проявляється у разі їх практичного застосування для оцінювання значень похідних, що показано у наступному прикладі.

Приклад 7.1

Оцінимо значення першої похідної від полінома

$$f(x) = -0,2x^4 - 0,2x^3 - 0,4x^2 - 0,3x + 1,5$$

у точці $x = 0,5$, використовуючи формули, наведені в табл. 7.1–7.3, із кроком $h = 0,25$.

Скористаємося пакетом Mathematica для визначення функції та її похідної:

```
In[]:= f[x_]:= -0.2x^4 - 0.2x^3 - 0.4x^2 - 0.3x + 1.5;
       f1[x_] = D[f[x], x];
```

Точне її значення легко обчислюється аналітично і дорівнює

```
In[]:= f1[0.5]
Out[]= -0.95,
```

Це значення дозволяє оцінити й порівняти похибки окремих формул диференціювання. Для цього спочатку обчислимо необхідні значення функції у фіксованих вузлах:

```
In[]:= TA = Table[{t, f[t]}, {t, 0, 1, .25}]
Out[]= {{0, 1.5}, {0.25, 1.39609}, {0.5, 1.2125}, {0.75, 0.902344}, {1., 0.4}}
```

і побудуємо таблицю значень (табл. 7.4).

Таблиця 7.4. Значення функції та її аргументу

Значення аргументу	Значення функції
$x_{i-2} = 0$	$f(x_{i-2}) = 1,5$
$x_{i-1} = 0,25$	$f(x_{i-1}) = 1,39609$
$x_i = 0,5$	$f(x_i) = 1,2125$
$x_{i+1} = 0,75$	$f(x_{i+1}) = 0,902344$
$x_{i+2} = 1$	$f(x_{i+2}) = 0,4$

Виконаємо обчислення згідно з несиметричними оберненими формулами, наведеними в табл. 7.1:

$$f'(0,5) = \frac{1,2125 - 1,39609}{0,25} = -0,734375, \quad \varepsilon = 22,6974\ \%;$$

$$f'(0,5) = \frac{3(1,2125) - 4(1,39609) + 1,5}{2(0,25)} = -0,89375, \quad \varepsilon = 5,92105\ \%.$$

Використання несиметричних прямих формул дозволяє отримати такі результати:

$$f'(0,5) = \frac{0,902344 - 1,2125}{0,25} = -1,240625, \quad \varepsilon = -30,5921\ \%;$$

$$f'(0,5) = \frac{-0,4 + 4(0,902344) - 3(1,2125)}{2(0,25)} = -0,85625, \quad \varepsilon = 9,8684\ \%.$$

Нарешті, застосовуючи симетричні формули, знаходимо:

$$f'(0,5) = \frac{0,902344 - 1,39609}{0,5} = -0,9875, \quad \varepsilon = -3,9474\ \%;$$

$$f'(0,5) = \frac{-0,4 + 8(0,902344) - 8(1,39609) + 1,5}{12(0,5)} = -0,95, \quad \varepsilon = 1,17 \cdot 10^{-14}\ \%.$$

Як видно з цього прикладу, найвищу точність забезпечують симетричні формули чисельного диференціювання. Крім того, за формулами, у яких порядок точності більше порядку апроксимуючого полінома, чи дорівнює йому, оцінка першої похідної обчислюється без похибки.

Для покращення оцінки похідних можна застосувати екстраполяцію Річардсона, яка була розглянута в розділі 1. Нагадаємо, що для формул із похибками обчислень $O(h^p)$ уточнений результат визначається виразом

$$D = D(h_2) + \frac{1}{(h_1/h_2)^p - 1}[D(h_2) - D(h_1)], \tag{7.21}$$

де $D(h_2)$ і $D(h_1)$ — оцінки похідних, обчислені з кроками h_2 і h_1. Наприклад, якщо $h_1 = 2h_2$, для симетричних формул із похибкою $O(h^2)$ уточнення за формулою (7.21) підвищує порядок точності до $O(h^4)$. І тоді формула має такий вигляд:

$$D = \frac{4}{3}D(h_2) - \frac{1}{3}D(h_1). \tag{7.22}$$

Для несиметричних формул із похибкою $O(h)$ вираз (7.21) спрощується:

$$D = D(h_2) + [D(h_2) - D(h_1)] \tag{7.23}$$

і дозволяє підвищити порядок точності до $O(h^2)$.

Приклад 7.2

Покажемо, як за допомогою екстраполяції Річардсона уточнити значення першої похідної, що були обчислені в прикладі 7.1 для полінома четвертого порядку.

Почнемо із симетричної формули, скориставшись таблицею TA, що містить значення функції та її аргументу:

```
In[]:= h = 0.5;  g1 = (TA[[5,2]] - TA[[1,2]])/(2h)
Out[]= -1.1
In[]:= h = h/2;  g2 = (TA[[4,2]] - TA[[2,2]])/(2h)
Out[]= -0.9875
```

На основі формули (7.22) знайдемо уточнене значення першої похідної й обчислимо похибку:

```
In[]:= g = 4/3*g2 - 1/3*g1
Out[]= -0.95
In[]:= Eps = Abs[(g - f1[0.5])/f1[0.5]]
Out[]= 2.33731*10^-16
```

Слід відмітити, що процедура чисельного диференціювання належить до погано обумовлених процедур, оскільки малі похибки окремих значень функції спричинюють істотні зміни скінченних різниць високого порядку, які використовуються в оцінках похідних. Останнє ілюструється прикладом побудови скінченних різниць для наступного вектора **y** = (0, 0, 0, 0, 1, 0, 0, 0, 0):

```
In[]:= y = {0, 0, 0, 0, 1, 0, 0, 0, 0};  n = Length[y];
    Do [A[1, j] = y[[j]], {j, n}];
```

```
Do [Do [A[i,j] = A[i-1, j+1] -A[i-1, j], {j, 1, n-i+1}], {i, 2, n}];
    TableForm[Table[A[j, i], {i, n}, {j, n-i+1}],
    TableDirections → {Column, Row}, TableAlignments → Center]
0  0  0  0  1  -5  15 -35 70
0  0  0  1 -4  10 -20  35
0  0  1 -3  6 -10  15
0  1 -2  3 -4   5
1 -1  1 -1  1
0  0  0  0
0  0  0
0  0
0
```

Тому табличні значення скінченних різниць слід попередньо згладжувати за допомогою інтерполяційного полінома.

Формули чисельного диференціювання застосовують дуже широко. Наприклад, симетричні формули для оцінок других похідних використовують під час розв'язання крайових задач для звичайних диференціальних рівнянь (розділ 11) і диференціальних рівнянь із частинними похідними (розділи 12–14). Несиметричні обернені формули застосовують для апроксимації перших похідних у граничних умовах для крайових задач (розділ 11) і розв'язання жорстких звичайних диференціальних рівнянь із початковими умовами (розділ 10) і т. д.

7.2. Чисельне інтегрування функцій

Розв'язати задачу інтегрування означає обчислити інтеграл Рімана для деякої функції $f(x)$ на заданому інтервалі $[a, b]$:

$$J = \int_a^b f(x)dx. \tag{7.24}$$

Якщо функція $f(x)$ неперервна на інтервалі $[a, b]$ і відома її початкова функція $F(x)$, то можна аналітично знайти інтеграл J за формулою Ньютона–Лейбніца:

$$J = F(b) - F(a).$$

Однак у багатьох випадках початкову функцію $F(x)$ не можна знайти аналітично чи $f(x)$ є занадто складною, що утруднює обчислення інтеграла за формулою Ньютона–Лейбніца, або воно взагалі стає неможливим. Часто на практиці підінтегральна функція $f(x)$ задається таблично, що також унеможливлює використання аналітичних методів.

У всіх згаданих випадках для обчислення інтеграла використовують чисельні методи. Традиційний підхід полягає в тому, що функцію $f(x)$ на відрізку $[a, b]$ заміняють інтерполяційною функцією $\varphi(x)$, наприклад поліномом Лагранжа або Ньютона, а потім приймають

$$\int_a^b f(x)dx = \int_a^b \varphi(x)dx + R(x), \tag{7.25}$$

де $R(x)$ – деяка похибка формули інтегрування.

У цьому випадку функція $\varphi(x)$ має бути такою, щоб інтеграл можна було обчислити безпосередньо. Якщо функція $f(x)$ задана аналітично, то наближено обчислити визначений інтеграл (7.24) можна заміною інтеграла скінченною сумою. При цьому відрізок інтегрування $[a, b]$ розбивається на n однакових частин з кроком

$$h = \frac{(b-a)}{n}.$$

У вузлах $x_i = a + ih,\ i = 0, 1, ..., n$ знаходяться значення підінтегральної функції $f(x_i),\ i = 0, 1, 2, ..., n,$ і шукана площа (значення інтеграла) обчислюється як

$$J \approx \sum_{i=0}^{n} c_i f(x_i), \tag{7.26}$$

де c_i — задані числові коефіцієнти.
Наближена рівність

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n} c_i f(x_i) + R(x) \tag{7.27}$$

називається квадратурною формулою, а c_i – коефіцієнтами квадратурної формули.

7.2.1. Формули прямокутників

Найпростіший підхід до обчислення значення інтеграла полягає у заміні площі криволінійної трапеції, обмеженої графіком функції, сумою площ прямокутників, тобто функція $f(x)$ апроксимується поліномом нульового степеня.

Для побудови формули чисельного інтегрування на всьому відрізку $[a, b]$ спочатку необхідно побудувати квадратурну формулу для інтеграла на частковому відрізку, а потім скористатися властивістю інтеграла

$$\int_a^b f(x)dx = \sum_{i=1}^{n} \int_{x_{i-1}}^{x_i} f(x)dx.$$

Спочатку розглянемо метод *центральних прямокутників*. Замінимо значення інтеграла на частковому відрізку $[x_0, x_1]$ площею прямокутника:

$$\int_{x_0}^{x_1} f(x)dx = (x_1 - x_0) f(\overline{x}) + R_1(f),$$

де $\overline{x} = (x_1 + x_0)/2$ — центральна точка відрізка.

Отримана формула називається формулою центральних прямокутників на частковому відрізку. Це означає, що площа криволінійної трапеції заміняється

площею прямокутника (рис. 7.1). Тоді похибку на частковому відрізку визначимо як

$$R_1(f) = \int_{x_0}^{x_1} f(x)dx - (x_1 - x_0)f(\overline{x}).$$

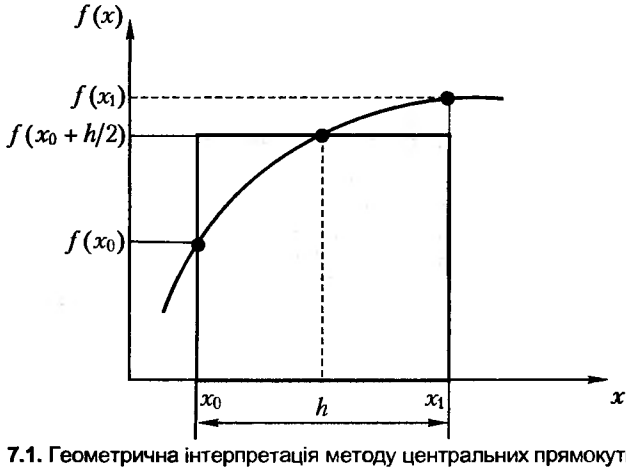


Рис. 7.1. Геометрична інтерпретація методу центральних прямокутників на частковому відрізку

Припустимо, що функція $f(x)$ неперервна разом зі своїми похідними до другого порядку включно. Розкладемо функцію $f(x)$, $x \in [x_0, x_1]$ в околі точки $\overline{x}$ у ряд Тейлора.

Проінтегруємо останній вираз на інтервалі $[x_0, x_1]$:

$$\int_{x_0}^{x_1} f(x)\,dx = \int_{x_0}^{x_1} f(\overline{x})\,dx + \int_{x_0}^{x_1} f'(\overline{x})(x-\overline{x})\,dx + \int_{x_0}^{x_1} f''(\xi)\frac{(x-\overline{x})^2}{2!}dx =$$

$$= f(\overline{x})(x_1 - x_0) + f'(\overline{x})\underbrace{\left[\frac{x^2}{2}\Big|_{x_0}^{x_1} - x\,\overline{x}\Big|_{x_0}^{x_1}\right]}_{=0} + f''(\xi)\int_{x_0}^{x_1}\frac{(x-\overline{x})^2}{2}dx.$$

Виразимо похибку:

$$R_1(f) = \frac{1}{2}f''(\xi)\int_{x_0}^{x_1}(x-\overline{x})^2 dx,$$

$$\inf(f''(x)) \leqslant f''(\xi) \leqslant \sup(f''(x)),$$

$$x_0 \leqslant x \leqslant x_1,$$

$$R_1(f) = \frac{1}{2}f''(\xi)\frac{(x-\overline{x})^3}{3}\Big|_{x_0}^{x_1} = \frac{1}{2}f''(\xi)\,\frac{1}{3}\left(\frac{h^3}{8} + \frac{h^3}{8}\right).$$

Виходячи з того, що $h = x_1 - x_0$, визначимо

$$R_1(f) = \frac{f''(\xi)}{24} h^3.$$

Приймемо, що $M_2 = \max|f''(\xi)|$. Тоді для похибки формули центральних прямокутників на частковому відрізку справедлива оцінка

$$R_1(f) \leqslant \frac{h^3}{24} M_2.$$

Підсумовуючи праві частини рівняння (7.27), отримаємо узагальнену формулу центральних прямокутників (рис. 7.2):

$$\int_a^b f(x)\,dx = h\left[f\left(x_0 + \frac{h}{2}\right) + f\left(x_1 + \frac{h}{2}\right) + \ldots + f\left(x_{n-1} + \frac{h}{2}\right)\right] + R(f) = \\ = h\sum_{k=1}^{n} f\left(x + \frac{(2k-1)h}{2}\right) + R(f). \tag{7.28}$$

Загальна похибка цієї формули дорівнює сумі похибок на всіх часткових відрізках:

$$R(f) = \frac{h^3}{24}\left(f''(\xi_1) + f''(\xi_2) + \ldots + f''(\xi_n)\right), \quad \text{де} \quad x_{i-1} < \xi_i < x_i, \ i = 1, 2, \ldots, n,$$

чи

$$R(f) = \frac{h^3}{24} n f''(\eta) = \frac{(b-a)}{24} h^2 f''(\eta), \\ R(f) = \frac{h^2(b-a)}{24} M_2, \tag{7.29}$$

де $M_2 = \max|f''(\xi)|$, $\xi \in [a; b]$.

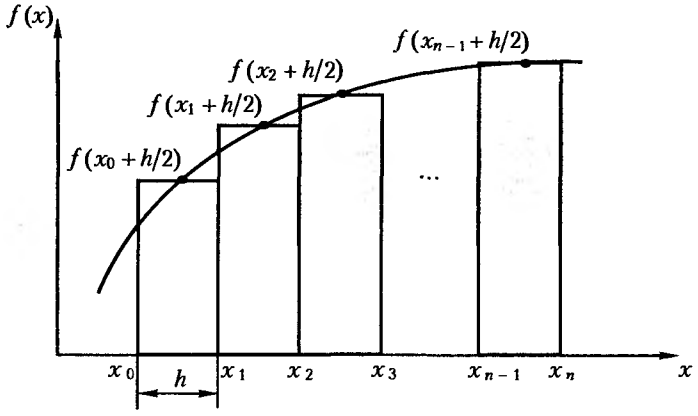


Рис. 7.2. Геометрична інтерпретація методу центральних прямокутників на інтервалі

Порядок точності отриманої формули — $O(h^2)$.

Для оцінювання точності потрібно знати найбільше значення другої похідної. Щоб похибка не перевищувала задане значення ε, крок інтегрування необхідно вибирати з умови

$$h \leqslant \sqrt{\frac{24\varepsilon}{(b-a)M_2}}. \tag{7.30}$$

Приклад 7.3

Методом центральних прямокутників обчислимо інтеграл

$$\int_{0,2}^{0,8} \frac{\sin(2x+0,5)}{2+\cos(x^2+1)}\,dx$$

із точністю ε = 0,0001.

Спочатку визначимо, з яким кроком необхідно вести розрахунки, щоб забезпечити бажану точність. Задамо функцію і обчислимо її другу похідну:

```
In[]:= f[x_] = Sin[2*x + 0.5]/(2 + Cos[x^2 + 1]);
       f2[x_] = Dt[f[x], {x, 2}]
```

Вона матиме такий вигляд:

Out[]=

$$-\frac{4\,\mathrm{Sin}[0.5+2x]}{2+\mathrm{Cos}[1+x^2]}+\frac{8x\,\mathrm{Cos}[0.5+2x]\,\mathrm{Sin}[1+x^2]}{(2+\mathrm{Cos}[1+x^2])^2}+$$
$$\mathrm{Sin}[0.5+2x]\left(\frac{4x^2\mathrm{Cos}[1+x^2]}{(2+\mathrm{Cos}[1+x^2])^2}+\frac{2\,\mathrm{Sin}[1+x^2]}{(2+\mathrm{Cos}[1+x^2])^2}+\frac{8x^2\,\mathrm{Sin}[1+x^2]^2}{(2+\mathrm{Cos}[1+x^2])^3}\right)$$

Тепер побудуємо графік функції для другої похідної (рис. 7.3) на заданому інтервалі інтегрування:

```
In[]:= Plot[f2[x], {x, 0.2, 0.8}]
```

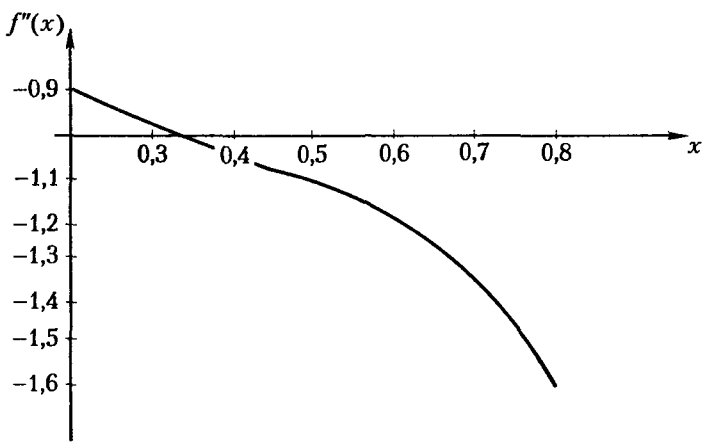


Рис. 7.3. Графік другої похідної функції

За графіком визначимо, що M2 = f2[0.8]. Знайдемо це значення:

```
In[]:= M2 = -N[f2[0.8]]
Out[]= 1.62082
```

Тепер можна обчислити крок інтегрування, виходячи з умови (7.30), як:

```
In[]:= h = Sqrt[24*0.0001/(0.6*M2)]
Out[]= 0.0496779
```

Задамо $h = 0{,}04$, тоді $n = 15$; обчислимо значення інтеграла:

```
In[]:= h = 0.04;
       n = IntegerPart[(0.8 - 0.2)/h];
       J = h*Sum[f[0.2 + h*(i - 0.5)], {i, 1, n}]
```

Одержимо значення інтеграла J = 0.248403.
Перевіримо результат, скориставшись стандартним оператором пакета Mathematica:

```
In[]:= JM = NIntegrate[f[x], {x, 0.2, 0.8}]
Out[]= 0.248357
```

Як бачимо, результати розрахунків практично не відрізняються. Визначимо похибку за правилом Рунге. Для цього обчислимо інтеграл з кроком $h_1 = 0{,}5h$ і збільшимо вдвічі кількість точок:

```
In[]:= h1 = h/2;
       n = IntegerPart[(0.8 - 0.2)/h1];
       J1 = h1*Sum[f[0.2 + h1*(i - 0.5)], {i, 1, n}]
Out[]= 0.248369
```

Похибка, знайдена за правилом Рунге, для формули центральних прямокутників дорівнює

```
In[]:= r = J - J1
Out[]= 0.0000346144
```

Це задовольняє заданій умові.

Формули інтегрування на основі прямокутників можуть бути побудовані й за іншого розташування вузлів. У загальному випадку формулу прямокутників можна записати в такому вигляді:

$$\int_{x_0}^{x_n} f(x)\,dx = h\sum_{i=0}^{n-1} f(q + ih) + R(f). \tag{7.31}$$

Виходячи з (7.31) формулу центральних прямокутників (7.28) можна отримати, якщо за q прийняти значення $q = x_0 + h/2$. Коли $q = x_0$ отримаємо формулу лівих прямокутників (рис. 7.4), а для $q = x_0 + h$ — формулу правих прямокутників (рис. 7.5).

У формулі лівих прямокутників залишковий член має такий вигляд:

$$R(f) = \frac{x_n - x_0}{2} h f'(\xi).$$

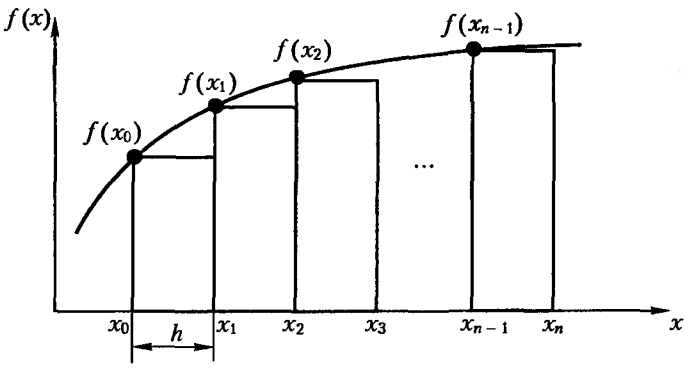


Рис. 7.4. Геометрична інтерпретація методу лівих прямокутників

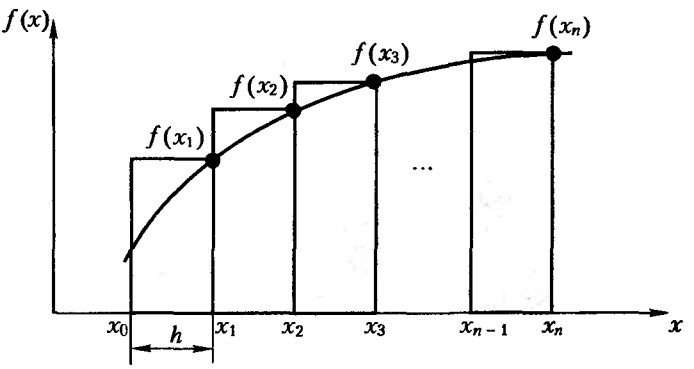


Рис. 7.5. Геометрична інтерпретація методу правих прямокутників

Залишковий член формули правих прямокутників має такий вигляд:

$$R(f) = -\frac{x_n - x_0}{2} h f'(\xi), \quad x_0 \leqslant \xi \leqslant x_n.$$

Похибка обчислень двох останніх формул дорівнює $O(h)$; вона більша, ніж для формули центральних прямокутників, через порушення симетрії.

7.2.2. Формула трапецій

Тепер розглянемо підхід, що полягає в заміні функції $f(x)$ інтерполяційним поліномом першого степеня. Спочатку визначимо варіант заміни на частковому відрізку $[x_0, x_1]$ площі криволінійної трапеції площею прямокутної трапеції, побудованої по тих же точках (рис. 7.6). Ця заміна може бути зроблена у такий спосіб:

$$\int_{x_0}^{x_1} f(x)\,dx = (x_1 - x_0)\left[\frac{1}{2} f(x_0) + \frac{1}{2} f(x_1)\right] + R_1(f) =$$

$$= \frac{x_1 - x_0}{2}[f(x_0) + f(x_1)] + R_1(f).$$

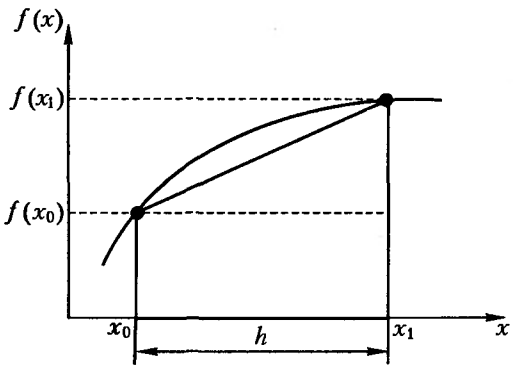


Рис. 7.6. Геометрична інтерпретація методу трапецій на частковому відрізку

Оцінимо похибку $R_1(f)$ у разі заміни $f(x)$ інтерполяційним поліномом першого степеня на частковому відрізку $[x_0, x_1]$:

$$f(x) = f(x_0) + (x - x_0)f(x_0, x_1) + r_1(x),$$

де

$$r_1(x) = \frac{(x - x_0)(x - x_1)f''(\xi)}{2!}.$$

Виконаємо такі перетворення:

$$\int_{x_0}^{x_1} f(x)\,dx = \int_{x_0}^{x_1} [f(x_0) + (x - x_0)f(x_0, x_1) + r_1(x)]\,dx =$$

$$= f(x_0)(x_1 - x_0) + \frac{(x - x_0)^2}{2} f(x_0, x_1)\Big|_{x_0}^{x_1} + \int_{x_0}^{x_1} r_1(x)\,dx =$$

$$= f(x_0)(x_1 - x_0) + \frac{(x_1 - x_0)^2}{2} \frac{f(x_1) - f(x_0)}{x_1 - x_0} + \int_{x_0}^{x_1} \frac{(x - x_0)(x - x_1)f''(\xi)}{2} dx =$$

$$= (x_1 - x_0)\left[\frac{2f(x_0) + f(x_1) - f(x_0)}{2}\right] + \underbrace{\frac{f''(\xi)}{2} \int_{x_0}^{x_1} (x - x_0)(x - x_1)\,dx}_{\text{похибка}},$$

$$\int_{x_0}^{x_1} (x - x_0)(x - x_1)\,dx = -\frac{1}{6}(x_1 - x_0)^3,$$

$$|R_1(f)| \leqslant \frac{(x_1 - x_0)^3}{12} M_2 = \frac{h^3}{12} M_2,$$

де $M_2 = \max\limits_{x_0 \leqslant x \leqslant x_1} |f''(\xi)|$.

Складена формула трапецій для всього інтервалу має такий вигляд:

$$\int_a^b f(x)\,dx = \frac{h}{2}\left[f(x_0) + 2\sum_{i=1}^{n-1} f(x_i) + f(x_n)\right] + R(f). \qquad (7.32)$$

Геометрично метод трапецій проілюстровано на рис. 7.7.

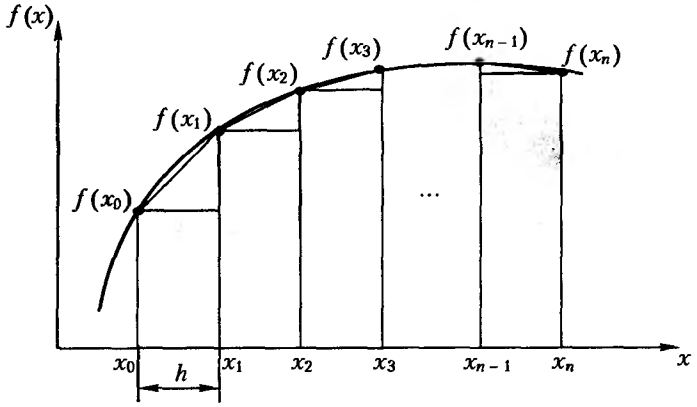


Рис. 7.7. Геометрична інтерпретація методу трапецій для інтервалу

Похибка цього методу обчислюється як сума похибок на всіх часткових інтервалах:

$$R(f) = \sum_{i=1}^{n} -\frac{h^3}{12} f''(\xi_i) = -\frac{h^3}{12} n f''(\eta) = -\frac{h^2(b-a)}{12} f''(\eta)$$

і визначається як

$$|R(f)| \leqslant \frac{h^2(b-a)}{12} f''(\eta) = \frac{h^2(b-a)}{12} M_2. \qquad (7.33)$$

Приклад 7.4

Обчислимо значення інтеграла від функції

$$f(x) = 300x^5 - 600x^4 + 500x^3 - 100x^2 + 50x + 0{,}5$$

у межах $a = 0$ і $b = 0{,}8$, використовуючи метод трапецій (7.32). Оцінимо також похибку обчислень. Спочатку визначимо в Mathematica функцію $f(x)$ і скористаємося можливостями пакета для інтегрування:

```
In[]:= f[x_]:= 300x^5 - 600x^4 + 500x^3 - 100x^2 + 50x + 0.5
       Jt = Integrate[f[x], {x, 0, 0.8}]
Out[]= 24.3189
```

Тепер знайдемо чисельний розв'язок за допомогою методу трапецій. Спочатку задамо кількість підінтервалів $n = 2$, а потім збільшимо їх кількість вдвічі й повторимо обчислення, досліджуючи залежність похибки оцінки інтеграла від кількості підінтервалів:

```
In[]:= a = 0; b = 0.8;
       Do [h[i]=(b - a)/(i + 1);
       J[i] = h[i]/2*(f[a] + 2*Sum[f[a + h[i]*j], {j,i}] + f[b]);
       Eps[i] = Abs[(Jt - J[i])/Jt*100], {i,9} ]
       p = TableForm[Table[{i + 1, J[i], Eps[i]}, {i, 9}],
       TableDirections → {Column, Row}, TableAlignments → Center]
```

Отримаємо такі результати:

n	J	Eps, %
2	26.7936	10.1759
3	25.4188	4.52262
4	24.9376	2.54397
5	24.7149	1.62814
6	24.5939	1.13065
7	24.5209	0.830684
8	24.4736	0.635993
9	24.4411	0.502513
10	24.4179	0.407035

7.2.3. Формули чисельного інтегрування вищих порядків

Розглянуті вище методи чисельного інтегрування мають невисокий порядок точності ($O(h)$ або $O(h^2)$). Тепер спробуємо отримати методи, порядок точності яких значно вище. Почнемо з виведення формули Сімпсона. Для обчислення інтеграла

$$J_i = \int_{x_i-h}^{x_i+h} f(x)\,dx$$

на частковому відрізку $[x_i - h; x_i + h]$ замінимо функцію $f(x)$ параболою, що проходить через точки $(x_i + jh; f(x_i + jh))$, $j = -1, 0, 1$. На частковому відрізку $[x_0, x_2]$ метод Сімпсона проілюстрований на рис. 7.8.

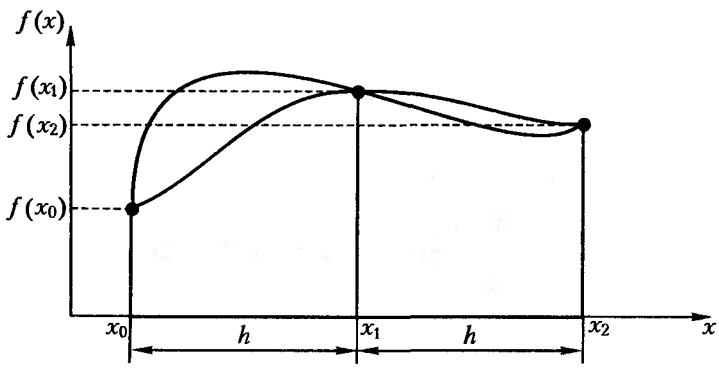


Рис. 7.8. Геометрична інтерпретація методу Сімпсона на чвстковому відрізку

Зобразимо функцію $f(x)$ наближено за допомогою інтерполяційного багаточлена

$$f(x) \approx L_{2,i}(x), \quad x \in [x_{i-1}, x_{i+1}],$$

де $L_{2,i}(x)$ — інтерполяційний поліном другого степеня. Запишемо його в Mathematica. Введемо таблицю для інтерполяції:

```
In[]:= Tb = Table[{xi + j*h, f[xi + j*h]}, {j, -1, 1}]
Out[]= {{-h + xi, f[-h + xi]}, {xi, f[xi]}, {h + xi, f[h + xi]}}
```

Побудуємо за цією таблицею інтерполяційний поліном:

```
In[]:= L2i = InterpolatingPolynomial[Tb, t]
```

$$\text{Out[]= } f[-h+x_i]+\left(\frac{f[x_i]-f[-h+x_i]}{h}+\frac{\left(-\frac{f[x_i]-f[-h+x_i]}{h}+\frac{-f[x_i]+f[h+x_i]}{h}\right)(t-x_i)}{2h}\right)(h+t-x_i)$$

Проводячи інтегрування, отримаємо:

```
In[]:= simps = Integrate[L2i, {t, xi - h, xi + h}]
```

$$\text{Out[]= } \frac{4}{3}h\,f[x_i]+\frac{1}{3}h\,f[-h+x_i]+\frac{1}{3}h\,f[h+x_i]$$

Спростимо останній вираз:

```
In[]:= Simpsn = Factor[simps]
```

$$\text{Out[]= } \frac{1}{3}h(4\,f[x_i]+f[-h+x_i]+f[h+x_i])$$

і отримаємо наближену рівність:

$$\int_{x_i-h}^{x_i+h} f(x)\,dx \approx \int_{x_i-h}^{x_i+h} L_{2,i}(x)\,dx = \\ = \frac{h}{3}(f(x_i-h)+4f(x_i)+f(x_i+h)), \tag{7.34}$$

яка називається формулою Сімпсона, чи формулою парабол.
Складена (узагальнена) формула Сімпсона має такий вигляд:

$$\int_a^b f(x)\,dx \approx \frac{h}{3}[f_0+4(f_1+f_3+\ldots+f_{2n-1})+2(f_2+f_4+\ldots+f_{2n-2})+f_{2n}] = \\ = \frac{h}{3}\left[f_0+f_{2n}+4\sum_{i=1}^{n} f_{2i-1}+2\sum_{i=1}^{n-1} f_{2i}\right], \tag{7.35}$$
$$2hn = b-a.$$

Перейдемо до оцінки похибки формули (7.35):

$$r_i = \int_{x_i-h}^{x_i+h} f(x)\,dx - \frac{h}{3}(f(x_i-h)+4f(x_i)+f(x_i+h)).$$

Замість змінної в інтегралі останньої формули введемо $z = x - x_i$ і отримаємо такий вираз:

$$r_i = \int_{-h}^{h} f(x_i+z)\,dy - \frac{h}{3}(f(x_i-h)+4f(x_i)+f(x_i+h)). \tag{7.36}$$

Розкладемо підінтегральну функцію $f(x_i + z)$ за приростом z у ряд Тейлора, зберігаючи члени до четвертого порядку, і проінтегруємо отриманий вираз:

```
In[]:= J[h] = Integrate[Normal[Series[f[xi + z], {z, 0, 4}]], {z, -h, h}]
```

$$\text{Out[]= } 2h\,f[x_i] + \frac{1}{3}h^3 f''[x_i] + \frac{1}{60}h^5 f^{(4)}[x_i]$$

Підставимо вираз для інтеграла у формулу (7.36), розкладемо вираз для функції Simpsn у ряд Тейлора і знайдемо оцінку для похибки на частковому інтервалі

```
In[]:= ri = Simplify[J[i] - h/3*Normal[Series[f[xi - h] + 4f[xi + h], {h, 0, 4}]]];
       Print["ri = ", ri]
```

$$r_i = -\frac{1}{90}h^5 f^{(4)}[x_i]$$

Похибка складеної формули Сімпсона на всьому інтервалі має таку оцінку:

$$|R| \leqslant \frac{h^4(b-a)}{180} M_4, \qquad M_4 = \sup\left|f^{(4)}(x)\right|, \quad x \in [a, b]. \tag{7.37}$$

Звідси видно, що формула Сімпсона значно точніша за формули прямокутників і трапецій. Вона може бути застосована для рівномірно розташованих вузлів у разі парної кількості підінтервалів n і непарної кількості вузлів. Для непарної кількості підінтервалів і парної кількості вузлів застосовується модифікація формули Сімпсона, відома як друга формула Сімпсона:

$$\int_{x_i-h}^{x_i+h} f(x)\,dx \approx \int_{x_i-h}^{x_i+h} L_{3,i}(x)\,dx = \frac{3h}{8}(f(x_i) + 3f(x_i + h) + 3f(x_i + 2h) + f(x_i + 3h)), \tag{7.38}$$

яку отримують за аналогією з попередньою, використовуючи інтерполяційний поліном Лагранжа третього порядку. Вона характеризується (незважаючи на більшу кількість використаних відліків) тією ж похибкою, що й основна формула (7.34), для неї похибка на частковому інтервалі

$$r_i = -\frac{3}{80}h^5 f^{(4)}[x_i],$$

а на всьому відрізку інтегрування —

$$|R| \leqslant \frac{3h^4(b-a)}{80} M_4, \qquad M_4 = \sup\left|f^{(4)}(x)\right|, \quad x \in [a, b].$$

Ще однією формулою чисельного інтегрування, яка будується заміною підінтегральної функції інтерполяційним поліномом, може бути формула Ньютона:

$$\int_{x_0}^{x_n} f(x)\,dx = \frac{3}{8}h[f_0 + f_n + 2(f_3 + f_6 + \ldots + f_{n-3}) + \\ +3(f_1 + f_2 + f_4 + f_5 + \ldots + f_{n-2} + f_{n-1})] + R_4(f) \tag{7.39}$$

із залишковим членом

$$|R| < \frac{h^4(b-a)}{80}M_4, \\ M_4 = \sup\left|f^{(4)}(x)\right|,\ x \in [a, b]. \tag{7.40}$$

У формулі (7.39) кількість підінтервалів n передбачається кратним 3, а значення підінтегральної функції $f_i = f(x + ih)$.

Приклад 7.5

Ще раз обчислимо оцінку інтеграла від функції

$$f(x) = 300x^5 - 600x^4 + 500x^3 - 100x^2 + 50x + 0{,}5$$

у межах $a = 0$ і $b = 0{,}8$, розглянутого в прикладі 7.4, використовуючи складений метод Сімпсона (7.35) для $n = 10$.

```
In[]:= f[x_]:= 300 x^5 - 600 x^4 + 500 x^3 - 100 x^2 + 50 x + 0.5;
       n = 10; h = (b - a)/(2n);
       Js = h/3*(f[0] + 4*Sum[f[(2i - 1)*h], {i,n}] + 2*Sum[f[2i*h], {i,n-1}] + f[2n*h])
Out[]= 24.3189
```

Необхідно відзначити, що похибка $\varepsilon = 4{,}38265 \cdot 10^{-14}$ % менша, ніж у прикладі 7.4 (за того ж значення $n = 10$), де обчислення проводилися за формулою трапецій, а похибка дорівнювала 0,407035 %.

7.2 4. Практичні способи оцінювання похибки інтегрування

Оцінювати значення похибки можна різними способами.

- **За залишковим членом.** Якщо під час обчислення залишкового члена виникають труднощі з визначенням максимуму похідної (підінтегральна функція складна чи задана таблично), варто застосовувати наближені формули для похибок, виражені через скінченні різниці:

$R_1 \approx \pm\frac{x_n - x_0}{2}\nabla\overline{y}$ — для лівих і правих прямокутників;

$R_2 \approx \frac{x_n - x_0}{12}\nabla^2\overline{y}$ — для формули трапецій;

$$R_3 \approx \frac{x_n - x_0}{180}\nabla^4 \overline{y} \quad \text{– для формули Сімпсона;}$$

$$R_4 \approx \frac{x_n - x_0}{80}\nabla^4 \overline{y} \quad \text{– для формули Ньютона.}$$

Для цього за табличними значеннями складається таблиця скінченних різниць певного порядку, з якої отримують максимальне за модулем значення відповідної різниці: $\nabla\overline{y}, \nabla^2\overline{y}$ чи $\nabla^4\overline{y}$.

- **За правилом Рунге.** Позначимо через J_h і $J_{h/2}$ наближені значення інтеграла J, знайдені за однією з формул (7.31, 7.32, 7.35, 7.38) із кроками h і $h/2$ відповідно. Тоді абсолютна похибка інтегрування наближено оцінюється за таким правилом Рунге:

$$\Delta \approx \frac{J_{h/2} - J_h}{2^k - 1} = \theta(J_{h/2} - J_h),$$

де $\theta = 1/(2^k - 1)$, k – порядок залишкового члена формули інтегрування (для формули трапецій $k = 2$, для формули Сімпсона $k = 4$).

- **Екстраполяцією за Річардсоном.** Можна знайти уточнене за Річардсоном значення інтеграла J із похибкою $O(h^{2k})$:

$$J = J_{h/2} + \theta(J_{h/2} - J_h).$$

Приклад 7.6

Для підінтегральної функції

$$f(x) = 300x^5 - 600x^4 + 500x^3 - 100x^2 + 50x + 0{,}5,$$

розглянутої в прикладах 7.4 і 7.5, оцінимо інтеграл у межах $a = 0$ і $b = 0{,}8$, застосовуючи екстраполяцію Річардсона і метод трапецій, як це реалізовано в методі Ромберга.

Скориставшись результатами приклада 7.4, перепишемо фрагмент отриманої там таблиці:

n	J	Eps, %
2	26.7936	10.1759
4	24.9376	2.54397

За аналогією з формулою Річардсона (7.21) запишемо:

$$J = J(h_2) + \frac{1}{(h_1/h_2)^2 - 1}[J(h_2) - J(h_1)].$$

Користуючись прикладом 7.4, визначимо додаткову функцію:

```
In[]:= Jr = 4/3*J[3] - 1/3*J[1]
Out[]= 24.3189
```

Результат отримано з похибкою $4{,}38265 \cdot 10^{-14}$ %, що за точністю збігається з оцінкою інтеграла складеним методом Сімпсона в прикладі 7.5.

7.2.5. Вибір кроку інтегрування

Завдання полягає у визначенні кроку h, що забезпечує задану точність ε обчислення інтеграла за обраною формулою. Існує два основних способи задавання допустимого значення кроку.

- **За залишковим членом.** Використовуючи формулу відповідного залишкового члена $R(x)$, вибирають h таким, щоб виконувалася нерівність $|R(x)| < \varepsilon/2$. Потім із отриманим кроком обчислюють інтеграл за квадратурною формулою. Обчислення варто робити з таким числом цифр, щоб похибка заокруглення не перевищувала ε.
- **Послідовним подвоєнням числа кроків.** Обчислюють інтеграл J за обраною квадратурною формулою двічі: спочатку з деяким кроком h, потім із кроком $h/2$, тобто подвоюють кількість n. Якщо $\theta|J_h - J_{h/2}| < \varepsilon$, то приймають $J_h \approx J_{h/2}$. Коли виявиться, що ця умова не виконується, то розрахунок повторюють із кроком $h/4$. Як початковий крок іноді можна рекомендувати число, близьке до $\sqrt[k]{\varepsilon}$.

Приклад 7.7

За допомогою методів чисельного інтегрування можна обчислювати і невизначені інтеграли. Наведемо приклад обчислення невизначеного інтеграла

$$J = \int_0^{\infty} \frac{e^{-x^2}}{1+x^2} dx,$$

значення якого необхідно знайти з похибкою, яка не перевищує заданої величини ε. Спочатку треба перевірити збіжність інтеграла J. Оскільки має місце нерівність

$$\frac{e^{-x^2}}{1+x^2} \leqslant \frac{1}{1+x^2}, \quad 0 \leqslant x \leqslant \infty,$$

то

$$\int_0^{\infty} \frac{e^{-x^2}}{1+x^2} dx \leqslant \int_0^{\infty} \frac{1}{1+x^2} dx = \operatorname{arctg} x \Big|_0^{\infty} = \frac{\pi}{2}.$$

Отже, інтеграл J сходиться. Записавши його як

$$J = \int_0^{\alpha} \frac{e^{-x^2}}{1+x^2} dx + \int_{\alpha}^{\infty} \frac{e^{-x^2}}{1+x^2} dx,$$

знайдемо таке значення α, за якого значення другого інтеграла було б менше, ніж $0{,}5\varepsilon$, тобто

$$\int_{\alpha}^{\infty} \frac{e^{-x^2}}{1+x^2} dx \leqslant \frac{\varepsilon}{2}.$$

Остання умова буде виконана, якщо зажадати виконання умови

$$\int_{\alpha}^{\infty} \frac{e^{-x^2}}{1+x^2} dx \leqslant \int_{\alpha}^{\infty} e^{-\alpha x} dx \leqslant \frac{\varepsilon}{2}, \quad \alpha > 1.$$

Оскільки

$$\int_{\alpha}^{\infty} e^{-\alpha x}\,dx = \frac{e^{-\alpha^2}}{\alpha},$$

то α визначається з умови

$$e^{-\alpha} \leqslant 0{,}5\alpha\varepsilon.$$

Таким чином, якщо ми приймемо, що

$$J \approx \int_{0}^{\alpha} \frac{e^{-x^2}}{1+x^2}\,dx,$$

то отримаємо похибку оцінки інтеграла не більше ніж $0{,}5\varepsilon$. Тепер необхідно обчислити значення останнього інтеграла з точністю, не меншою за $0{,}5\varepsilon$. Нехай задане значення $\varepsilon = 0{,}0001$, визначимо відповідне α. Для цього побудуємо графік підінтегральної функції (рис. 7.9) і знайдемо початкове наближення кореня:

```
In[]:= ep = 0.0001;
       g[x_] = E^(-x^2) - 0.5*x*ep;
       Plot[g[x], {x, 2.8, 3.2}]
```

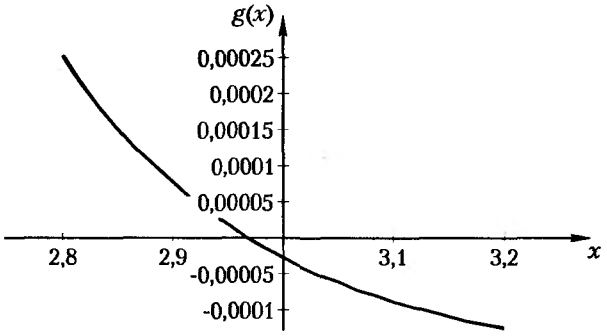


Рис. 7.9. Графік функції $\dfrac{e^{-x^2}}{1+x^2}$

Тепер можна уточнити значення α:

```
In[]:= A = FindRoot[g[x]] == 0, {x, 3}];
       a = x/.A[1]]
Out[]= 2.96905
```

Таким чином, можна вибрати $\alpha > 2{,}96905$. Приймемо $\alpha = 3$ і знайдемо крок для інтегральної формули Сімпсона, що забезпечує точність $0{,}5\varepsilon$. Похибка формули Сімпсона визначається формулою (7.37).

Знайдемо четверту похідну підінтегральної функції та визначимо M_4:

```
In[]:= f[x_]:= E^(-x^2)/(1 + x^2);
       f4[x_] = Simplify[Dt[f[x], {x, 4}]]
```

$$\text{Out[]}= \frac{1}{(1+x^2)^5}\left(4e^{-x^2}\left(15 - 150x^2 - 70x^4 + 22x^6 + 39x^8 + 20x^{10} + 4x^{12}\right)\right)$$

Побудуємо графік четвертої похідної підінтегральної функції (рис. 7.10):

```
In[]:= Plot[ f4[x], {x, 0, 3}]
```

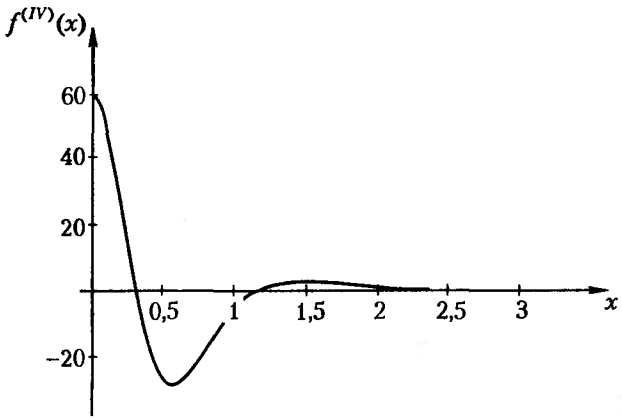


Рис. 7.10. Графік четвертої похідної від функції $\dfrac{e^{-x^2}}{1+x^2}$

Виходячи з отриманої формули і графіка (рис. 7.10), маємо $M_4 = 60$. Тепер можна обчислити оцінку для максимального кроку h:

$$h \leqslant \sqrt[4]{\frac{180\varepsilon}{2(b-a)M_4}}$$

```
In[]:= M = 60; Eps = ep/2;
       a = 0; b = alpha;
       h = N[Power[180*Eps/(2*(b - a)*M), 1/4]]
Out[]= 0.0707107
```

Визначимо кількість кроків інтегрування

```
In[]:= n = (b - a)/h/2
Out[]= 21.2132
```

Для кількості кроків інтегрування оберемо найближче ціле зверху

```
In[]:= n = 22;
```

Таким чином, крок інтегрування становить

```
In[]:= h = N[(b - a)/2/n]
Out[]= 0.0681818
```

і значення інтеграла за формулою Сімпсона можна обчислювати таким чином:

```
In[]:= Jp = h/3*(f[0] + 4*Sum[f[(2i-1)*h], {i,n}] + 2*Sum[f[2i*h], {i,n-1}] + f[2n*h])
Out[]= 0.671645
```

Тепер знайдемо значення заданого інтеграла засобами пакета Mathematica:

```
In[]:= J = N[Integrate[f[x], {x, 0, ∞}]]
Out[]= 0.671647
```

Знайдемо різницю значень інтеграла, обчисленого нами і отриманого засобами пакета Mathematica:

```
In[]:= δ = Jp - J
Out[]= -1.79927×10^-6
```

7.2.6. Рекурентні формули та інтегрування за Ромбергом

Скористатися апріорними оцінками квадратурних формул подібно тому, як це було зроблено в попередніх розділах, вдається не завжди. Тому обчислення інтегралів із потрібною точністю виконують звичайно, використовуючи послідовне зменшення кроку поділом його навпіл до виконання визначених критеріїв точності. Під час кожного такого зменшення кроку подвоюється кількість підінтервалів, а отже, подвоюється число точок, у яких треба обчислювати значення підінтегральної функції.

Розглянемо спочатку алгоритм, який дозволяє обчислювати інтеграл на дрібній сітці, використовуючи знайдені значення його на великій попередній сітці, додаючи лише значення в тих точках, що знову з'явилися під час дроблення.

Рекурентна формула трапецій

Позначимо через $T(0) = h(f(a) + f(b))/2$ формулу трапецій із кроком $h = b - a$. Потім для кожного $j \geqslant 1$ визначимо $T(j) = T(f, h)$ — формулу трапецій із кроком $h = (b-a)/2^j$. Тоді справедлива така рекурентна формула:

$$T(j) = \frac{1}{2}T(j-1) + h\sum_{k=1}^{n} f(x_{2k-1}), \quad j = 1, 2, \ldots, \tag{7.41}$$

де $n = 2^{j-1}$ і $x_k = a + kh$.

Для доведення використаємо для парних вузлів формулу трапецій із кроком $2h$:

$$T(j-1) = h\sum_{k=0}^{n} f(x_{2k}).$$

Для всіх вузлів використаємо формулу з кроком h:

$$T(j) = \frac{h}{2}\sum_{k=0}^{n} f(x_{2k}) + \frac{h}{2}\sum_{k=1}^{n} f(x_{2k-1}),$$

звідки й отримаємо формулу (7.41).

Приклад 7.8

Послідовно застосуємо формулу трапецій, щоб обчислити наближення $T(j)$, $j = 0, 1, \ldots, 4$ для інтеграла $\int_0^1 e^{-x^2}\,dx$, використовуючи пакет Mathematica:

```
In[]:= a = 0; b = 1; M = 5;
       f[x_]:= Exp[-x^2];
```

```
T[0] = 0.5(b - a)*(f[a] + f[b]);
DO [h = (b - a)/2^j; n = 2^(j - 1);
T[j] = 0.5*T[j - 1] + h*Sum[f[a + (2k - 1)*h], {k, n}], {j, 1, M-1} ]
Array[T, M, 0]
Out[]= {0.68394, 0.73137, 0.742984, 0.745866, 0.746585}
```

Отримано наближені значення інтеграла з кроками $h = 1, 1/2, 1/4, 1/8, 1/16$ відповідно.

Рекурентна формула Сімпсона

Наведений нижче результат встановлює важливе співвідношення між формулами трапецій і Сімпсона. Розглянемо спочатку наближені значення інтеграла:

$$I_i = \int_{x_i-h}^{x_i+h} f(x)\,dx,$$

отримані за формулою трапецій із кроком $2h$ і h, а також за формулою Сімпсона з кроком h (рис 7.11).
Маємо

$$I_i^T(2h) = h(f_{i-1} + f_{i+1}),$$

$$I_i^T(h) = \frac{h}{2}(f_{i-1} + 2f_i + f_{i+1}),$$

$$I_i^S(h) = \frac{h}{3}(f_{i-1} + 4f_i + f_{i+1}).$$

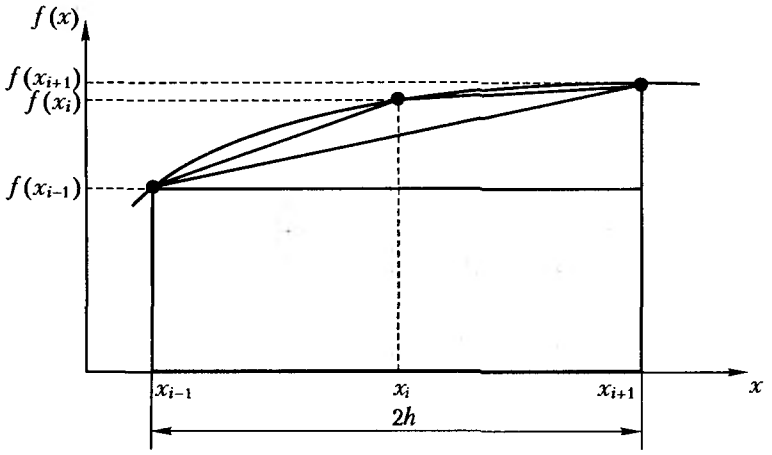


Рис. 7.11. Зв'язок між формулами трапецій і Сімпсона

Помноживши другий вираз на 4/3 і віднявши від нього перший, помножений на 1/3, отримаємо

$$\frac{4I_i^T(h) - I_i^T(2h)}{3} = \frac{h}{3}(f_{i-1} + 4f_i + f_{i+1}) = I_i^S(h).$$

Оскільки

$$I = \sum_{i=1}^{n} I_i, \quad \text{де} \quad n = (b-a)/2h,$$

то останній вираз справедливий для всього відрізка інтегрування, тобто для формули Сімпсона справедливе таке співвідношення:

$$S(f,h) = \frac{4T(f,h) - T(f,2h)}{3}, \tag{7.42}$$

або в позначеннях, використаних у рекурентних формулах:

$$S(j) = \frac{4T(j) - T(j-1)}{3}, \quad j = 1, 2, \ldots. \tag{7.43}$$

Рекурентна формула Буля

Тепер отримаємо співвідношення для формули Буля (рис. 7.12), яка для обчислення значення інтеграла на частковому відрізку має такий вигляд:

$$J_i = \int_{x_i-2h}^{x_i+2h} f(x)\,dx \approx \frac{2h}{45}(7f_{i-2} + 32f_{i-1} + 12f_i + 32f_{i+1} + 7f_{i+2}). \tag{7.44}$$

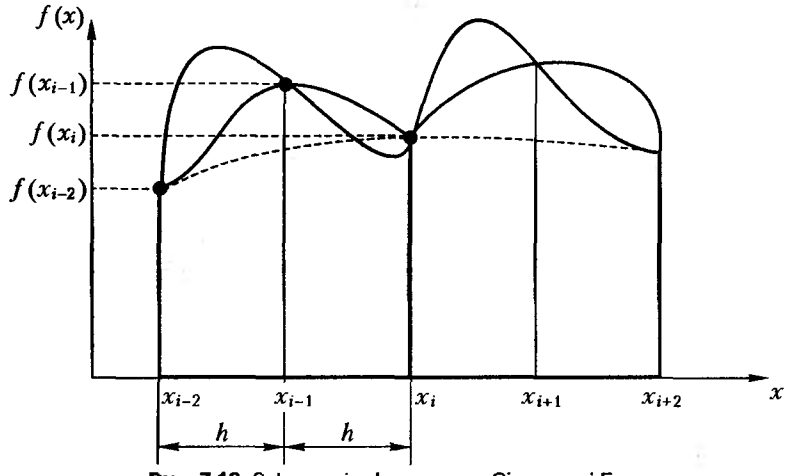


Рис. 7.12. Зв'язок між формулами Сімпсона і Буля

Розглянемо спочатку оцінки для інтеграла

$$I_i = \int_{x_i-2h}^{x_i+2h} f(x)\,dx,$$

отримані за формулою Сімпсона з кроком $2h$ і h, а також за формулою Буля (7.44) із кроком h (рис. 7.12). Маємо

$$J_i^S(h) = \frac{h}{3}[(f_{i-2} + 4f_{i-1} + f_i) + (f_i + 4f_{i+1} + f_{i+2})],$$

$$J_i^S(2h) = \frac{2h}{3}(f_{i-2} + 4f_i + f_{i+2}).$$

Помножимо першу з цих формул на 16 і віднімемо від неї другу формулу:

$$16J_i^S(h) - J_i^S(2h) = \frac{16h}{3}[(f_{i-2} + 4f_{i-1} + f_i) + (f_i + 4f_{i+1} + f_{i+2})] -$$

$$-\frac{2h}{3}(f_{i-2} + 4f_i + f_{i+2}) = \frac{2h}{3}(7f_{i-2} + 32f_{i-1} + 12f_i + 32f_i + 7f_{i+2}).$$

Якщо розділимо останній вираз на 15, то отримаємо формулу Буля (7.44). Таким чином,

$$\frac{16J_i^S(h) - J_i^S(2h)}{15} = \frac{2h}{45}(7f_{i-2} + 32f_{i-1} + 12f_i + 32f_i + 7f_{i+2}) = I_i^B(h).$$

Поширюючи останню формулу на весь відрізок інтегрування, отримаємо для формули Буля співвідношення

$$B(f, h) = \frac{16S(f, h) - S(f, 2h)}{15}, \tag{7.45}$$

або у вигляді рекурентної формули:

$$B(j) = \frac{16S(j) - S(j-1)}{15}, \quad j = 2, 3, \ldots. \tag{7.46}$$

Інтегрування за Ромбергом

У підрозділах 7.2.2 і 7.2.3 було встановлено, що залишкові члени для складених формул трапецій і Сімпсона мають порядок $O(h^2)$ і $O(h^4)$ відповідно. Можна показати, що залишковий член складеної формули Буля має порядок $O(h^6)$.

Припустимо, що у формулі наближеного інтегрування використовуються кроки h і $2h$. У цьому випадку за рекурентними співвідношеннями для формул Сімпсона і потім для формул Буля можна отримати більш точний результат. Кожен наступний рівень поліпшення зменшує залишковий член з $O(h^{2n})$ до $O(h^{2n+2})$. Такий процес послідовного уточнення результату називається *інтегруванням за Ромбергом*. Удосконалення інтегрування за Ромбергом ґрунтується на припущенні, що коли $f(x) \in C^n[a, b]$ для всіх точок, то залишковий

член формули трапецій можна подати у вигляді ряду, який містить тільки парні степені h, тобто:

$$\int_a^b f(x)\,dx = T(f,h) + \alpha_1 h^2 + \alpha_2 h^4 + \alpha_3 h^6 + \dots . \tag{7.47}$$

Рекурентну формулу (7.42) можна записати в іншій формі:

$$S(f,h) = \frac{4T(f,h) - T(f,2h)}{3} = T(f,h) + \frac{T(f,h) - T(f,2h)}{3}, \tag{7.48}$$

тут другий доданок є поправкою до формули трапецій із кроком h, що дозволяє отримати більш точний результат за формулою Сімпсона. Ця поправка, отримана за допомогою подвійного розрахунку з кроком h і $2h$, називається *поправкою Річардсона*. Аналогічно отримаємо поправку для формули Буля (7.45):

$$B(f,h) = S(f,h) + \frac{S(f,h) - S(f,2h)}{15}. \tag{7.49}$$

Для запису в загальному випадку процесу послідовного інтегрування за Ромбергом визначимо послідовність $\{R(j,k) : j \geqslant k\}$ формул наближеного інтегрування в такий спосіб:

$$\begin{aligned} &R(j,0) = T(j) \quad \text{для} \quad j \geqslant 0, \text{ послідовна формула трапецій;} \\ &R(j,1) = S(j) \quad \text{для} \quad j \geqslant 1, \text{ послідовна формула Сімпсона;} \\ &R(j,2) = B(j) \quad \text{для} \quad j \geqslant 2, \text{ послідовна формула Буля.} \end{aligned} \tag{7.50}$$

Загальною формулою побудови поправок є:

$$R(j,k) = R(j,k-1) + \frac{R(j,k-1) - R(j-1,k-1)}{4^k - 1}, \quad 1 \leqslant k \leqslant j, \tag{7.51}$$

де другий доданок позначають як

$$\mathrm{Er}(j,k) = \frac{R(j,k-1) - R(j-1,k-1)}{4^k - 1} \tag{7.52}$$

і називають узагальненою поправкою Річардсона.

Приклад 7.9

Скористаємося формулою інтегрування за Ромбергом, щоб знайти наближене значення для визначеного інтеграла:

$$I = \int_0^{\pi/2} (x^2 + x + 1)\cos x\,dx$$

у пакеті Mathematica. Нижче наведена програма інтегрування за методом Ромберга, що використовує рекурентні формули:

```
In[]:= ROMBERG[a_, b_, f_, m_, T_, S_, B_, R_]:= Block[{h, M, k},
       T[0] = 0.5(b - a)*(f[a] + f[b])];
       DO [h = (b - a)/2^j; M = 2^(j-1);
           T[j] = 0.5*T[j-1] + h*Sum[f[a + (2k-1)*h), {k, M}, [j, m]]; S[0] = 0;
       DO [S[j] = (4*T[j] - T[j-1]/3, {j, m}]; B[0] = 0; B[1] = 0;
       DO [B[j] = (16*S[j] - S[j-1])/15, {j, 2, m}]; R[0] = 0; R[1] = 0; R[2] = 0;
       DO [R[j] = (64*B[j] - B[j-1])/63, {j, 3,m}] ];
```

Введемо початкові дані задачі і звернемося до процедури інтегрування за Ромбергом:

```
In[]:= a = 0; b = pi/2; f[x_]:= (x^2 + x + 1)*Cos[x]; m = 6; Array [R, m, 0];
       Array [S, m, 0]; Array [B, m, 0];  ROMBERG[a, b, f, m, T, S, B, R];
```

Надрукуємо результати:

```
In[]:= V = N[Table[{T[i], S[i], B[i], R[i]}, {i, 0, m}];
       Print["Таблиця 7.5. "];
       TableForm[V, TableHeadings → {Automatic, {"R(j, 0) = T", "R(j, 1) = S",
       "R(j, 2) = B", "R(j, 3)"}}]
```

Таблиця 7.5. Результати застосування формул інтегрування за Ромбергом

j	$R(j, 0) = T$	$R(j, 1) = S$	$R(j, 2) = B$	$R(j, 3)$
1	0,785398	0	0	0
2	1,72681	2,04062	0	0
3	1,96053	2,03844	2,0383	0
4	2,01879	2,03821	2,0382	2,0382
5	2,03335	2,0382	2,0382	2,0382
6	2,03698	2,0382	2,0382	2,0382
7	2,03789	2,0382	2,0382	2,0382

Обчислимо значення цього ж інтеграла в пакеті Mathematica:

```
In[]:= N[Integrate[f[x], {x, 0, π/2}], 12]
Out[]= 2.03819742707
```

Як бачимо, результат отримано з досить високою точністю.

7.2.7. Квадратура Гаусса, чи метод невизначених коефіцієнтів

У розглянутих методах обчислення інтеграла проводилося за значеннями функції $f(x_i)$ у попередньо заданих вузлах. І побудова методів полягала у знаходженні коефіцієнтів c_i квадратурної формули. У методі квадратури Гаусса передбачається вибір таких коефіцієнтів квадратурної формули c_i і значень x_i на інтервалі $[a, b]$, які забезпечують оцінювання інтеграла з найвищою точністю. Ці величини визначаються за умови, що квадратурна формула (7.27) повинна бути точною для всіх $f(x) = x^k$, $k = 0, 1, 2, ..., n$.

Почнемо з найпростішого випадку оптимального вибору двох значень x_0 і x_1 на інтервалі $[-1, 1]$ так, щоб на цьому інтервалі точно оцінювалися інтеграли

від постійної, лінійної, параболічної та кубічної функцій. Запишемо відповідні рівняння:

$$c_0 f(x_0) + c_1 f(x_1) = \int_{-1}^{1} 1\,dx = 2,$$

$$c_0 f(x_0) + c_1 f(x_1) = \int_{-1}^{1} x\,dx = 0,$$

$$c_0 f(x_0) + c_1 f(x_1) = \int_{-1}^{1} x^2\,dx = \frac{2}{3},$$

$$c_0 f(x_0) + c_1 f(x_1) = \int_{-1}^{1} x^3\,dx = 0,$$

звідки

$$c_0 = c_1 = 1,$$

$$x_0 = \frac{-1}{\sqrt{3}} = -0{,}577350629, \quad x_1 = \frac{1}{\sqrt{3}} = 0{,}577350629.$$

Отже, оцінка інтеграла обчислюється за формулою

$$J = f\left(\frac{-1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right). \tag{7.53}$$

Аналогічно виводяться співвідношення для більшої кількості точок на інтервалі $[-1, 1]$, основні параметри яких зведені в табл. 7.6.

Таблиця 7.6. Параметри методу квадратури Гаусса

Кількість точок	Вагові коефіцієнти	Аргументи функції	Локальна похибка
3	$c_0 = 0{,}555555556$	$x_0 = -0{,}774596669$	$\approx f^{(6)}(\xi)$
	$c_1 = 0{,}888888889$	$x_1 = 0{,}0$	
	$c_2 = 0{,}555555556$	$x_2 = 0{,}774596669$	
4	$c_0 = 0{,}347854845$	$x_0 = -0{,}861136312$	$\approx f^{(8)}(\xi)$
	$c_1 = 0{,}652145155$	$x_1 = -0{,}339981044$	
	$c_2 = 0{,}652145155$	$x_2 = 0{,}339981044$	
	$c_3 = 0{,}347854845$	$x_3 = 0{,}861136312$	

Заповнити табл. 7.6 для іншої кількості точок можна за допомогою спеціальних операторів `GaussianQuadratureWeights[n, a, b]` і `GaussianQuadratureError[n, f, a, b]` пакета Mathematica. Наприклад, для $n = 4$, $a = -1$ та $b = 1$ отримаємо дані другого рядка табл. 7.6:

```
In[]:= <<NumericalMath`GaussianQuadrature`
In[]:= GaussianQuadratureWeights[4,-1,1]
Out[]= {{-0.861136, 0.347855}, {-0.339981, 0.652145},
        {0.339981, 0.652145}, {0.861136, 0.347855}}
```

```
In[]:= GaussianQuadratureError[4, f, -1,1]
Out[]= -2.87946*10^-7*f^(8)
```

Похибка методу квадратури Гаусса в загальному випадку визначається виразом

$$r_n = \frac{2^{2n+3}((n+1)!)^4}{(2n+3)((2n+2)!)^3} f^{(2n+2)}(\xi).$$

Метод досить ефективний, якщо функція відома і можуть бути визначені необхідні її значення. Перехід до реальних меж інтегрування здійснюється заміною змінних $x = a_0 + a_1 x_d$. На нижній межі інтервалу $x = a$ і $x_d = -1$, тому

$$a = a_0 + a_1(-1).$$

На верхній межі інтервалу $x = b$ і $x_d = 1$, в результаті

$$b = a_0 + a_1(1).$$

Отже, $a_0 = (b+a)/2$ і $a_1 = (b-a)/2$, тому $x = ((b+a) + (b-a)x_d)/2$.
При цьому $dx = (b-a)/2\, dx_d$.

Приклад 7.10

Оцінимо інтеграл від функції

$$f(x) = 300x^5 - 600x^4 + 500x^3 - 100x^2 + 50x + 0,5,$$

розглянутої в прикладах 7.4, 7.5 і 7.6, в межах $a = 0$ і $b = 0,8$, застосовуючи метод квадратури Гаусса.

Проведемо заміну $x = 0,4 + 0,4x_d$ у поліномі. Перехід до реальних меж інтегрування здійснюється заміною $x = a_0 + a_1 x_d$.

На нижній межі інтервалу $x = a$ і $x_d = -1$, тому

$$a = a_0 + a_1(-1).$$

На верхній межі інтервалу $x = b$ і $x_d = 1$, в результаті

$$b = a_0 + a_1(1).$$

Отже,

```
In[]:= a = 0; b = 0.8; z = ((b + a) + (b - a)*xd)/2
```

$$\text{Out[]}= \frac{1}{2}(0.8 + 0.8\ x_d)$$

Перетворений поліном набуває вигляду

```
In[]:= fl[z_]:= f[x]/. x → z; N[fl[z]]
Out[]=  .5 + 25. (0.8 + 0.8 xd) - 25. (0.8 + 0.8 xd)^2 + 62.5 (0.8 + 0.8 xd)^3 -
      37.5 (0.8 + 0.8 xd)^4 + 9.375 (0.8 + 0.8 xd)^5
```

Запишемо його в більш наочній формі:

$$f_1(x_d) = 300(0,4 + 0,4x_d)^5 - 600(0,4 + 0,4x_d)^4 +$$
$$+500(0,4 + 0,4x_d)^3 - 100(0,4 + 0,4x_d)^2 + 50(0,4 + 0,4x_d) + 0,5.$$

Вибравши в табл. 7.6 варіант методу з трьома точками, обчислимо необхідні значення полінома $f_1(x_d)$ і оцінимо шуканий інтеграл:

$$J = \int_0^{0,8} f(x)\,dx = 0,4\int_{-1}^{+1} f_1(x_d)\,dx_d;$$

$$J = 0,4[0,555555556 f_1(-0,774596669) + 0,888888889 f_1(0) + 0,555555556 f_1(0,774596669)] =$$
$$= 0,4[2,5132 + 21,5218 + 36,7624] = 0,4 \cdot 60,7974 = 24,319.$$

Неважко перевірити, що результат отриманий майже точно, оскільки значення цього ж інтеграла, отримане за допомогою пакета Mathematica, таке:

```
In[]:= G[z] = 0.4*Integrate[f1[z], {xd, -1, 1}]
Out[]= 24.3189
```

7.3. Засоби пакета Mathematica для чисельного диференціювання та інтегрування функцій

Для чисельного диференціювання в пакеті Mathematica використовується оператор `N[D[f, x]]`, який дозволяє апроксимувати значення похідної $(df(x)/dx)$. Для оцінювання похідних вищих порядків оператор модифікується до вигляду `N[D[f, {x,n}]]`. Його застосування було проілюстровано в прикладах 7.1 і 7.2.

У пакеті передбачений також оператор `NIntegrate [f, {x, xmin, xmax}]`, за допомогою якого чисельно апроксимується значення інтеграла $\int_{x_{min}}^{x_{max}} f(x)\,dx$. Як базовий використовується адаптивний метод квадратури Гаусса, коли кількість точок Гаусса на інтервалі інтегрування вибирається автоматично, виходячи з бажаної похибки апроксимації. Так, для рівняння з прикладу 7.9 отримаємо:

```
In[]:= <<NumericalMath`GaussianQuadrature`
In[]:= NIntegrate[300 x^5 - 600 x^4 + 500 x^3 - 100 x^2 + 50 x + 0.5, {x, 0, 0.8}]
Out[]= 24.3189
```

Оператор `NIntegrate` має багато параметрів і допускає, зокрема, перехід до рекурентної формули трапецій (7.41) за допомогою параметра `Method → Trapezoidal`:

```
In[]:= NIntegrate[300 x^5 - 600 x^4 + 500 x^3 - 100 x^2 + 50 x + 0.5, {x, 0, 0.8},
       Method → Trapezoidal]
Out[]= NIntegrate::ncvi: NIntegrate failed to converge to prescribed accuracy
           after 7 iterated refinements in x in the interval {{x, 0., 0.8}}.
       24.319
```

Щоб покращити результат, збільшимо кількість рекурсій з 6 (значення за замовчуванням) до 20:

```
In[]:= NIntegrate[300 x^5 - 600 x^4 + 500 x^3 - 100 x^2 + 50 x + 0.5, {x, 0, 0.8},
       Method → Trapezoidal, MaxRecursion → 20]
Out[]= 24.3189
```

Оператор NIntegrate застосовується і для обчислення подвійного інтеграла $\int_{x_{min}}^{x_{max}} \int_{y_{min}}^{y_{max}} f(x, y)\,dx\,dy$, при цьому його формат змінюється в такий спосіб:

```
NIntegrate [f, {x, xmin, xmax}, {y, ymin, ymax}]].
```

Висновки

1. Методи апроксимації похідних та інтегралів застосовуються у випадках, коли неперервна функція настільки складна, що безпосереднє використання аналітичних методів утруднене чи навіть неможливе. Чисельний підхід використовують і в тому випадку, коли сама функція задана в дискретній формі масивом своїх значень для обраних значень аргументів.
2. Похідні звичайно апроксимуються розділеними різницями, і відповідні формули залежно від числа врахованих членів розкладу в ряд Тейлора відрізняються похибкою апроксимації, яка може змінюватися від $O(h)$ до $O(h^{n-1})$, де h — крок дискретизації аргументу.
3. Розрізняють прямі несиметричні формули диференціювання вперед і обернені несиметричні формули диференціювання назад у залежності від розташування точки відліку — ліворуч чи праворуч у ряді значень аргументів $x_i, x_{i-1}, x_{i-2}, \ldots$ чи $x_i, x_{i+1}, x_{i+2}, \ldots$, для яких обчислюються значення функції, що беруть участь в апроксимації похідних.
4. Найбільшу точність $O(h^2)$ чи $O(h^4)$ забезпечують симетричні формули диференціювання з вибором точки відліку в середині діапазону значень аргументів на інтервалі $[a, b]$, що використовуються для обчислення відповідних значень функції.
5. Явні методи чисельного розв'язання звичайних диференціальних рівнянь і диференціальних рівнянь із частинними похідними, що будуть розглянуті в розділах 8–12, базуються на використанні переважно симетричних і прямих несиметричних формул апроксимації похідних, а неявні методи розв'язання жорстких диференціальних рівнянь (розділ 10) — на використанні обернених формул диференціювання назад.
6. Залежно від порядку використаного апроксимуючого полінома розрізняють такі чисельні методи інтегрування: прямокутників із похибками $O(h)$ і $O(h^2)$, трапецій із похибкою $O(h^2)$, Ньютона з похибкою $O(h^4)$. Формули Сімпсона (перша і друга) мають той же четвертий порядок точності, хоча використовують різне число відліків (3 і 4 відповідно). Доцільність застосування кожної з них визначається парністю чи непарністю кількості наявних відліків функції.
7. Найбільшу точність оцінки інтеграла забезпечує метод квадратури Гаусса, однак він передбачає вибір спеціально нерівномірно розташованих точок відліку.

8. Підвищення точності чисельного інтегрування можна досягти або багаторазовим застосуванням формул інтегрування і розбиттям інтервалу інтегрування на підінтервали з використанням великого масиву значень функції, або застосуванням екстраполяції Річардсона.
9. У системах моделювання інтегрування ведеться зі змінним кроком, коли застосовуються формули чисельного диференціювання та інтегрування з нерівномірно розподіленими точками відліку значень функції.

Контрольні запитання та завдання

1. Посадка винищувача на палубу авіаносця характеризується даними, наведеними в таблиці:

t, с	0	0,51	1,03	1,74	2,36	3,34	3,82
x, м	154	186	209	250	262	272	274

де x — відстань від кінця палуби.

Визначте зміну швидкості й прискорення винищувача під час посадки.

2. На основі таблиці значень функції $\cos(x)$ для $x = 0, \pi/6, \pi/4, \pi/2$ оцініть наближене значення похідної для $x = 171°$.
3. Обчисліть оцінку невизначеного інтеграла за формулою центральних прямокутників

$$\int_1^{\infty} \frac{e^{-x}}{x} dx$$

із точністю до $\varepsilon = 10^{-2}$.

4. Обчисліть оцінку визначеного інтеграла за формулами Сімпсона і трапецій

$$\int_{\pi/4}^{\pi/2} \frac{\sin x}{x} dx$$

із точністю до $\varepsilon = 10^{-2}$.

5. Обчисліть оцінку невизначеного інтеграла за формулою Ньютона

$$\int_2^{\infty} \frac{e^{-2x}}{1 + x^3} dx$$

із точністю до $\varepsilon = 10^{-2}$.

Розділ 8

Розв'язання задачі Коші для звичайних диференціальних рівнянь

- Методи Ейлера
- Апріорна оцінка похибки методом Ейлера
- Методи Рунге–Кутта
- Апостеріорна оцінка похибки методів Рунге–Кутта
- Вбудовані методи Рунге–Кутта–Фельберга
- Розв'язання систем диференціальних рівнянь
- Стійкість методів Рунге–Кутта

На практиці для розв'язання звичайних диференціальних рівнянь (або систем) застосовують здебільшого чисельні методи [21, 36, 40], які дають можливість знайти наближений розв'язок у вигляді деякої таблиці.

8.1. Основні поняття

Звичайне диференціальне рівняння (ЗДР) — це рівняння виду

$$F(t, y, y', ..., y^{(m)}) = 0, \tag{8.1}$$

яке зв'язує незалежну змінну t, шукану функцію $y(t)$ та її похідні включно до m-го порядку. Функція $y(t)$, яка задовольняє рівняння (8.1), називається розв'язком цього рівняння, а задача знаходження розв'язків диференціального рівняння — задачею інтегрування диференціального рівняння.

Порядок ЗДР — це порядок старшої похідної від шуканої функції. Наприклад, рівняння

$$y'' + t^3 y' + y^3 = 0$$

є ЗДР другого порядку.

ЗДР *лінійне*, якщо воно має такий вигляд:

$$a_k(t)y^{(m)} + a_{k-1}(t)y^{(m-1)} + \ldots + a_1(t)y' + a_0(t)y = f(t). \qquad (8.2)$$

Наприклад, ЗДР $y'' + t^3y' + ty = e^{-t}$ — лінійне, а $y'' + t^3y' + y^3 = 0$ — нелінійне рівняння другого порядку.

Загальним інтегралом рівняння (8.1) називають функцію $\Phi(t, y, C_1, \ldots, C_m)$, що зв'язує незалежну змінну t, шукану функцію $y(t)$ і m констант інтегрування за допомогою рівняння:

$$\Phi(t, y, C_1, \ldots, C_m) = 0, \qquad (8.3)$$

тобто розв'язок $y(t)$ входить у (8.3) неявно, причому кількість констант інтегрування дорівнює порядку ЗДР.

Загальним розв'язком ЗДР називається функція

$$y(t) = \varphi(t, C_1, \ldots, C_m), \qquad (8.4)$$

яка з'єднує незалежну змінну t і m констант інтегрування C_i, тобто розв'язок $y(t)$ визначається явно.

Для знаходження констант інтегрування C_i задаються додаткові умови, кількість яких дорівнює порядку ЗДР. Якщо в додаткові умови підставити функцію (8.3) і розв'язати отриману систему відносно C_i, а знайдені значення C_i потім підставити в (8.4), то отримаємо *частковий інтеграл* $\psi(t, y(t)) = 0$. Аналогічні процедури з загальним розв'язком (8.4) дають *частковий розв'язок* $y(t) = \psi(t)$.

У випадку, коли всі додаткові умови задаються в одній точці t_0, сукупність ЗДР і додаткових умов називають *задачею Коші*, а додаткові умови — *початковими* умовами.

Якщо ж додаткові умови задаються більш ніж в одній точці, то сукупність ЗДР і додаткових умов називають *крайовою задачею Коші* для ЗДР, а додаткові умови — граничними умовами.

У загальному вигляді задачу Коші для ЗДР (8.1) можна записати таким чином:

$$\begin{gathered} F(t, y, y', \ldots, y^{(m)}) = 0, \\ y(t_0) = y_0, \\ y'(t_0) = y_0', \\ \ldots \quad \ldots \quad \ldots \\ y^{(m-1)}(t_0) = y_0^{(m-1)}, \end{gathered}$$

де $y_0, y_0', \ldots, y_0^{(m-1)}$ — задані числа. Порядок старшої похідної в початкових умовах дорівнює $(m-1)$ і не перевищує m, де m — порядок ЗДР.

Коли старша похідна в ЗДР визначається як функція t, $y(t)$ і похідних більш низького порядку, то рівняння вважають розв'язаним *щодо старшої похідної* і воно записується в такому вигляді: $y^{(m)} = f(t, y, y', \ldots, y^{(m-1)})$.

Система рівнянь першого порядку виду:

$$\begin{aligned} y_1' &= f_1(t, y, y_1, ..., y_m), \\ y_2' &= f_2(t, y, y_1, ..., y_m), \\ &\dots \quad \dots \quad \dots \quad \dots \quad \dots \\ y_m' &= f_m(t, y, y_1, ..., y_m), \end{aligned} \tag{8.5}$$

тобто рівнянь, розв'язаних явно щодо похідних від шуканих функцій, називається системою, яка має нормальну форму Коші. Часто систему (8.5) записують у скороченій і більш зручній для виконання обчислень векторній формі. Введемо позначення:

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{bmatrix}, \quad \mathbf{Y}' = \begin{bmatrix} y_1' \\ y_2' \\ \dots \\ y_m' \end{bmatrix}, \quad \mathbf{F}(t, \mathbf{Y}) = \begin{bmatrix} f_1(t, y_1, ..., y_m) \\ f_2(t, y_1, ..., y_m) \\ \dots \\ f_m(t, y_1, ..., y_m) \end{bmatrix}, \quad \mathbf{Y}_0 = \begin{bmatrix} y_1^0 \\ y_2^0 \\ \dots \\ y_m^0 \end{bmatrix} \tag{8.6}$$

Тоді задача Коші для системи (8.5) набуде вигляду

$$\mathbf{Y}' = \mathbf{F}(t, \mathbf{Y}), \quad \mathbf{Y}(t_0) = \mathbf{Y}_0. \tag{8.7}$$

Задачу Коші для ЗДР m-го порядку $y^{(m)} = f(t, y, y', ..., y^{(m-1)})$ заміною $y'(t) = x_1(t)$, $y''(t) = x_2(t)$, ..., $y^{(m-1)}(t) = x_{m-1}(t)$ можна звести до задачі Коші для нормальної системи ЗДР:

$$\begin{aligned} y'(t) &= x_1(t), \\ x_1'(t) &= x_2(t), \\ &\dots \quad \dots \quad \dots \\ x_{m-2}'(t) &= x_{m-1}(t), \\ x_{m-1}'(t) &= f(t, y, x_1, x_2, ..., x_{m-1}), \\ y(t_0) &= y_0, \\ x_1(t_0) &= y_0', \dots, x_{m-1}(t_0) = y_0^{(m-1)}. \end{aligned}$$

Наведемо умови, які гарантують існування і єдиність розв'язку задачі Коші (8.7) [29]. Припустимо, що функції f_i, $i = 1, 2, ..., m$ в (8.6) неперервні по всіх аргументах у замкнутій області

$$D = \{|t| \leqslant a, \quad |y_i - y_i^0| \leqslant b, \quad i = 1, 2, ..., m\}.$$

Неперервністю функцій f_i зумовлюється їх обмеженість, тобто існування константи $M > 0$, такої, що всюди в D виконуються нерівності $|f_i| \leqslant M$, $i = 1, 2, ..., m$.

Припустимо також, що в області D функції f_i задовольняється умова Ліпшиця за аргументами $y_1, y_2, ..., y_m$ тобто

$$|f_i(t, y_{1,1}, y_{1,2}, ..., y_{1,m}) - f_i(t, y_{2,1}, y_{2,2}, ..., y_{2,m})| \leqslant$$
$$\leqslant L[|y_{1,1} - y_{2,1}| + |y_{1,2} - y_{2,2}| + ... + |y_{1,m} - y_{2,m}|]$$

для будь-яких точок $(t, y_{1,1}, y_{1,2}, ..., y_{1,m})$ і $(t, y_{2,1}, y_{2,2}, ..., y_{2,m})$ області D.

Якщо сформульовані вище припущення вірні, то існує єдиний розв'язок $\mathbf{Y} = [y_1(t), y_2(t), ..., y_m(t)]^T$ системи (8.7), який визначений за умови, що $|t_0| \leqslant t = \min(a, b/M)$, і набуває, коли $t = t_0$, заданих початкових значень $\mathbf{Y}_0$. Вивчаючи чисельні методи розв'язання задачі Коші, припускаємо, що існує її єдиний і досить гладкий розв'язок.

8.2. Методи Ейлера і Рунге–Кутта

Чисельні методи розв'язання задачі Коші можна розділити на дві групи:

- *Однокрокові методи* — для знаходження розв'язку в деякій точці відрізка t_{n+1} використовується інформація про розв'язок тільки на відрізку (t_{n+1}, t_n). До таких методів належать методи Ейлера, Хейна, Рунге–Кутта.
- *Багатокрокові методи* — для знаходження розв'язку в деякій точці t_{n+1} використовується інформація про розв'язок на декількох попередніх кроках $(t_{n+1}, t_n), (t_n, t_{n-1}), ..., (t_{n-1}, t_{n+1-m})$. До таких методів належать методи Адамса–Башфорта, Адамса–Мултона, Гіра–Брайтона.

У зв'язку із широким застосуванням задачі Коші в багатьох галузях науки і техніки для її розв'язання розроблено велику кількість як аналітичних (де це можливо), так і наближених чисельних методів. Останні істотно відрізняються за необхідним обсягом обчислень, за стійкістю розв'язку до вибору і зміни кроку τ, за реалізованою точністю розв'язку та ін.

Щоб краще довести до читача основні ідеї обчислювальних методів розв'язання задач із початковими умовами, розглянемо спочатку звичайне диференціальне рівняння першого порядку:

$$y' = f(t, y), \quad y(t_0) = y_0. \tag{8.8}$$

Введемо для змінної t рівномірну сітку з кроком τ, тобто вкажемо множину точок $\{t_n = n\tau, n = 0, 1, 2, ...\}$, яку будемо називати сіткою. Позначимо через $y(t_n)$ точний розв'язок задачі (8.8), а через u_n — наближений розв'язок в момент часу t_n. Відмітимо, що наближений розв'язок є сітковою функцією. Похибкою наближеного методу на кроці n називається різниця $\varepsilon_n = u_n - y_n$.

Під час використання наближених методів основною вимогою є збіжність наближеного розв'язку u_n до точного розв'язку $y(t_n)$ задачі (8.8). Наведемо визначення збіжності, коли $\tau \to 0$. Різницевий метод збігається в точці t_n, якщо $|y(t_n) - u_n| \to 0$ і $\tau \to 0$. Метод збігається на відрізку $[t_0, T]$, якщо він збігається в кожній точці $t \in [t_0, T]$.

Метод має p-й порядок точності, якщо існує число p, таке, що

$$|y(t_n) - u_n| = O(\tau^p) \quad \text{для} \quad \tau \to 0.$$

8.2.1. Метод Ейлера

Метод Ейлера є найпростішим методом розв'язання задачі Коші, але має невисоку точність, тому на практиці його використовують досить рідко. Однак на прикладі методу Ейлера зручно пояснити існуючі способи дослідження однокрокових чисельних методів.

Розкладемо точний розв'язок $y_{n+1} = y(t_n + \tau)$ задачі (8.8) в околі вузла сітки t_n у ряд Тейлора, отримаємо:

$$y(t_n + \tau) = y(t_n) + y'(t_n)\tau + \frac{1}{2}y''(t_n)\tau^2 + O(\tau^3). \tag{8.9}$$

Залишимо у формулі (8.9) тільки два перших члени, тоді замість точного розв'язку $y(t_{n+1})$ отримаємо його наближене значення u_{n+1}, що знаходять за формулою:

$$u_{n+1} = u_n + \tau f(t_n, u_n), \quad u_0 = y_0, \quad n = 0, 1, 2, \ldots . \tag{8.10}$$

Оскільки значення y_0 відоме, то, послідовно застосовуючи формулу (8.10), знаходимо наближений розв'язок у всіх наступних точках.

Формулу (8.10) можна отримати, використовуючи графік (рис. 8.1). Якщо значення розв'язку $y(t_0)$ у точці t_0 відоме з початкової умови (8.10), можна обчислити значення похідної $y'(t_0) = f(t_0, y_0)$ у цій точці. Запишемо рівняння дотичної до кривої точного розв'язку $y(t)$ у точці (t_0, y_0):

$$y(t) = y_0 + f(t_0, y_0)(t - t_0).$$

За досить малого кроку τ ордината $u_1 = y_0 + \tau f(t_0, y_0)$ цієї дотичної буде мало відрізнятися від ординати $y(t_1)$ розв'язку задачі (8.8). Отже, точка (t_1, u_1) може бути розв'язком і може бути прийнята за нову початкову точку. Через цю точку знову проведемо дотичну і т. д. Нарешті отримаємо формулу (8.8).

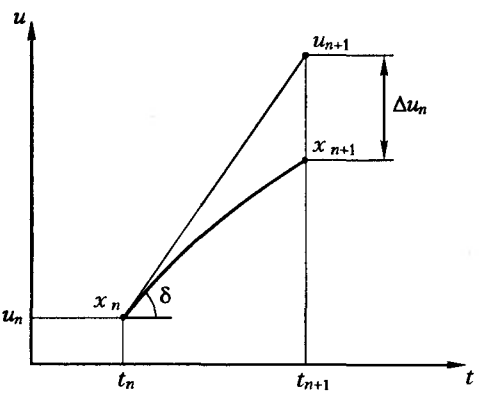


Рис. 8.1. Геометрична інтерпретація обчислень за методом Ейлера

8.2.2. Апріорна оцінка похибки

Припустимо, що задачу (8.8) необхідно розв'язати на відрізку $t \in [t_0, T]$. Знайдемо похибку наближеного розв'язку, яку позначимо через

$$\varepsilon_{n+1} = u_{n+1} - y(t_{n+1}), \quad n = 0, 1, 2, \ldots . \tag{8.11}$$

Віднімемо від рівності (8.10) рівність (8.9), отримаємо:

$$\varepsilon_{n+1} = \varepsilon_n + \tau(f(t_n, u_n) - f(t_n, y(t_n))) - \frac{\tau^2}{2} y''(\varsigma), \quad t_n < \varsigma < t_{n+1}. \tag{8.12}$$

Формула (8.12) визначає залежність між похибками наближеного розв'язку у двох сусідніх вузлах сітки. Очевидно, що багаторазове застосування формули (8.10) призведе до накопичення похибок обчислень ε_{n+1} (8.12). Отримувана за n кроків похибка називається *глобальною* похибкою чи накопиченою за певну кількість кроків. Оцінимо цю похибку, використовуючи формулу (8.12).

Спочатку отримаємо формулу для локальної (або покрокової) похибки. За умови, що $u_n = y(t_n)$, маємо $f(t_n, y(t_n)) = f(t_n, u_n)$ і з виразу (8.11) отримуємо формулу для локальної похибки

$$\eta_{n+1} = -\frac{\tau^2}{2} y''(\varsigma), \quad t_n < \varsigma < t_{n+1}. \tag{8.13}$$

Звідси випливає, що локальна похибка має другий порядок відносно кроку τ. Далі для нас буде важливо, що локальна похибка методу

$$\eta_{n+1} = -\frac{\tau^2}{2} y''(\varsigma) = -\frac{\tau^2}{2} y''(t_n + \theta\tau) = -\frac{\tau^2}{2} y''(t_n) + O(\tau^3), \tag{8.14}$$

де $0 < \theta < 1$, має головний член виду

$$\gamma_{n+1} = -\frac{\tau^2}{2} y''(t_n), \tag{8.15}$$

тобто головний член асимптотичного розкладання локальної похибки (8.16) для методу Ейлера (8.10) має *другий порядок відносно* τ. Для похідної $y''(t)$ від точного розв'язку задачі Коші (8.8) можна отримати такий вираз:

$$y''(t) = (f(t, y(t)))' = \frac{\partial f(t, y)}{\partial t} + f(t, y)\frac{\partial f(t, y)}{\partial y}.$$

Підставивши його у формулу (8.15), отримаємо головний член локальної похибки для методу Ейлера:

$$\gamma_{n+1} = -\frac{\tau^2}{2}\left(\frac{\partial f(t_n, y(t_n))}{\partial t} + f(t_n, y_n)\frac{\partial f(t_n, y(t_n))}{\partial t}\right). \tag{8.16}$$

Припустимо, що права частина $f(t, y(t))$ та її частинні похідні першого порядку за t і $y(t)$ рівномірно обмежені в області $D\{t_0 \leqslant t \leqslant T, -\infty \leqslant y \leqslant +\infty\}$. Тоді головний член локальної похибки (8.15) можна оцінити за формулою (8.16). Нехай

$$M_2 = \sup\left|\frac{d^2y(t)}{dt^2}\right|, \quad t \in [t_0, T].$$

Позначивши через L постійну Ліпшиця функції $f(t, y(t))$ за змінною $y(t)$, матимемо для глобальної похибки (8.12) таку оцінку:

$$|\varepsilon_{n+1}| \leqslant |\varepsilon_n|(1 + L\tau) + M_2\frac{\tau^2}{2}.$$

Виконуючи рекурентні перетворення для $\varepsilon_n, \varepsilon_{n-1}, ..., \varepsilon_0$, отримаємо:

$$|\varepsilon_{n+1}| \leqslant |\varepsilon_0|(1 + L\tau)^{n+1} + M_2\frac{\tau^2}{2}(1 + (1 + L\tau) + ... + (1 + L\tau)^n).$$

Оскільки $\varepsilon_0 = 0$, остання нерівність спрощується:

$$|\varepsilon_{n+1}| \leqslant M_2\frac{\tau}{2L}((1 + L\tau)^{n+1} - 1).$$

Враховуючи нерівність $e^{L\tau} > 1 + L\tau$, спростимо цей вираз до вигляду

$$|\varepsilon_{n+1}| \leqslant M_2\frac{\tau}{2L}(e^{L\tau(n+1)} - 1) \tag{8.17}$$

і для $\tau(n + 1) = T$ отримаємо остаточно:

$$|\varepsilon_{n+1}| \leqslant M_2\frac{\tau}{2L}e^{LT}. \tag{8.18}$$

Таким чином, якщо виконуються наведені вище умови, метод Ейлера збігається на будь-якому кінцевому відрізку $t \in [t_0, T]$ і є *методом першого порядку точності*.

Формула (8.18) може в принципі застосовуватися для вибору кроку обчислень τ за заданої похибки обчислень ε_{n+1} з урахуванням особливостей диференціального рівняння (значень L і M_2).

Однак знайдені оцінки для кроку, на обчислення яких витрачають багато часу, можуть виявитися неефективними. Тому на практиці частіше користуються апостеріорними оцінками похибки і кроку обчислень, які будуть розглянуті нижче.

8.2.3. Поліпшення точності методу Ейлера застосуванням екстраполяції Річардсона

В інженерній практиці рекомендація зменшувати крок обчислень ($\tau \to 0$) для підвищення точності розрахунків у разі застосування методу Ейлера не завжди може бути прийнятною. Однак є методи, які дозволяють істотно підвищити точність розрахунків у разі збереження досить великих значень кроку τ. Їх застосування обмежується лише можливістю втрати стійкості обчислень. До таких методів належить процедура екстраполяції Річардсона (1.4), яку ми вже використовували в розділі 7. Нагадаємо: коли застосовується деякий алгоритм із відомою залежністю глобальної (або локальної) похибки від кроку обчислень, достатньо розв'язати задачу з різними кроками τ і $q\tau$, щоб отримати потім більш точний розв'язок a_0 за знайденими наближеннями $F(\tau)$ і $F(q\tau)$:

$$\begin{cases} F(\tau) = a_0 + a_1\tau^p + 0(\tau^r), \\ F(q\tau) = a_0 + a_1(q\tau)^p + 0(q\tau)^r, \end{cases}$$

$$a_0 = F(\tau) + \frac{F(\tau) - F(q\tau)}{q^p - 1} + O(\tau^r). \tag{8.19}$$

До того ж процедуру екстраполяції Річардсона можна узагальнити, як це було зроблено в рекурентних формулах Ромберга (підрозділ 7.9), для розкладу в ряд глобальної похибки типу:

$$F(\tau) = a_0 + a_1\tau^{p_1} + a_2\tau^{p_2} + \dots,$$

де $p_1 < p_2 < p_3 < \dots$.

Тоді за наближеннями $F(\tau), F(q\tau), F(q^2\tau), \dots, F(q^k\tau)$ рекурсивно обчислюється оцінка для показника степеня p_k:

$$F_{k+1}(\tau) = F_k(\tau) + \frac{F_k(\tau) - F_k(q\tau)}{q^{p_k} - 1}. \tag{8.20}$$

Якщо $|F_{k+1}(\tau) - F_k(\tau)| \leqslant \varepsilon$, де ε — задана похибка обчислень, то $F_{k+1}(\tau)$ є розв'язком. Відповідна схема обчислень набуває такого вигляду:

$\frac{\Delta}{\tau^{p_1} - 1}$	$\frac{\Delta}{\tau^{p_2} - 1}$	$\frac{\Delta}{\tau^{p_3} - 1}$	
F_{01}			
F_{11}	F_{12}		
F_{21}	F_{22}	F_{23}	
F_{31}	F_{32}	F_{33}	F_{24}

Оскільки для методу Ейлера

$$F(\tau) = a_0 + a_1\tau + a_2\tau^2 + a_3\tau^3 + a_4\tau^4 + a_5\tau^5 + \dots, \qquad (8.21)$$

то для $q = 2$, перший рядок схеми обчислень стає рівним $\Delta/1, \Delta/3, \Delta/7, \Delta/15, \Delta/31, \dots$.

Приклад 8.1

Розв'яжемо методом Ейлера з екстраполяцією Річардсона рівняння: $y' = t^2 - y^2$, $t_0 = 0$, $y_0 = 1$. Потрібно з кроком $\tau = 0,1$ знайти значення $y(0, 2)$.

Скориставшись формулою (8.10), послідовно обчислюємо значення

$$t_1 = \tau + t_0 = 0,1, \quad u_1 = 1 + 0,1(0^2 - 1^2) = 0,9,$$
$$t_2 = \tau + t_1 = 0,2, \quad u_2 = 0,9 + 0,1(0,1^2 - 0,9^2) = 0,82.$$

Отже $u(0,2) = 0,82$.

Тепер проведемо додатковий розрахунок із кроком $\tau = 0,2$:

$$t_1 = \tau + t_0 = 0,2, \quad u_1 = 1 + 0,2(0^2 - 1^2) = 0,8, \quad \text{або} \quad u(0,2) = 0,8.$$

Застосовуючи формулу екстраполяції Річардсона (8.20), обчислюємо:

$$u_{12} = 0,82 + \frac{1}{2-1}(0,82 - 0,8) = 0,84.$$

Приклад 8.2

У разі розв'язання методом Ейлера лінійного диференціального рівняння $y' = -y$ з початковими умовами $y(0) = 1$, кроком $\tau = 0,25$ і коефіцієнтом $q = 1/2$ отримаємо для моменту $t = 1$ чотири початкові розв'язки $F_{i1} = y(1)$ (перший стовпець табл. 8.1). Дані, отримані в результаті їх рекурсивної обробки з використанням формули (8.20), розміщуємо в інших стовпцях цієї таблиці.

Таблиця 8.1. Результати обчислень за рекурентною формулою Річардсона

$y(1)$	$(\Delta/1)\cdot 10^3$		$(\Delta/3)\cdot 10^3$		$(\Delta/7)\cdot 10^3$	
$F_{01} = 0,316406$						
	27,2027					
$F_{11} = 0,343609$		$F_{12} = 0,370812$				
	12,4652		−0,7574			
$F_{21} = 0,356074$		$F_{22} = 0,368539$		$F_{23} = 0,367781$		
	5,9812		−0,1676		0,0124	
$F_{31} = 0,362055$		$F_{32} = 0,368036$		$F_{33} = 0,367868$		$F_{34} = 0,36788$

Можна прийняти за розв'язок значення $F_{34} = 0,36788$, тому що $|F_{33} - F_{34}| = 0,000012$ менше локальної похибки $1/2 \cdot 10^{-6}$. Точне ж значення розв'язку рівняння дорівнює $y(1) = e^{-1} = 0,367879$. Екстраполяція за Річардсоном може бути *пасивною*, як у нашому прикладі, коли обробляється вже отриманий масив даних, чи *активною*, коли уточнені значення функції використовуються в процесі розрахунку.

8.3. Методи Рунге–Кутта

Методи Рунге–Кутта базуються на використанні скороченого ряду Тейлора. Розкладемо точний розв'язок $y_{n+1} = y(t_n + \tau)$ задачі (8.8) в околі вузла сітки t_n у ряд Тейлора і обмежимо ряд членами p-го порядку:

$$y(t_{n+1}) = y(t_n) + y'(t_n)\tau + \frac{1}{2}y''(t_n)\tau^2 + \ldots + \frac{1}{p!}\tau^p y^{(p)}(t_n) + O(\tau^{p+1}). \tag{8.22}$$

Основна ідея побудови явних методів Рунге–Кутта p-го порядку полягає в заміні частини ряду Тейлора

$$T_P(y_n, t_n, \tau) = y'(t_n)\tau + \frac{1}{2}y''(t_n)\tau^2 + \ldots + \frac{1}{p!}\tau^p y^{(p)}(t_n)$$

функцією $\varphi_p(y_n, t_n, \tau)$, яка містить значення правої частини рівняння $f(t, y)$ у проміжних точках відрізку $[t_n, t_{n+1}]$ і наближає $T_p(y_n, t_n, \tau)$ з точністю до p-го порядку

$$\left|\varphi_P(y_i, t_i, \tau) - T_P(y_i, t_i, \tau)\right| < C\tau^{P+1},$$

де C — константа, яка не залежить від кроку τ.

У загальному випадку методи Рунге–Кутта записуються у вигляді:

$$u_{n+1} = u_n + \tau\varphi_p(u_n, t_n, \tau), \tag{8.23}$$

де

$$\varphi_p(y_n, t_n, \tau) = b_1k_1 + b_2k_2 + \ldots + b_pk_p, \tag{8.24}$$

b_i — константи, а k_i мають такий рекурсивний вигляд:

$$\begin{aligned}
k_1 &= f(t_n, u_n),\\
k_2 &= f(t_n + c_2\tau, \quad u_n + a_{21}k_1\tau),\\
k_3 &= f(t_n + c_3\tau, \quad u_n + a_{31}k_1\tau + a_{32}k_2\tau),\\
&\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots\\
k_p &= f(t_n + c_p\tau, \quad u_n + a_{p,1}k_1\tau + a_{p,2}k_2\tau + \ldots + a_{p,p-1}k_{p-1}\tau).
\end{aligned} \tag{8.25}$$

Методи, визначені формулами (8.23)–(8.25), називаються *явними* методами Рунге–Кутта, оскільки в них наступні наближені значення знаходять прямим розрахунком за формулами (8.24) і (8.25). Порядок точності методів Рунге–Кутта залежить від кількості членів у виразі (8.24) і значень констант b_i, c_i, a_{ij}. Зупинимось спочатку на методах другого порядку точності.

8.3.1. Явний метод Рунге–Кутта другого порядку точності

Розглянемо метод другого порядку, для якого

$$u_{n+1} = u_n + (b_1 k_1 + b_2 k_2)\tau, \tag{8.26}$$

де $k_1 = f(t_n, u_n)$, $k_2 = (t_n + c_2\tau, u_n + a_{21}k_1\tau)$.

Коефіцієнти b_1, b_2, c_1, a_{21} знаходять, прирівнюючи рівняння (8.26) і скорочений ряд Тейлора y_{n+1} (8.24) з урахуванням члена з другої похідної, наведеного раніше в (8.16):

$$\begin{aligned} u_{n+1} &= u_n + \tau f(t_n, u_n) + \frac{\tau^2}{2} f'(t_n, u_n) = \\ &= u_n + \tau f(t_n, u_n) + \frac{\tau^2}{2}\left[\frac{\partial f}{\partial t} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial t}\right]. \end{aligned} \tag{8.27}$$

Для цього спочатку розкладають k_2 як функцію двох змінних у ряд Тейлора:

$$f(t_n + c_2\tau, u_n + a_{21}k_1\tau) = f(t_n, u_n) + c_2\tau\frac{\partial f}{\partial t} + a_{21}k_1\tau\frac{\partial f}{\partial y} + o(\tau^2). \tag{8.28}$$

Підстановкою отриманого виразу в рівняння (8.23) з урахуванням значення для k_1 знаходимо:

$$u_{n+1} = u_n + b_1\tau f(t_n, u_n) + b_2\tau f(t_n, u_n) + b_2c_2\tau^2\frac{\partial f}{\partial y} + c_2a_{21}\tau^2 f(t_i, u_i)\frac{\partial f}{\partial y} + O(\tau^3).$$

Після зведення подібних членів отримуємо:

$$\begin{aligned} u_{n+1} &= u_n + [b_1 f(t_n, u_n) + b_2 f(t_n, u_n)]\tau + \\ &+ \left[b_2c_2\frac{\partial f}{\partial y} + c_2a_{21}f(t_i, u_i)\frac{\partial f}{\partial y}\right]\tau^2 + O(\tau^3). \end{aligned} \tag{8.29}$$

Порівнюючи аналогічні члени в рівняннях (8.27) і (8.29), бачимо, що дані рівняння будуть еквівалентними у разі виконання таких умов:

$$\begin{aligned} b_1 + b_2 &= 1, \\ b_2c_2 = \frac{1}{2}, &\quad b_2a_{21} = \frac{1}{2}. \end{aligned} \tag{8.30}$$

Система трьох рівнянь містить чотири невідомих, тому не має єдиного розв'язку. Однак, задаючи апріорне значення однієї з констант, ми можемо визначити інші. Тому існує ціла множина методів Рунге–Кутта другого порядку.

Припустимо, що ми задаємо значення константи b_2, тоді з системи (8.30) маємо:

$$\left.\begin{aligned} b_1 &= 1 - b_2, \\ c_2 = a_{21} &= \frac{1}{2b_2}. \end{aligned}\right\} \tag{8.31}$$

Найбільш відомі методи Рунге–Кутта другого порядку наведені нижче.

Метод Хейна

Даний метод реалізується, якщо $b_2 = 1/2$. Тоді відповідно до рівнянь (8.30) і (8.26):

$$u_{n+1} = u_n + \frac{1}{2}(k_1 + k_2)\tau, \tag{8.32}$$

де $k_1 = f(t_n, u_n)$, $k_2 = f(t_n + (1/2)\tau, u_n + (1/2)\tau k_1)$. Як бачимо, k_1 відповідає значенню похідної (нахилу дотичної до кривої розв'язку) на початку інтервалу $[t_n, t_{n+1}]$, а k_2 — значення цієї похідної наприкінці інтервалу.

Метод полігону

Цей метод реалізується, якщо $b_2 = 1$. Тоді згідно з рівнянням (8.29) $b_1 = 0$, $c_2 = a_{21} = 1/2$ формула обчислень набуває такого вигляду:

$$u_{n+1} = u_n + k_2\tau, \tag{8.33}$$

де $k_1 = f(t_n, u_n)$, $k_2 = f(t_n + (1/2)\tau, u_n + (1/2)\tau k_1)$. Цей же метод можна реалізувати трохи інакше, а саме: спочатку обчислюється методом Ейлера проміжне значення:

$$\tilde{u}_{n+1/2} = u_i + \frac{\tau}{2} f(t_n, u_n), \tag{8.34}$$

а потім — значення наближеного розв'язку в точці t_{n+1}:

$$u_{n+1} = u_n + \tau f\left(t_n + \frac{\tau}{2}, \tilde{u}_{n+1/2}\right). \tag{8.35}$$

Реалізація методу (8.33) у два етапи — (8.34) і (8.35) — називається *методом прогнозу-корекції*, оскільки на першому етапі обчислень за формулою (8.33) наближене значення прогнозується з невисокою точністю $O(\tau)$, а на другому — це прогнозоване значення уточнюється, так що сумарна похибка має другий порядок за τ.

Метод Ралстона–Рабіновича

Для цього методу $b_2 = 2/3$, завдяки чому мінімізується локальна похибка методів другого порядку. Якщо $b_1 = 1/3$, $c_2 = a_{21} = 3/4$:

$$u_{n+1} = u_n + \left(\frac{1}{3}k_1 + \frac{2}{3}k_2\right)\tau, \tag{8.36}$$

де $k_1 = f(t_n, u_n)$, $k_2 = f(t_n + (3/2)\tau, u_n + (3/2)\tau k_1)$.

8.3.2. Явний метод Рунге–Кутта третього порядку точності

Під час розв'язання звичайних диференціальних рівнянь часто використовують методи третього і четвертого порядків точності. Вони реалізуються у такий же спосіб, як і метод другого порядку, — формули отримують тим же шляхом (див. підрозділ 8.3.1).

Для $p = 3$ у розкладі Тейлора (8.22) використовується додатковий член, що містить другу похідну від правої частини диференціального рівняння $y' = f(t, y)$:

$$f''(y,t) = \frac{\partial\left[\frac{\partial f}{\partial t} + \left(\frac{\partial f}{\partial y}\right)\left(\frac{dy}{dt}\right)\right]}{\partial t} + \frac{\partial\left[\frac{\partial f}{\partial t} + \left(\frac{\partial f}{\partial y}\right)\left(\frac{dy}{dt}\right)\right]}{\partial y} f.$$

Замість рівнянь (8.27) отримуємо систему з шести рівнянь з вісьмома невідомими. Отже, значення двох із них повинні бути обрані апріорі для знаходження інших констант. Один із методів третього порядку виглядає так:

$$u_{n+1} = u_n + \frac{1}{6}(k_1 + 4k_2 + k_3)\tau, \tag{8.37}$$

де $k_1 = f(t_n, u_n)$, $k_2 = f(t_n + (1/2)\tau, u_n + (1/2)\tau k_1)$, $k_3 = f(t_n + \tau, u_n - \tau k_1 + 2\tau k_2)$.

Для будь-якого з таких методів третього порядку локальна похибка $O(\tau^4)$ визначається першим неврахованим членом розкладу в ряд Тейлора, а глобальна похибка $O(\tau^3)$ — порядком методу. Як і в разі використання методу Ейлера, для уточнення розв'язку можливе застосування екстраполяції Річардсона. Однак тепер замість виразу (8.21) справедливим буде такий запис:

$$F(\tau) = a_0 + a_2\tau^2 + a_3\tau^3 + a_4\tau^4 + a_5\tau^5 + \ldots .$$

Тобто якщо $q = 2$, перший рядок схеми обчислень за Річардсоном містить значення $\Delta/3, \Delta/7, \Delta/15, \ldots$.

8.3.3. Явний метод Рунге–Кутта четвертого порядку точності

Найбільше поширення отримали методи Рунге–Кутта четвертого порядку точності. Якщо до їх побудови застосувати той самий підхід, що і до побудови методів другого порядку, то отримаємо велику кількість можливих варіантів цих методів. Один із них, що називається класичним, має таку формулу наближення:

$$u_{n+1} = u_n + \frac{\tau}{6}(k_1 + 2k_2 + 2k_3 + k_4), \tag{8.38}$$

де

$$k_1 = f(t_n, u_n),$$
$$k_2 = f\left(t_n + \frac{1}{2}\tau, u_n + \frac{1}{2}\tau k_1\right),$$
$$k_3 = f\left(t_n + \frac{1}{2}\tau, u_n + \frac{1}{2}\tau k_2\right),$$
$$k_4 = f(t_n + \tau, u_n + \tau k_3).$$

Графічна інтерпретація методу (8.38) представлена на рис. 8.2.

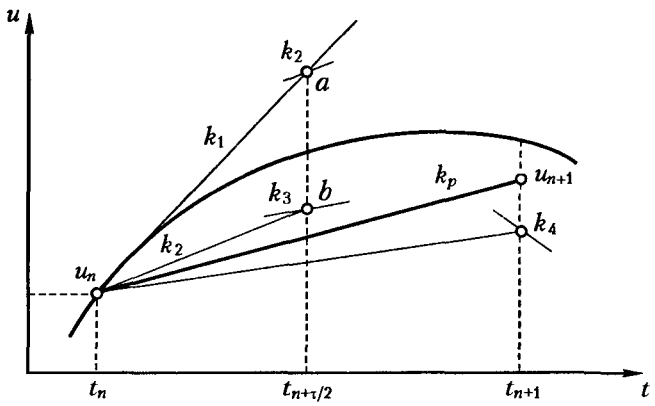


Рис. 8.2. Графічна інтерпретація методу Рунге–Кутта четвертого порядку

На рис. 8.2 видно, як з початкової точки u_n виконується перший півкрок за дотичною з нахилом k_1 і знаходиться точка a, в якій визначається нахил дотичної до кривої розв'язку k_2. Потім з тієї ж самої початкової точки виконується наступний півкрок по прямій, але вже з нахилом k_2. Знаходиться нова точка b і новий нахил дотичної до кривої розв'язку k_3. Тепер вже по прямій з нахилом k_3 з початкової точки робиться повний крок, що дозволяє знайти нову точку c і нове значення нахилу дотичної k_4. Обчислені раніше нахили дотичної усереднюються і з отриманим значенням k_p виконується заключний повний лінійний крок, в результаті якого знаходиться наступна точка наближення — u_{n+1}.

8.3.4. Явні методи Рунге–Кутта високих порядків

Якщо потрібна більш висока точність розрахунку, застосовують метод Рунге–Кутта п'ятого порядку, запропонований Бутчером:

$$u_{n+1} = u_n + \tau\left[\frac{1}{90}(7k_1 + 32k_3 + 12k_4 + 32k_5 + 7k_6)\right], \qquad (8.39)$$

де

$$k_1 = f(t_n, u_n),$$
$$k_2 = f(t_n + \tau/4, u_n + k_1\,\tau/4),$$
$$k_3 = f(t_n + \tau/4, u_n + k_1\,\tau/8 + k_2\,\tau/8),$$
$$k_4 = f(t_n + \tau/2, u_n - k_2\,\tau/2 + k_3\tau),$$
$$k_5 = f(t_n + 3\tau/4, u_n + 3k_1\,\tau/16 + 9k_4\,\tau/16),$$
$$k_6 = f(t_n + \tau, u_n - 3k_1\,\tau/7 + 2k_2\,\tau/7 + 12k_3\,\tau/7 - 12k_4\,\tau/7 + 8k_5\,\tau/7).$$

Метод (8.39), в якому підвищення точності досягається за рахунок збільшення обсягу обчислень, застосовується рідше порівняно з класичним методом Рунге–Кутта четвертого порядку.

Приклад 8.3

Розв'яжемо методом Рунге–Кутта рівняння: $y' = y + t$, $y(0) = 1$ і для кроку $\tau = 0,2$ знайдемо значення для $y(0, 2)$.

Скористаємося формулою методу Рунге–Кутта четвертого порядку (8.38). Для $\tau = 0,2$ заповнимо табл. 8.2.

Таблиця 8.2. Результати обчислень за методом Рунге–Кутта четвертого порядку

t	y	y'	$k = \tau y'$
0	1	1	0,2
0,1	1,1	1,2	0,24
0,1	1,12	1,22	0,244
0,2	1,244	1,444	0,2888

Використовуючи дані останнього стовпця таблиці, знаходимо:

$$y(0,2) = 1 + \frac{0,2 + 2 \cdot 0,24 + 2 \cdot 0,244 + 0,2888}{6} = 1,2428.$$

8.4. Апостеріорні оцінки похибки методів Рунге–Кутта

Під час виконання практичних розрахунків необхідно забезпечити задану точність результатів обчислень. Апріорні оцінки точності для цього малокорисні. По-перше, головні члени похибки визначаються через похідні розв'язку, який до початку розрахунку невідомий (див. підрозділ 8.2.2, формулу (8.15)). По-друге, апріорні оцінки звичайно є мажорантними і можуть у багато разів перевищувати фактичну помилку розрахунку. Тому основним практичним прийомом є апостеріорна оцінка точності. Розрахунок при цьому проводять на двох або більшій кількості сіток, що згущаються, і застосовують правило Рунге, яке

можна сформулювати, базуючись на основному виразі екстраполяційної формули Річардсона (8.19):

$$F(\tau) - a_0 = \frac{F(\tau) - F(q\tau)}{q^p - 1}$$

або для $q = 1/2$:

$$u_{n+1}^{(0,5\tau)} - x(t_n + \tau) \approx \frac{u_{n+1}^{(\tau)} - u_{n+1}^{(0,5\tau)}}{2^p - 1}. \tag{8.40}$$

Нагадаємо, що локальна похибка методу порядку p визначається як

$$\eta_n = O(\tau^{p+1}) \quad \text{або} \quad \eta_{n+1}(\tau) = \varphi^{(p+1)}(t_n, u_n)\tau^{p+1}. \tag{8.41}$$

Приклад 8.4

Розв'яжемо задачу Коші:

$$y' = 2(t-2) + t(t-2)^2 - ty, \quad y(2) = 0 \tag{8.42}$$

на відрізку $t \in [2, 4]$ методом Рунге–Кутта четвертого порядку за допомогою пакета Mathematica. Виберемо крок $\tau = 0,2$, складемо програму обчислення розв'язку й максимальної похибки. Спочатку визначимо в Mathematica праву частину рівняння і вирази для коефіцієнтів формули Рунге–Кутта:

```
In[]:= f[t_,y_]:= 2*(t - 2) + t*(t - 2)^2 - t*y;
       k1[τ_]:= f[t,y];
       k2[τ_]:= f[t + 0.5*τ, y + 0.5*τ*k1[τ]];
       k3[τ_]:= f[t + 0.5*τ, y + 0.5*τ*k2[τ]];
       k4[τ_]:= f[t + τ, y + τ*k3[τ]];
```

Введемо границі відрізку інтегрування, початкову умову і розв'яжемо задачу з кроками τ і $0,5\tau$:

```
In[]:= m = 10; t0 = 2; y0 = 0; tm = 4; τ = (tm - t0)/m;
       U = Array[u, {2, m + 1}, 0];        u[0,0] = t0; u[1,0] = y0;
       U1 = Array[u1, {2, 2*m + 1}, 0];    u1[0,0] = t0; u1[1,0] = y0;
       Ep = Array[ep, m + 1, 0]; ep[0] = 0;
       j = 0; t = t0; y = y0;
       Do [y1 = y; t1 = t;
         Do [dy1 = τ/2*(k1[0.5* τ] + 2*k2[0.5* τ] + 2*k3[0.5*τ] + k4[0.5*τ])/6;
             t = t + 0.5*τ; y = y + dy1; j = j + 1; u1[0,j] = t; u1[1,j] = y, {k, 1, 2} ];
       y = y1; t = t1; dy = τ*(k1[τ] + 2*k2[τ]+ 2*k3[τ] + k4[τ])/6;
       t = t + τ; u[0,i] = t; u[1,i] = y + dy;
       ep[i] = Abs[u1[1, 2*i] - u[1, i]]/(2^5 - 1);
       y = u1[1, 2*i], {i, 1, m} ];
```

Побудуємо графіки сіткових функцій, які відповідають наближеним розв'язкам із кроками τ і $0,5\tau$. Щоб розмістити обидва графіки на одному рисунку, відкриємо графічний пакет і пакет, який дозволяє додати до рисунка ідентифікуючі написи:

```
In[]:= <<Graphics`MultipleListPlot`
In[]:= <<Graphics`Legend`
```

Відобразимо графіки отриманих розв'язків (рис. 8.3):

```
In[]:= U = Array[u, {2, m + 1}, 0];
       U1 = Array[u1, {2, 2*m + 1}, 0];
       MultipleListPlot[Transpose[U], Transpose[U1],
       PlotLegend → {" крок τ ", " крок 0.5 τ "}, AxesLabel → {"t", "U"}]
```

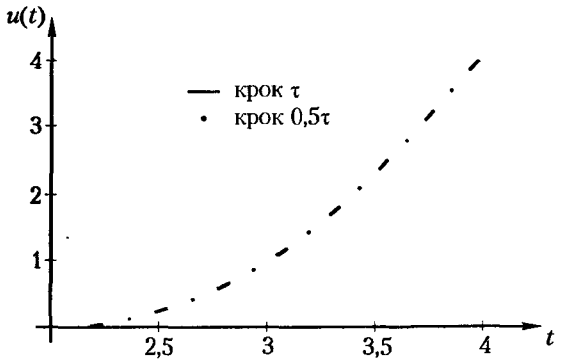


Рис. 8.3. Графік чисельного розв'язку задачі Коші методом Рунге–Кутта

Помітити на графіку різницю між значеннями наближених розв'язків у вузлах, що збігаються, неможливо через малу похибку. Знайдемо за правилом Рунге оцінку похибки у вузлах основної сітки за формулою (8.43) і побудуємо графік цієї оцінки:

```
In[]:= Ep = Array[ep, m+1, 0];
       ListPlot[Abs[Ep], PlotStyle → PointSize[0.015], AxesLabel → {"n", "Оцінка Рунге"}]
```

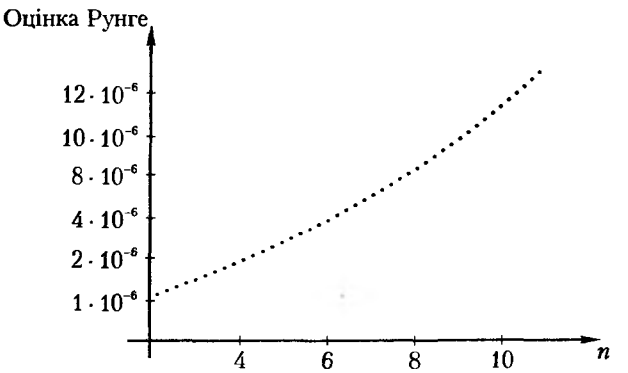


Рис. 8.4. Графік оцінки за Рунге похибки чисельного розв'язку методом Рунге–Кутта задачі Коші (8.42)

Як випливає з одержаних оцінок, розв'язок для $\tau = 0{,}5$ отримано з високою точністю. У Mathematica можна знайти аналітичний розв'язок даної задачі Коші за допомогою оператора DSolve:

```
In[]:= f[t_,y_]:= 2*(t - 2) + t*(t - 2)^2 - t*y;
       V = [{v'[t] == f[t, v[t]], v[2] == 0}, v[t], t];
       z[t_] = Simplify[V[[1,1,2]]];
       Print["z[t]=", z[t]]
Out[]= z[t] = (-2 + t)2
```

Аналітичний розв'язок одержано, тому можна обчислити дійсну глобальну похибку розв'язку, отриманого за методом Рунге–Кутта:

```
In[]:= Wl = Array[wl, {2, 2*m + 1}, 0];
       Do[wl[0,i] = ul[0,i]; wl[1,i] = ul[1,i] - y[ul[0,i]], {i, 0, 2*m} ];
       ListPlot[Transpose[Wl], PlotStyle → PointSize[0.02],
                AxesLabel → {"t", "Глобальна похибка"}]
```

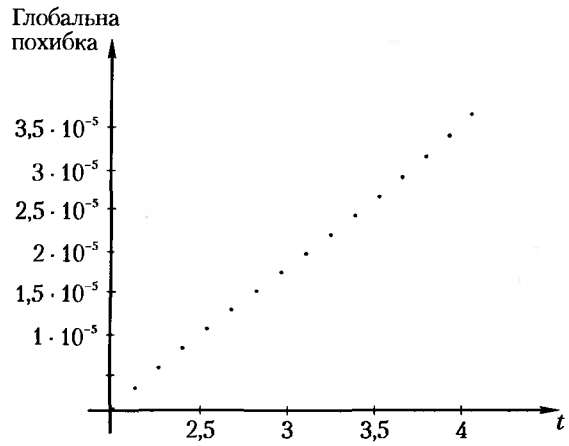


Рис. 8.5. Графік похибки чисельного розв'язку методом Рунге–Кутта задачі Коші (8.42)

8.5. Методи Рунге–Кутта–Фельберга

Щоби більш гнучко керувати процесом інтегрування, бажано мати можливість змінювати крок інтегрування і оцінювати похибку за меншої кількості обчислюваних значень правих частин рівняння. Застосування розглянутого в прикладі 8.4 методу покрокового контролю точності чисельного розв'язку шляхом вибору кроку інтегрування на основі подвійного розрахунку за правилом Рунге призводять до значного збільшення обсягу обчислень. Дійсно, нехай для розв'язання задачі (8.8) використовується метод Рунге–Кутта четвертого порядку точності. Тоді для виконання одного його кроку з контролем точності за правилом Рунге потрібно одинадцять разів обчислити праву частину рівняння (див. приклад 8.4). Щоб підвищити ефективність чисельного інтегрування диференціальних рівнянь і контроль покрокової похибки, були запропоновані методи, які отримали назву вбудованих методів Рунге–Кутта. Вони передбачають знаходження чисельного розв'язку в точці t_{n+1} двома методами — порядків точності p і $p+1$, — для того щоб отримати оцінку похибки інтегрування порядку p. За аналогією з формулою (8.22) запишемо:

$$\begin{cases} F(p) = a_0 + a_1\tau^p + O(\tau^r), \\ F(p+1) = a_0 + a_1(\tau)^{p+1} + O(\tau^r), \end{cases}$$

звідки

$$a_0 = F(p) + \frac{F(p) - F(p+1)}{\tau - 1} + O(\tau^r),$$

або в прийнятих у цьому розділі позначеннях, коли $\tau \ll 1$:

$$u_{n+1} = u_{n+1}^{(p)} + \frac{u_{n+1}^{(p)} - u_{n+1}^{(p+1)}}{\tau - 1} \approx u_{n+1}^{(p)} + (u_{n+1}^{(p+1)} - u_{n+1}^{(p)}). \tag{8.43}$$

Згідно з виразами (8.23) і (8.24) формули методів Рунге–Кутта порядків p і $p+1$ можна записати відповідно:

$$u_{n+1}^{(p)} = u_n + \tau \sum_{i=1}^{p} b_i k_i \quad \text{та} \quad u_{n+1}^{(p+1)} = u_n + \tau \sum_{i=1}^{p+1} b_i^* k_i^*,$$

тому що константи b_i і оцінки похідних k_i для методів різних порядків, на жаль, найчастіше не збігаються. Похибка обчислень згідно з (8.43) у загальному випадку оцінюється формулою

$$\varepsilon_{n+1} = u_{n+1}^{(p)} - u_{n+1}^{(p+1)} = \tau \sum_{i=1}^{p+1} (b_i k_i - b_i^* k_i^*). \tag{8.44}$$

Тут деякі b_i і k_i, відсутні в методі порядку p, приймають нульові значення. Формулу похибки (8.43) можна спростити, якщо з можливої множини методів різних порядків підібрати такі, для яких оцінки похідних k_i збігаються. Такі дослідження проведені Фельбергом для поєднання методів Рунге–Кутта різного порядку.

Наприклад, у формулах Фельберга другого порядку разом використовуються методи Рунге–Кутта другого і третього порядків точності. Як основу обрано метод Рунге–Кутта другого порядку, що визначається виразом (8.32), у якому $b_1 = 1/2$ і $k_1 = f(t_n, u_n)$, $k_2 = f(t_n + \tau, u_n + \tau k_1)$. Нагадаємо, що в разі вибору коефіцієнтів формули методу третього порядку формується система з шести рівнянь із вісьмома невідомими. Отже, значення двох із них мають бути обрані апріорі, щоб можна було знайти інші параметри. Тепер вільні параметри приймаються рівними чотирьом відповідним параметрам методу другого порядку, після чого визначаються інші чотири. Як результат отримуємо:

$$u_{n+1} = u_n + \left(\frac{1}{6}k_1 + \frac{1}{6}k_2 + \frac{4}{6}k_3\right)\tau, \tag{8.45}$$

де $k_1 = f(t_n, u_n)$, $k_2 = f(t_n + \tau, u_n + \tau k_1)$, $k_3 = f(t_n + (1/2)\tau, u_n + (1/4)k_1\tau + (1/4)k_2\tau)$.

Фельберг отримав багато формул різних порядків. Нижче наведено формули найбільш розповсюдженого на сьогодні методу Фельберга, який поєднує

методи Рунге–Кутта четвертого і п'ятого порядків і має похибку апроксимації порядків τ^4 і τ^5 відповідно:

$$u_{n+1} = u_n + \left(\frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4104}k_4 - \frac{1}{5}k_5\right)\tau, \tag{8.46}$$

$$u_{n+1} = u_n + \left(\frac{16}{135}k_1 + \frac{6656}{12825}k_3 + \frac{28561}{56430}k_4 - \frac{9}{50}k_5 + \frac{2}{55}k_6\right)\tau, \tag{8.47}$$

де

$$\begin{aligned}
k_1 &= f(t_n,\ u_n),\\
k_2 &= f\left(t_n + \frac{1}{4}\tau,\ u_n + \frac{1}{4}k_1\tau\right),\\
k_3 &= f\left(t_n + \frac{3}{8}\tau,\ u_n + \frac{3}{32}k_1\tau + \frac{9}{32}k_2\tau\right),\\
k_4 &= f\left(t_n + \frac{12}{13}\tau,\ u_n + \frac{1932}{2197}k_1\tau - \frac{7200}{2197}k_2\tau + \frac{7296}{2197}k_3\tau\right),\\
k_5 &= f\left(t_n + \tau,\ u_n + \frac{439}{216}k_1\tau - 8k_2\tau + \frac{3680}{513}k_3\tau - \frac{845}{4104}k_4\tau\right),\\
k_6 &= f\left(t_n + \frac{1}{2}\tau,\ u_n - \frac{8}{27}k_1\tau + 2k_2\tau - \frac{3544}{2565}k_3\tau + \frac{1859}{4104}k_4\tau - \frac{11}{40}k_5\tau\right).
\end{aligned} \tag{8.48}$$

Оцінка похибки, яка отримана відніманням рівнянь (8.46) і (8.47), дорівнює:

$$\varepsilon_{i+1} = \left(\frac{1}{360}k_1 - \frac{128}{4275}k_3 - \frac{2197}{75240}k_4 + \frac{1}{50}k_5 + \frac{2}{55}k_6\right)\tau. \tag{8.49}$$

Є також інший альтернативний вбудований метод — метод Зонефельда, який поєднує методи Рунге–Кутта четвертого і п'ятого порядків, але має більш прості коефіцієнти:

$$u_{n+1} = u_n + \left(\frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{4}k_4\right)\tau, \tag{8.50}$$

$$u_{n+1} = u_n + \left(-\frac{1}{2}k_1 + \frac{7}{3}k_2 + \frac{7}{3}k_3 + \frac{13}{6}k_4 - \frac{16}{3}k_5\right)\tau, \tag{8.51}$$

де

$$\begin{aligned}
k_1 &= f(t_n, u_n),\\
k_2 &= f\left(t_n + \frac{1}{2}\tau, u_n + \frac{1}{2}k_1\tau\right),\\
k_3 &= f\left(t_n + \frac{1}{2}\tau, u_n + \frac{1}{2}k_2\tau\right),\\
k_4 &= f(t_n + \tau, u_n + k_3\tau),\\
k_5 &= f\left(t_n + \frac{5}{32}\tau, u_n + \frac{7}{32}k_1\tau + \frac{13}{32}k_3\tau - \frac{1}{32}k_4\tau\right).
\end{aligned} \tag{8.52}$$

Звичайна практика контролю похибки обчислень на певному кроці полягає в тому, що користувач вказує в програмі дані про інтервал інтегрування T і бажану загальну похибку розв'язку ε^*. Програма визначає припустиму похибку розв'язку на поточному кроці τ_{n+1} як частину загальної похибки ε, що припадає на цей крок,

$$\varepsilon^*_{n+1} = \frac{\varepsilon}{T}\tau_{n+1}, \tag{8.53}$$

і порівнює її з оцінкою похибки ε_{n+1}, отриманої або на основі розв'язку з різними кроками (8.40), або на основі розв'язку методами різних порядків (8.43). Якщо $\varepsilon_{n+1} < \varepsilon^*_{n+1}$, крок збільшується (наприклад, подвоюється), і навпаки — коли $\varepsilon_{n+1} > \varepsilon^*_{n+1}$, крок зменшується (наприклад, удвічі).

Приклад 8.5

Розв'яжемо методом Фельберга задачу Коші

$$y' = 2(t-2) + t(t-2)^2 - ty, \quad y(2) = 0,$$

на відрізку $t \in [2, 4]$ за допомогою пакета Mathematica.

Задамо крок $\tau = 0{,}1$, складемо програму розв'язання задачі й обчислення максимальної похибки отриманого розв'язку:

```
In[]:= f[t_,y_]:= 2*(t - 2) + t*(t - 2)^2 - t*y;
```

$$\text{Z[t\_,tn\_,un\_]:= } E^{-\frac{t^2}{2}}\left(4E^{\frac{t^2}{2}} - 4E^{\frac{t^2}{2}}t + E^{\frac{t^2}{2}}t^2 - E^{\frac{tn^2}{2}}(4 - 4tn + tn^2 - un)\right);$$

$$\text{k2[}\tau\text{\_]:= f[t + 0.25*}\tau\text{, y + 0.25* }\tau\text{*k1[}\tau\text{]];}$$

$$\text{k3[}\tau\text{\_]:= f[t + }\frac{3}{8}\text{*}\tau\text{, y + }\frac{3}{32}\text{*}\tau\text{*k1[}\tau\text{] + }\frac{9}{32}\text{*}\tau\text{*k2[}\tau\text{]];}$$

$$\text{k4[}\tau\text{\_]:= f[t + }\frac{12}{13}\text{*}\tau\text{, y + }\tau\left(\frac{1932}{2197}\text{k1[}\tau\text{] + }\frac{7200}{2197}\text{k2[}\tau\text{] + }\frac{7296}{2197}\text{k3[}\tau\text{])];}$$

$$\text{k5[}\tau\text{\_]:= f[t + }\tau\text{, y + }\tau\left(\frac{439}{216}\text{*k1[}\tau\text{] - 8*k2[}\tau\text{] + }\frac{3680}{513}\text{*k3[}\tau\text{] - }\frac{845}{4104}\text{*k4[}\tau\text{])];}$$

$$\text{k6[}\tau\text{\_]:= f[t + 0.5*}\tau\text{,}$$
$$\text{y + }\tau\left(-\frac{8}{27}\text{*k1[}\tau\text{] - 2*k2[}\tau\text{] - }\frac{3544}{2565}\text{*k3[}\tau\text{] + }\frac{1859}{4104}\text{*k4[}\tau\text{] - }\frac{11}{40}\text{*k5[}\tau\text{])];}$$

Введемо границі відрізка інтегрування, початкову умову і розв'яжемо задачу з кроком $\tau = 0.1$:

```
In[]:= m = 20; t0 = 2; y0 = 0; tm = 4; τ = (tm - t0)/m;
     .U = Array[u,{m + 1, 2}, 0]; u[0,0] = N[t0]; u[0,1] = y0;
      Ep = Array[ep, {m + 1, 2}, 0]; Array[el, {m+1, 2}, 0]; Et = Array[et, {m+1, 2}, 0];
      ep[0,0] = t0; ep[0,1] = 0; et[0,0] = t0; et[0,1] = 0; el[0,0] = t0; el[0,1] = 0;
      j = 0; t = t0; y = x0;
      Do [dy1 = N[τ*(-16/135 *k1[τ] + 6656/12825 *k3[τ] + 28561/56430 *k4[τ] - 9/50 *k5[τ] + 2/55 *k6[τ])];
      dy = N[τ*(25/216 *k1[τ] + 1408/2565 *k3[τ] + 2197/4104 *k4[τ] - 1/5 *k5[τ])];
      ep[i,1] = dy - dy1; y = y + dy; u[i,1] = y; t = N[t + τ];
      ep[i,0] = t;        et[i,0] = t; el[i,0] = t; u[i,0] = t;
      el[i,1] = Z[t, t - τ, u[i-1,1]] - u[i,1]; et[i,1] = y-(t - 2)^2, {i,1,m}];
```

```
In[]:= W = Abs [N[Table[{u[i,0], u[i,1], ep[i,1], et[i,1]}, {i,0,10}], 4]];
       Print["Таблиця 8.3."];
       TableForm[W, TableHeadings → {Automatic, {"n", "t", "uₙ", "εₚ", "εₜ"}}]
```

У табл. 8.3 наведено перші десяти значень наближеного розв'язку u_n, значення оцінки похибки, отриманої методом Фельберга ε_p, і значення покрокової помилки ε_t, визначеної за точним аналітичним розв'язком задачі.

Таблиця 8.3. Результати обчислень за методом Фельберга

n	t	u_n	ε_p	ε_t
1	2,0	0	0	0
2	2,1	0,01	$2{,}81107 \times 10^{-7}$	$3{,}30114 \times 10^{-7}$
3	2,2	0,04	$3{,}19822 \times 10^{-7}$	$6{,}42736 \times 10^{-7}$
4	2,3	0,09	$3{,}62239 \times 10^{-7}$	$9{,}4069 \times 10^{-7}$
5	2,4	0,16	$4{,}08557 \times 10^{-7}$	$1{,}22698 \times 10^{-6}$
6	2,5	0,25	4.58976×10^{-7}	1.50465×10^{-6}
7	2,6	0,36	$5{,}13701 \times 10^{-7}$	$1{,}77668 \times 10^{-6}$
8	2,7	0,49	$5{,}72938 \times 10^{-7}$	$2{,}04592 \times 10^{-6}$
9	2,8	0,64	$6{,}36897 \times 10^{-7}$	$2{,}31502 \times 10^{-6}$
10	2,9	0,81	$7{,}05791 \times 10^{-7}$	$1{,}58640 \times 10^{-6}$
11	3,0	1,0	$7{,}79833 \times 10^{-7}$	$2{,}86222 \times 10^{-6}$

Коефіцієнти формули, яка визначає наближений розв'язок за формулою нижчого порядку, обчислювались шляхом мінімізації локальної похибки результату. Як наслідок цього випливає, що для оцінки похибки за формулою (8.51) можна не враховувати локальну похибку.

8.6. Чисельне розв'язання систем диференціальних рівнянь першого порядку

Наведені вище формули Рунге–Кутта розв'язання задачі Коші для диференціального рівняння першого порядку можна застосувати і до чисельного розв'язання системи диференціальних рівнянь першого порядку. Для цього таку систему достатньо записати у векторній формі.

Нехай потрібно знайти розв'язок задачі Коші для системи диференціальних рівнянь першого порядку виду:

$$\begin{aligned}
&\frac{dy_1}{dt} = f_1(t, y_1, ..., y_m), \quad y_1(t_0) = y_1^0,\\
&\frac{dy_2}{dt} = f_2(t, y_1, ..., y_m), \quad y_2(t_0) = y_2^0,\\
&\qquad\qquad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots\\
&\frac{dy_m}{dt} = f_m(t, y_1, ..., y_m), \quad y_m(t_0) = y_m^0.
\end{aligned} \tag{8.54}$$

Введемо векторні позначення:

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \dots \\ y_m \end{bmatrix}, \quad \mathbf{Y}' = \begin{bmatrix} y_1' \\ y_2' \\ \dots \\ y_m' \end{bmatrix}, \quad \mathbf{F}(t, \mathbf{Y}) = \begin{bmatrix} f_1(t, y_1, \dots, y_m) \\ f_2(t, y_1, \dots, y_m) \\ \dots \quad \dots \quad \dots \\ f_m(t, y_1, \dots, y_m) \end{bmatrix}, \quad \mathbf{Y}_0 = \begin{bmatrix} y_1^0 \\ y_2^0 \\ \dots \\ y_m^0 \end{bmatrix}. \tag{8.55}$$

У використаних позначеннях задача Коші набуде вигляду

$$\mathbf{Y}' = \mathbf{F}(t, \mathbf{Y}), \quad \mathbf{Y}(t_0) = \mathbf{Y}_0. \tag{8.56}$$

Формальна відмінність цієї задачі від (8.6) полягає в тому, що у відповідні співвідношення замість скалярних величин входять вектори. Тому до векторного диференціального рівняння (8.56) можна застосувати будь-який із чисельних методів, які вивчалися в цьому розділі. Наведемо приклад формул для розв'язання системи диференціальних рівнянь методом Рунге–Кутта четвертого порядку для системи, записаної у вигляді (8.56):

$$\begin{gathered} \mathbf{Y}_0 = \mathbf{Y}(0), \\ \mathbf{K}_1 = \mathbf{F}(t_n, \mathbf{U}_n), \\ \mathbf{K}_2 = \mathbf{F}(t_n + 0,5\tau, \mathbf{U}_n + 0,5\tau\mathbf{K}_1), \\ \mathbf{K}_3 = \mathbf{F}(t_n + 0,5\tau, \mathbf{U}_n + 0,5\tau\mathbf{K}_2), \\ \mathbf{K}_4 = \mathbf{F}(t_n + \tau, \mathbf{U}_n + \tau\mathbf{K}_3), \\ \mathbf{U}_{n+1} = \mathbf{U}_n + \frac{\tau}{6}(\mathbf{K}_1 + 2\mathbf{K}_2 + 2\mathbf{K}_3 + \mathbf{K}_4), \\ n = 0, 1, 2, \dots . \end{gathered} \tag{8.57}$$

Формули (8.57) за формою збігаються з формулами розв'язання одного диференціального рівняння першого порядку. Наведемо приклад їх використання.

Приклад 8.6

Розв'яжемо методом Ейлера задачу Коші

$$\begin{aligned} y_1' &= -0,5y_1, \\ y_2' &= 4 - 0,3y_2 - 0,1y_1. \end{aligned}$$

При цому задано такі початкові умови: $y_1(0) = 4$, $y_2(0) = 6$.

Із міркувань точності виберемо $\tau = 0,5$. Знайдемо значення $y_1(2)$ і $y_2(2)$, виконавши чотири кроки обчислень. Для першого кроку відповідно до методу Ейлера

$$\begin{aligned} y_1(0,5) &= y_1(0) + 0,5[-0,5y_1(0)] = 4 - 1 = 3, \\ y_2(0,5) &= y_2(0) + 0,5[4 - 0,3y_2(0) - 0,1y_1(0)] = 6 + 0,9 = 6,9. \end{aligned}$$

Результати розрахунку, отримані в такий же спосіб на наступних ітераціях, зведені в табл. 8.4.

Таблиця 8.4. Розв'язок системи диференціальних рівнянь, отриманий методом Ейлера

t	y_1	y_2
1,0	2,25	7,715
1,5	1,6875	8,44525
2,0	1,265625	9,0940875

Ще раз обчислимо значення $y_1(2)$ і $y_2(2)$ із кроком $\tau = 1$:

$$y_1(1) = y_1(0) + 1[-0,15y_1(0)] = 4 - 2 = 2,$$

$$y_2(1) = y_2(0) + 1[4 - 0,3y_2(0) - 0,1y_1(0)] = 6 + 1,18 = 7,18,$$

$$y_1(2) = y_1(1) + 1[-0,5y_1(1)] = 2 - 1 = 1,$$

$$y_2(2) = y_2(1) + 1[4 - 0,3y_2(1) - 0,1y_1(1)] = 7,8 + 1,8 = 9,26.$$

Тепер, застосувавши екстраполяцію Річардсона, отримаємо більш точні результати:

$$y_1(2) = 1,265625 + (1,265625 - 1) = 1,531251,$$

$$y_2(2) = 9,0940875 + (9,0940875 - 9,26) = 8,928175.$$

Приклад 8.7

Розв'яжемо в пакеті Mathematica систему диференціальних рівнянь із заданими початковими умовами

$$y_1' = e^{-(y_1^2+y_2^2)} + 2t; \quad y_2' = 2y_1^2 + y_2; \quad y_1(0) = 0,5; \quad y_2(0) = 1.$$

Задамо початкові умови, величину та кількість кроків інтегрування

```
In[]:= t0 = 0;
       y0 = {0.5, 1};
       τ = 0.01; n = 10;
```

Визначимо вектор-функцію правих частин рівнянь:

```
In[]:= F[t_, y_] = {E^-(y[[1]]^2 + y[[2]]^2) + 2t, 2y[[1]]^2 + y[[2]]};
```

Опишемо масив наближеного розв'язку U:

```
In[]:= Array[U, n + 1, 0];
```

Далі наведено програму обчислення наближеного розв'язку на перших десяти кроках.

```
In[]:= U[0] = y0; t = t0;
       Do [K1 = F[t, U[i]];
           K2 = F[t + 0.5τ, U[i] + 0.5τK1];
           K3 = F[t + 0.5τ, U[i] + 0.5τK2]
           K4 = F[t + τ, U[i] + τK3];
           U[i+1] = U[i] + τ/6 (K1 + 2K2 + 2K3 + K4);
           t = t + τ, {i, 0, n}
```

Виведемо таблицю перших дев'яти значень розв'язку задачі:

```
In[]:= TA = Table[{t0 + (i-1)*τ, U[i-1][[1]], U[i-1][[2]]}, {i, 10}];
       Print["Таблиця 8.5"];
       TableForm[TA, TableHeadings → {Automatic, {"n", "t", "u₁", "u₂"}}]
```

Таблиця 8.5. Результати розв'язання системи диференціальних рівнянь методом Рунге–Кутта

n	t	u_1	u_2
1	0	0,5	1
2	0,01	0,502918	1,0151
3	0,02	0,505942	1,03042
4	0,03	0,509073	1,04595
5	0,04	0,512311	1,06171
6	0,05	0,515656	1,07769
7	0,06	0,51911	1,0939
8	0,07	0,522673	1,11035
9	0,08	0,526346	1,12704
10	0,09	0,53013	1,14397

8.7. Стійкість методів Рунге–Кутта

Для практичних розрахунків важливо знати, як змінюється розв'язок диференціального рівняння чи системи рівнянь залежно від обраного значення кроку обчислень τ. Для методів Рунге–Кутта високих порядків з їх достатнім запасом точності розв'язку бажано збільшувати значення кроку τ для зменшення кількості обчислень правої частини розв'язуваного рівняння і, як наслідок, — загального обсягу обчислень. Однак для явних методів цього робити не можна, оскільки для них існує жорстке обмеження на величину кроку [21]:

$$\tau_{\min} < \frac{c}{\max|\lambda|}, \tag{8.58}$$

де c — константа ($c < 3$), $\max|\lambda|$ — максимальне власне значення матриці Якобі для розв'язуваної системи диференціальних рівнянь (8.56). Якщо розв'язується тільки одне рівняння типу (8.8), то $\max|\lambda|$ обчислюється на основі частинної похідної від правої частини цього рівняння, тобто

$$\max|\lambda| = \max\left|\frac{\partial f}{\partial y}\right|.$$

Для «жорстких» диференціальних рівнянь матриця Якобі має досить рознесені власні значення ($\max|\lambda|/\min|\lambda| = 10^6 - 10^9$), тому обчислення слід проводити тільки з кроком, який визначається $\max|\lambda|$ за формулою (8.58), навіть на ділянці з повільною зміною розв'язку, обумовленою значенням $\min|\lambda|$. Через це

загальна кількість кроків розв'язання надмірно зростає, що дуже обмежує застосування методів Рунге–Кутта для моделювання широкосмугових інформаційних систем.

Справедливість загальної формули (8.58) найпростіше довести, виходячи з умов стійкості розв'язку різницевих рівнянь, що і буде зроблено в розділі 9. Тут же вона буде підтверджена прикладами розв'язання методами Рунге–Кутта модельного рівняння типу $y' = \lambda y$ $(\lambda < 0)$, яке широко застосовується для аналізу стійкості чисельних методів розв'язання диференціальних рівнянь.

При цьому розв'язок модельного рівняння знаходиться у такий спосіб:

$$u_{n+1} = Mu_n \quad \text{або} \quad u_{n+1} = M^{n+1}u_0,$$

де u_0 — початкове значення, u_{n+1} — значення змінної на поточному кроці, а стійкість розв'язку, коли $u_{n+1} \to 0$ і $n \to \infty$, гарантується за умови

$$|M| < 1. \tag{8.59}$$

Наприклад, для методу Ейлера (8.12) з урахуванням модельного рівняння маємо

$$u_{n+1} = u_n + \tau\lambda x_n = (1 + \tau\lambda)u_n,$$

звідки $M = |1 + \tau\lambda| < 1, \tau < -2/\lambda$ для $\lambda < 0$.

Приклад 8.8

Оцінимо допустиме значення кроку обчислень у методі Ейлера для задачі Коші

$$y_1' = -100y_1, \quad y_2' = 4 - 0{,}3y_2 - 10y_1$$

з початковими умовами $y_1(0) = 4$, $y_2(0) = 6$.
Оскільки матриця Якобі для нашого прикладу дорівнює

$$\mathbf{J} = \begin{pmatrix} -100 & 0 \\ -10 & -0{,}3 \end{pmatrix},$$

то $|\lambda_{\max}| = 100$ і максимальне значення кроку τ відповідно до умови (8.58) має бути обмежене значенням $\tau < 0{,}02$.

Можна спростити процедуру оцінки максимального допустимого кроку обчислень, якщо значення максимального модуля власних значень матриці Якобі оцінювати за нормою цієї матриці $\lambda_{\max} \leqslant \|\mathbf{J}\|$.

Для нашого прикладу $\|\mathbf{J}\|$ дорівнює 100.

Аналіз стійкості методів Рунге–Кутта проводиться за аналогічною процедурою. Наприклад, для методу другого порядку (8.32) і модельного рівняння $y' = \lambda y$ знаходимо:

$$u_{n+1} = u_n + \frac{1}{2}(k_1 + k_2)\tau,$$

де $k_1 = \lambda u_n$, $k_2 = \lambda(u_n + \tau k_1) = \lambda u_n + \tau\lambda^2$.

Після відповідних підстановок і спрощень отримуємо:

$$u_{n+1} = u_n + \tau\left(\lambda u_n + \frac{\tau}{2}\lambda^2 u_n\right) = u_n\left(1 + \tau\lambda + \frac{\tau^2\lambda^2}{2}\right),$$

звідки умова стійкості має такий вигляд:

$$M = 1 + \tau\lambda + \frac{\tau^2\lambda^2}{2} < 1.$$

Отже, для методу Рунге–Кутта четвертого порядку буде справедлива така умова вибору кроку для стійких наближень:

$$M = 1 + \tau\lambda + \frac{\tau^2\lambda^2}{2} + \frac{\tau^3\lambda^3}{3!} + \frac{\tau^4\lambda^4}{4!} < 1. \tag{8.60}$$

Розв'язання нерівностей типу (8.60) для різних τ — доволі складне завдання, тому умови стійкості прийнято відображати в графічній формі. На рис. 8.6 наведені графіки функцій $M(\lambda\tau)$ для явних методів Рунге–Кутта різних порядків. На основі графіків цих функцій для дійсних $\lambda < 0$ отримуємо такі оцінки:

$$\begin{aligned} &0 < \tau|\lambda| < 2{,}7853 \quad \text{для} \quad k = 4, \\ &0 < \tau|\lambda| < 2{,}5 \quad \text{для} \quad k = 3, \\ &0 < \tau|\lambda| < 2 \quad \text{для} \quad k = 2, 1. \end{aligned} \tag{8.61}$$

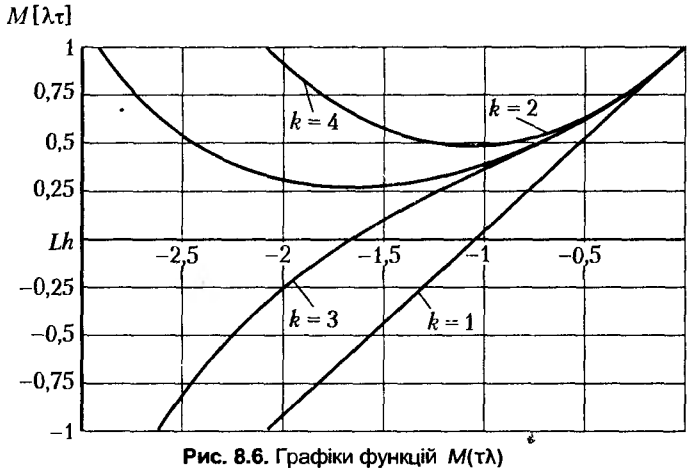


Рис. 8.6. Графіки функцій $M(\tau\lambda)$

Отримані результати показують, що зі збільшенням порядку методу обмеження на крок інтегрування послабляються незначно. Однак загальний обсяг обчислень зростає, тому що на кожному кроці методу необхідно обчислювати функцію правої частини рівняння чотири рази, а не один, як у методі Ейлера.

Оскільки λ може бути комплексним, нерівність (8.60) прийнято зображувати у вигляді годографів у комплексній площині $\tau\lambda$. Побудуємо геометричне місце точок комплексної площини $\mu = \tau\lambda$, для яких $|M| = 1$, тобто вважатимемо

$M = e^{i\varphi}$, де φ — аргумент M (рис. 8.7). Наприклад, рівняння годографа для явного методу Рунге–Кутта четвертого порядку на основі виразу (8.60) буде мати такий вигляд:

$$1 + \mu + 0,5\mu^2 + \mu^3/6 + \mu^4/24 = e^{i\varphi}.$$

Побудову годографів стійкості за наведеними рівняннями можна виконати із застосуванням пакета Mathematica.

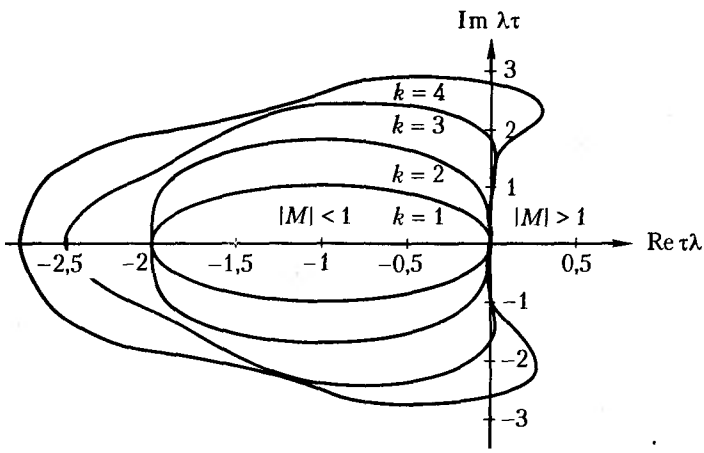


Рис. 8.7. Годографи стійкості явних методів Рунге–Кутта різних порядків

Із практичної точки зору нас цікавить, чи будуть годографи розташовані в лівій комплексній півплощині, оскільки математичні моделі фізично стійких об'єктів характеризуються від'ємними значеннями дійсних частин комплексної змінної $\lambda\tau$ ($\mathrm{Re}(\lambda\tau) < 0$). Будь-які точки площини всередині годографа відповідають стійким розв'язкам, а поза годографом — нестійким. Аналіз годографів стійкості методів Рунге–Кутта показує, що з ростом порядку методу область його стійкості розширюється, що робить їх дуже зручними для практичних обчислень.

8.8. Розв'язання задачі Коші в пакеті Mathematica методами Рунге–Кутта

Для розв'язання звичайних диференціальних рівнянь у формі Коші в пакеті Mathematica реалізовано стандартний оператор NDSolve, який має такий формат:

```
NDSolve[{eqn1, eqn2, ...}, y, {t, tmin, tmax}]
```

За його допомогою можна реалізувати, зокрема, і метод Рунге–Кутта– Фельберга, якщо у формат включити параметр, який задає бажаний метод:

```
NDSolve[{eqn1, eqn2, ...}, y, {t, tmin, tmax}, Method → ExplicitRungeKutta];
```

Під час розв'язання автоматично вибирається крок обчислень τ, виходячи з особливостей вихідної задачі. Проілюструємо застосування цього оператора для розв'язання задач, раніше розглянутих у прикладах 8.1 і 8.7. Так, задача з прикладу 8.1 $y' = t^2 - y^2, t_0 = 0, y_0 = 1$ в інтервалі часу $[0, 0,5]$ розв'язується за допомогою такого оператора:

```
In[]:= Y = NDSolve[{y'[t] == -y[t]^2 + t^2, y[0] == 1.}, y, {t, 0, 0.5},
                   Method → ExplicitRungeKutta]
Out[]= {{y → InterpolatingFunction[{{0., 0.5}}, <>]}}
```

Отримано інтерполяційну функцію, яка визначає наближений розв'язок рівняння на заданому відрізку часу. Підстановкою визначимо функцію

```
In[]:= y[t_]=y[t]/.Y
Out[]= InterpolatingFunction[{{0., 0.5}}, <>][t]}}
```

Значення наближеного розв'язку задачі для точки $t = 0,2$ допоможе отримати такий запис:

```
In[]:= y[0.2]
Out[]= {0.835785}
```

У разі розв'язання тієї ж самої задачі методом Ейлера з екстраполяцією Річардсона в прикладі 8.1 отримано близьке значення $y(0,2) = 0,84$, але обсяг обчислень був істотно меншим.

Задача Коші в інтервалі $[0, 0,9]$, що була розглянута в прикладі 8.7:

$$y_1' = e^{-(y_1^2+y_2^2)} + 2t, \quad y_2' = 2y_1^2 + y_2,$$
$$y_1(0) = 0,5, \quad y_2(0) = 1,$$

розв'язується у такий спосіб:

```
In[]:= sol = NDSolve[{y'1[t] == Exp[-y2[t]^2 + y1[t]^2] + 2t,
                      y'2[t] == 2y1[t]^2 + y2[t],
                       y1[0] == 0,5
                       y2[0] == 1.},
                     {y2, y1}, {t, 0, 0.1},
      Method → ExplicitRungeKutta]
Out[]= {{y2 → InterpolatingFunction[{{0., 0.1}}, <>],
         y1 → InterpolatingFunction[{{0., 0.1}}, <>]}}
```

Підстановкою визначимо відповідні функції:

```
In[]:= y1[t_] = y1[t]/.sol;
       y2[t_] = y2[t]/.sol
Out[]= {InterpolatingFunction[{{0., 0.9}}, <>][t]}
```

Одержимо значення наближеного розв'язку системи і виведемо їх у табл. 8.6:

```
In[]:= τ = 0.01;
       Y = TableForm[Table[{i*τ, y[i*τ][[1]], x[i*τ][[1]]} {i, 0, 9}]
       TableHeadings → {Automatic, {"t", "y1", "y2"}}]
```

Таблиця 8.6. Результати розв'язання системи диференціальних рівнянь за допомогою стандартного оператора Mathematica

i	t	y_1	y_2
1	0	0,5	1
2	0,01	0,504764	1,01512
3	0,02	0,509607	1,0305
4	0,03	0,51453	1,04612
5	0,04	0,519531	1,06201
6	0,05	0,524611	1,07816
7	0,06	0,529767	1,09458
8	0,07	0,535001	1,11128
9	0,08	0,540312	1,12826
10	0,09	0,5457	1,14553

Порівняння даних табл. 8.5 і 8.6 показує, що розв'язки, обчислені двома методами, досить близькі для заданих значень кроку.

Висновки

1. Однокрокові методи Ейлера і Рунге–Кутта будують на основі скороченого ряду Тейлора. Кількість членів цього ряду p визначає порядок методу, а перший неврахований член ряду Тейлора – локальну похибку обчислень $O(\tau^{p+1})$.
2. Щоб уникнути обчислення похідних високих порядків, що входять до членів ряду Тейлора, у методах Рунге–Кутта використовується спеціальна процедура апроксимуючих функцій, перші p членів розкладу яких у ряд Тейлора збігаються з p членами ряду Тейлора для початкової функції.
3. Порядок методу p визначає залежність глобальної похибки розв'язку від кроку обчислень $O(\tau^p)$. Чим *вище* порядок методу, тим *точніше* розв'язок, але й *більше* обсяг обчислень.
4. Зменшити похибку обчислень методом Ейлера можна як шляхом зменшення кроку, так і за допомогою процедури екстраполяції Річардсона, яка передбачає знаходження більш точного розв'язку по двох наближеннях, знайдених із різними кроками.
5. Вбудовані методи Рунге–Кутта–Фельберга (наприклад, четвертого і п'ятого порядків) з автоматичною процедурою регулювання кроку обчислень досить ефективно застосовують для розв'язання диференціальних рівнянь у всіх відомих математичних пакетах.
6. Основним недоліком методів Ейлера і Рунге–Кутта, властивим, до речі, всім явним методам, є обмеження максимального кроку обчислень τ, перевищення якого призводить до розбіжності розв'язку. Це робить дані методи малопридатними для розв'язання жорстких задач.

7. У методах Рунге–Кутта порядку $p \geqslant 3$ кількість значень правої частини розв'язуваного диференціального рівняння, які необхідно обчислити на кожному кроці, дорівнює p, що робить їх трудомісткими. Тому на практиці замість них застосовують різницеві методи, які використовують інформацію, отриману на попередніх кроках. Ці методи будуть описані в наступному розділі.

Контрольні запитання та завдання

1. Для задачі Коші $y' = 1 + t^2 y^2$, $y(0) = 0$ знайдіть $y(0,5)$ з чотирма значущими цифрами методом Ейлера з наступною екстраполяцією Річардсона наближень, отриманих із кроками $\tau = 0,5$, $\tau = 0,25$, $\tau = 0,125$.
2. Використовуючи метод Ейлера за різних кроків, отримали такі результати:

t	0,05	0,1	0,2
y	1,22726	1,22595	1,22345

Уточніть розв'язок за допомогою екстраполяції Річардсона.

3. Визначіть діапазон можливих значень кроку обчислень τ у разі використання методу Ейлера для розв'язання задачі Коші: $y' = -2$, $y(0) = 1$.
4. Для задачі Коші $y' = -2t^3 + 12t^2 - 20t + 8,5$, $y(0) = 1$ знайдіть $y(1,5)$ за кроку $\tau = 0,5$:

   а) методом Рунге–Кутта другого порядку (8.32);

   б) методом полігону (8.33);

   в) методом Ралстона (8.36);

   г) методом Рунге–Кутта третього порядку (8.37);

   д) методом Рунге–Кутта четвертого порядку (8.38).

   Знайдіть точне значення, проінтегрувавши задане диференціальне рівняння, і обчисліть глобальні похибки оцінок згаданими вище методами.
5. Побудуйте вбудовані формули Фельберга методу Рунге–Кутта першого (другого) порядку, що задовольняють умову $\alpha_{2i} = c_i$.
6. Розв'яжіть задачу $y' = y^2 e^t - 2y$, $y(0) = 1/2$ за зразком прикладу 8.5 методом Ческіно:

$$u_{n+1} = u_n + \left(\frac{1}{2}k_1 - \frac{3}{2}k_3 + 2k_4\right)\tau, \quad u_{n+1} = u_n + \left(\frac{1}{6}k_1 + \frac{2}{3}k_4 + \frac{1}{6}k_5\right)\tau,$$

де

$$k_1 = f(t_n, u_n),$$
$$k_2 = f\left(t_n + \frac{1}{4}\tau, u_n + \frac{1}{4}k_1\tau\right),$$
$$k_3 = f\left(t_n + \frac{1}{2}\tau, u_n + \frac{1}{2}k_2\tau\right),$$
$$k_4 = f(t_n + \tau, u_n + k_1\tau - 2k_2\tau + 2k_3\tau).$$

Точний розв'язок тестової задачі знайдіть за допомогою системи Mathematica. Побудуйте графіки похибок та їх оцінок.

7. Складіть програму розв'язання задачі Коші для звичайного диференціального рівняння першого порядку вбудованим методом Мерсона з автоматичним вибором кроку, що забезпечує обчислення із заданою точністю:

$$u_{n+1} = u_n + \left(\frac{1}{2}k_1 - \frac{3}{2}k_3 + 2k_4\right)\tau, \quad u_{n+1} = u_n + \left(\frac{1}{6}k_1 + \frac{2}{3}k_4 + \frac{1}{6}k_5\right)\tau,$$

де

$$k_1 = f(t_n, u_n),$$
$$k_2 = f\left(t_n + \frac{1}{3}\tau, u_n + \frac{1}{3}k_1\tau\right),$$
$$k_3 = f\left(t_n + \frac{1}{3}\tau, u_n + \frac{1}{6}k_1\tau + \frac{1}{6}k_2\tau\right),$$
$$k_4 = f\left(t_n + \frac{1}{2}\tau, u_n + \frac{1}{8}k_1\tau + \frac{3}{8}k_3\tau\right),$$
$$k_5 = f\left(t_n + \tau, u_n + \frac{1}{2}k_1\tau - \frac{3}{2}k_3\tau + 2k_4\tau\right).$$

Знайдіть розв'язок задачі Коші $y' = y^2 t - y$, $y(0) = 1$. За складеною програмою порівняйте дійсну похибку з оцінкою похибки методу Мерсона.

8. Користуючись вбудованим методом Зонефельда (8.50) – (8.52), знайдіть наближений розв'язок системи рівнянь

$$y_1' = y_2/6, \quad y_2' = -y_1 y_2^2/54, \quad y_1(4) = 2, \quad y_2(4) = 3/2.$$

Наведіть графіки оцінки похибки.

9. Побудуйте геометричне місце точок комплексної площини $\mu = \lambda\tau$, для яких виконується умова стійкості явного методу Ейлера $M = 1 + \lambda\tau$.

6.

Розділ 9

Багатокрокові методи розв'язання диференціальних рівнянь

- Екстраполяційні методи Адамса–Башфорта і Адамса–Мултона
- Багатокрокові різницеві методи
- Порядок апроксимації багатокрокових методів
- Розв'язання систем диференціальних рівнянь
- Умови стійкості багатокрокових методів

У попередньому розділі було розглянуто однокрокові методи обчислення наближеного розв'язку диференціального рівняння в точці t_{n+1} з використанням інформації про розв'язок тільки на відрізку (t_n, t_{n+1}). Логічно припустити, що точність результатів можна підвищити, якщо використовувати інформацію про поведінку розв'язку в кількох попередніх точках, $t_n, t_{n-1}, ..., t_{n-m+1}$. Такі методи розв'язання диференціальних рівнянь отримали назву *багатокрокових* [1, 4, 21].

9.1. Явні методи Адамса–Башфорта

Розглянемо один із підходів до побудови багатокрокових методів, що використовуються для знаходження наближеного розв'язку задачі Коші (8.10). Підставимо точний розв'язок у рівняння, проінтегруємо його на відрізку $[t_n, t_{n+1}]$ і обчислимо значення розв'язку $y(t_{n+1})$:

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(t, y(t))\,dt. \tag{9.1}$$

Припустимо, що вже відомий наближений розв'язок задачі (8.8) u_i в m вузлах сітки t_{n-m+1}, t_{n-1}, t_n. Отже, у цих точках відомо m значень функції $f_i = f(t_i, u_i)$, що входить до правої частини диференціального рівняння (8.8), причому $f(t_i) = f(t_i, u(t_i))$ буде вже функцією лише однієї змінної t_i. Замінимо в (9.1) підінтегральну функцію інтерполяційним поліномом $H_m(t)$, наприклад

поліномом Ньютона, побудованим для системи вузлів $t_{n-m+1}, ..., t_{n-1}, t_n$, і обчислимо наближене значення розв'язку u_{n+1}:

$$u_{n+1} = u_n + \int_{t_n}^{t_{n+1}} H_m(t)\,dt. \tag{9.2}$$

Інтегруючи це рівняння, отримаємо різницеву схему для розв'язання диференціального рівняння. Порядок схеми визначається значенням залишкового члена інтерполяційного полінома. По суті, інтерполяційний многочлен $H_m(t)$ у формулі (9.2) використовується поза відрізком інтерполяції, тобто в даному випадку маємо екстраполяційний поліном. Тому отриманий таким чином метод часто називають *екстраполяційним методом Адамса–Башфорта*, а значення m вказує на кількість кроків методу.

Отримаємо формули Адамса–Башфорта для різної кількості точок m. Для побудови формули необхідно знати m значень наближеного розв'язку у вузлах $t_{n-m+1}, ..., t_{n-1}, t_n$, за якими можна обчислити значення $f_{n-i} = f(t_{n-i}, u_{n-i})$.

У найпростішому випадку для $m = 1$ поліном $H_m(t)$ є константою ($H_1(t) = f_n$), і формула (9.2) фактично задає звичайний однокроковий метод Ейлера.

Для $m = 2$ відповідний поліном Ньютона є лінійною функцією, що проходить через точки $(t_{n-1}, f_{n-1}), (t_n, f_n)$ і записується у вигляді

$$H_1(t) = f_n + (t - t_n) f(t_n, t_{n-1}) = f_n + (t - t_n)(f_n - f_{n-1})/\tau.$$

Підставимо $H_1(t)$ у формулу (9.2):

$$u_{n+1} = u_n + \int_{t_n}^{t_{n+1}} (f_n + (t - t_n)(f_n - f_{n-1})/\tau)\,dt.$$

Виконавши інтегрування, отримаємо таку формулу двокрокового методу для обчислення наближеного розв'язку:

$$u_{n+1} = u_n + \frac{\tau}{2}(3f_n - f_{n-1}). \tag{9.3}$$

На відміну від методу Рунге–Кутта, для якого не потрібно заздалегідь знаходити чисельний розв'язок у початкових вузлах інтервалу інтегрування, екстраполяційний метод Адамса вже для $m = 2$ передбачає попереднє знаходження значення u_1. Для цього необхідно використовувати будь-який однокроковий метод.

Обчислимо похибку апроксимації отриманої формули як нев'язку, яка утворюється у разі підстановки в різницеве рівняння (9.3) значень точного розв'язку:

$$r_{n+1} = -\frac{y(t_{n+1}) - y(t_n)}{\tau} - \frac{1}{2}(f(t_{n-1}, y(t_{n-1})) - 3f(t_n, y(t_n))).$$

Розкладемо в ряд Тейлора в околі точки t_n функції, що входять у формулу для нев'язки. Для точного розв'язку $y(t_n + \tau)$ отримаємо:

$$y(t_n + \tau) = y(t_n) + y'(t_n)\tau + \frac{1}{2}y''(t_n)\tau^2 + \frac{1}{6}y^{(3)}(t_n)\tau^3 + o(\tau)^4.$$

Оскільки $f_{n-1} = f(t_{n-1}, y(t_{n-1})) = y'(t_n - \tau)$, то, розкладаючи в ряд похідну від точного розв'язку, отримаємо:

$$y'(t_n - \tau) = y'(t_n) - y''(t_n)\tau + \frac{1}{2}y^{(3)}(t_n)\tau^2 + O(\tau^3).$$

Знайдемо нев'язку в точці t_{n+1}:

$$r_{n+1} = -(y(t_{n+1}) - y(t_n))/\tau - (y'(t_{n-1}) - 3y'(t_n))/2 = -\frac{5}{12}y^{(3)}(t_n)\tau^2 + O(\tau^3). \quad (9.4)$$

Похибка формули (9.3) визначається останнім виразом.

Формули Адамса–Башфорта для іншої кількості кроків m можна вивести аналогічно. Отримаємо їх за допомогою пакета Mathematica. Спочатку сформуємо таблицю наближених значень розв'язку (t_{n-i}, u_{n-i}), $i = 0, 1, ..., m-1$:

```
In[]:= T = Table [{tn - i*τ, fn-i}, {i, 0, 4}]
Out[]= {{tn, fn}, {tn - τ, f-1+n}, {tn - 2τ, f-2+n}, {tn - 3τ, f-3+n}, {tn - 4τ, f-4+n}}
```

Почнемо з $m = 3$ і виділимо з таблиці перші три значення. Побудуємо відповідний інтерполяційний многочлен:

```
In[]:= Int3 = InterpolatingPolynomial[Take[T, 3], t];
```

і проінтегруємо його згідно з формулою (9.2):

```
In[]:= Ad3= un + Simplify[Integrate[Int3, {t, tn, tn + τ}]]; Print["un+1 = ", Ad3]];
```

$$\text{Out[]= } u_{n+1} = \frac{1}{12}\tau(5f_{n-2} - 16f_{n-1} + 23f_n) + u_n$$

Цей трикроковий явний різницевий метод Адамса–Башфорта, очевидно, можна застосовувати, починаючи з $n = 2$. Два початкові значення можна обчислити, наприклад, методом Рунге-Кутта четвертого порядку. Похибка останньої формули обчислюється так само, як і в попередньому випадку. Визначимо похибку апроксимації отриманої формули як нев'язку, що виникає в разі підстановки точного розв'язку в різницеве рівняння:

$$r_{n+1} = -\frac{y(t_{n+1}) - y(t_n)}{\tau} + \frac{1}{12}(5f_{n-2} - 16f_{n-1} + 23f_n).$$

Розкладемо функції, що входять у вираз для нев'язки, в ряд Тейлора в околі точки t_n і отримаємо:

```
In[]:= yn = y[tn]; yn+1 = Series[y[tn + τ], {τ, 0, 4}]; Print["yn+1 = ", yn+1];
```

$$\text{Out[]= } y_{n+1} = y[t_n] + y'[t_n]\,\tau + \frac{1}{2}y''[t_n]\,\tau^2 + \frac{1}{6}y^{(3)}[t_n]\,\tau^3 + \frac{1}{24}y^{(4)}[t_n]\tau^4 + O[\tau]^5;$$

Оскільки

```
In[]:= fn-1 = f[tn-1, y[tn-1] = y'[tn - τ], fn-2 = f[tn-2, y[tn-2] = y'[tn - 2τ],
```

то, розкладаючи в ряд похідні від точного розв'язку, знаходимо:

```
In[]:= fn = y'[tn]; fn-1 = Series[y[tn - τ], {τ, 0, 4}];
```

$$\text{Out[]= } y_{n+1} = y'[t_n] + y''[t_n]\tau + \frac{1}{2}y^{(3)}[t_n]\tau^2 + \frac{1}{6}y^{(4)}[t_n]\tau^3 + \frac{1}{24}y^{(5)}[t_n]\tau^4 + O[\tau]^5$$

```
In[]:= fn-2 = Series[y'[tn - 2τ], {τ, 0, 4}];
```

$$\text{Out[]= } y'[t_n] + 2y''[t_n]\tau + 2y^{(3)}[t_n]\tau^2 + \frac{4}{3}y^{(4)}[t_n]\tau^3 + \frac{2}{3}y^{(5)}[t_n]\tau^4 + O[\tau]^5$$

І, таким чином, отримаємо нев'язку

```
In[]:= rn+1 = -(yn+1 - yn)/τ + (5fn-2 - 16fn-1 + 23fn)/12; Print["rn+1 =", rn+1];
```

$$\text{Out[]= } r_{n+1} = -\frac{3}{8}y^{(4)}[t_n]\tau^3 + O[\tau]^4$$

яка дорівнює:

$$r_{n+1} = -\frac{3}{8}y^{(4)}(t_n)\tau^3 + O(\tau^4).$$

Звідси випливає, що головний член похибки на одному кроці для трикрокового методу Адамса–Башфорта становить

$$\eta_{n+1} = -\frac{3}{8}y^{(4)}(t_n)\tau^4.$$

Можна узагальнити наведений вище алгоритм для трикрокового методу Адамса–Башфорта і отримати процедуру для виведення формул Адамса–Башфорта з довільною кількістю кроків m:

```
In[]:= ADBA[m_]:= Block[{Tm, i, t, tn},
       Tm = Table[{tn - i*τ, fn-i}, {i, 0, m - 1}];
       Inm = InterpolatingPolynomial[Tm, t];
       Ad = un + Simplify{Integrate{Inm, {t, tn, tn + τ}]];
       Print[m, " -кроковий метод Адамса-Башфорта"]; Print ["un+1 =", Ad]];
```

Для $m = 5$

```
In[]:= ADBA[5]
```

отримаємо п'ятикроковий явний різницевий метод Адамса–Башфорта:

$$u_{n+1} = u_n + \frac{\tau}{720}(251f_{n-4} - 1274f_{n-3} + 2616f_{n-2} - 2774f_{n-1} + 1901f_n). \qquad (9.5)$$

По аналогії з трикроковим методом Адамса–Башфорта можна скласти процедуру для обчислення нев'язки m-крокового методу. Текст цієї процедури наведено нижче; m — порядок, Tk – таблиця коефіцієнтів методу:

```
In[]:= AppAB[m_, Tk_]:= Block[{a, b},
       a = Series[y[tn + τ] - y[tn], {τ, 0, m + 1}]/ τ;
       b = Tk[[1]]y'[tn] + Series[Sum[Tk[[i]] y'[tn - (i - 1)τ], {i, 2, m}], {τ, 0, m}];
       Print ["rn+1 =", b - a]];
```

Використаємо процедуру AppAB для визначення похибки апроксимації п'ятикрокового методу Адамса–Башфорта. Для цього звернемося до неї, вказавши значення порядку метода і таблицю коефіцієнтів формули Адамса–Башфорта:

```
In[]:= AppAB[5, {1901, -2774, 2616, -1274, 251}/720];
```

$$\text{Out[]= } r_{n+1} = -\frac{95}{288} y^{(6)}[t_n]\ \tau^5 + O[\tau]^6$$

Тобто похибка методу дорівнює:

$$r_{n+1} = -\frac{95}{288} y^{(6)}(t_n)\tau^5 + o(\tau^6).$$

Цю формулу, очевидно, можна застосовувати, починаючи з $n = 4$. Звичайно значення функцій, яких не вистачає, обчислюються в точках $t_1, t_2, ..., t_{m-1}$ за методом Рунге–Кутта відповідного порядку. Необхідність використання на початку інтервалу інтегрування інших методів є недоліком багатокрокових методів Адамса. Перевага ж методів Адамса полягає в тому, що на кожному кроці праву частину диференціального рівняння необхідно обчислювати лише один раз, тоді як на кожному кроці однокрокового методу Рунге–Кутта четвертого порядку функцію $f(t, u)$ треба обчислювати чотири рази. Тому слід враховувати співвідношення між кроками обох методів, щоб забезпечити задану точність.

9.2. Інтерполяційні методи Адамса–Мултона

Побудову інтерполяційних m-крокових методів Адамса–Мултона можна здійснити так само, як і побудову екстраполяційних формул Адамса–Башфорта. Для цього слід використовувати вираз (9.3), в якому інтерполяційний поліном $H_m(t)$ побудовано по вузлах інтерполяції $t_{n-m+2}, ..., t_n, t_{n+1}$. Як і в методах Адамса–Башфорта значення m вказує на кількість кроків методу.

Покажемо, як за допомогою пакета Mathematica побудувати інтерполяційні формули Адамса–Мултона різних порядків точності. Введемо таблицю наближених значень розв'язку (t_{n-i}, u_{n-i}), $i = m-1, m-2, ..., -1$:

```
In[]:= T = Table [{tn - i*τ, fn-i}, {i, -1, 4}]
Out[]= {{tn + τ, F-1 + n}, {tn, Fn}, {tn - τ, F-1 + n}, {tn - 2 τ, F-2 + n},
        {tn - 3 τ, F-3 + n}, {tn - 4 τ, F-4 + n}}
```

Припустимо, що $m = 3$, і виділимо з таблиці перші три значення. Побудуємо інтерполяційний многочлен і проінтегруємо його згідно з формулою (9.2). Отримаємо двокроковий неявний різницевий метод Адамса–Мултона:

```
In[]:= In2 = InterpolatingPolynomial[Take[T, 3], t];
       Ad2 = un + Simplify[Integrate[In2, {t, tn, tn + τ}]];
       Print["un+1 = ", Ad2];
```

$$\text{Out[]= } u_{n+1} = -\frac{1}{12}\tau(-f_{n-1} + 8f_n + 5f_{n+1}) + u_n$$

Формули Адамса–Мултона для довільної кількості кроків m можна отримати за допомогою такої процедури:

```
In[]:= ADMU [m_]:= Block[{Tm, i, t, tn},
       Tm = Table[{tn - i*τ, fn-i}, {i, -1, m - 2}];
       Inm = InterpolatingPolynomial[Tm, t];
       Adm = un + Simplify{Integrate{Inm, {t, tn, tn + τ}]];
       Print[m-1, " -кроковий неявний різницевий метод Адамса-Мултона"];
       Print ["un+1 =", Adm]];
       ADMU[4];
```

Розглянемо інтерполяційні формули Адамса–Мултона різних порядків точності. Наприклад, для $m = 2$ маємо трикроковий неявний метод Адамса–Мултона:

```
Out[]= 3-кроковий неявний різницевий метод Адамса-Мултона
```

$$u_{n+1} = \frac{1}{24}\tau(f_{-2+n} - 5f_{-1+n} + 19f_n + 9f_{1+n}) + u_n$$

Процедура для отримання формули похибки апроксимації m-крокового методу Адамса–Мултона наведена нижче і має назву AppAM[m_,Tk_]. Вона подібна до програми оцінювання нев'язки методів Адамса–Башфорта:

```
In[]:= AppAM [m_,Tk_]:= Block[{a, b},
       a = Series[x[tn + τ] - x[tn], {τ, 0, m + 1}]/τ;
       b = Series[Tk[[1]]x'[tn + τ], {τ, 0, m + 1}] + Tk[[2]]x'[tn] +
       Series[ Σ (i=3 to m) Tk[[i]]x'[tn - (i - 2)τ], {τ, 0, m}];
       Print ["rn+1 = ", b - a]];
```

Використовуючи процедуру AppAM, визначимо нев'язку двокрокового методу Адамса–Мултона:

```
In[]:= AppAM[3, {5, 8, -1}/12]
```

$$\text{Out[]= } r_{n+1} = -\frac{1}{24}x^{(4)}[t_n]\tau^3 + O[\tau]^4$$

Те ж саме зробимо для похибки трикрокового методу Адамса–Мултона:

```
In[]:= AppAM[4, {9, 19, -5, 1}/24]
```

$$\text{Out[]= } r_{n+1} = \frac{19}{720}x^{(5)}[t_n]\tau^4 + O[\tau]^5$$

Відмітимо, що у всіх наведених вище інтерполяційних формулах Адамса значення $f_{n+1} = f(t_{n+1}, u_{n+1})$ невідоме, оскільки ще невідоме u_{n+1}. Отже, інтерполяційні методи Адамса–Мултона визначають u_{n+1} неявно. Так, наприклад, формула методу другого порядку точності

$$u_{n+1} = u_n + \frac{\tau}{2}(f(t_n, u_n) + f(t_{n+1}, u_{n+1})) \tag{9.6}$$

насправді є нелінійним рівнянням щодо невідомого значення u_{n+1}. Тому інтерполяційний метод Адамса–Мултона називають *неявним*.

Взагалі якщо рівняння (8.8) не є «жорстким», рівняння (9.6) звичайно не розв'язують, а спільно використовують явну і неявну формули і отримують так званий метод *прогнозу* (*prediction*) і *корекції* (*correction*). Один із найбільш відомих методів прогнозу і корекції являє собою об'єднання методів Адамса–Башфорта і Адамса–Мултона четвертого порядку точності:

$$u_{n+1}^{(\text{pred})} = \frac{1}{24}\tau(-9f_{n-3} + 37f_{n-2} - 59f_{n-1} + 55f_n) + u_n,$$
$$f_{n+1}^{(\text{pred})} = f(t_{n+1}, u_{n+1}^{(\text{pred})}), \tag{9.7}$$
$$u_{n+1}^{(\text{cor})} = \frac{1}{24}\tau(f_{n-2} - 5f_{n-1} + 19f_n + 9f_{n+1}^{(\text{pred})}) + u_n.$$

Зауважимо, що за такої послідовності обчислень цей метод явний. Спочатку за екстраполяційною формулою Адамса–Башфорта обчислюють значення $u_{n+1}^{(\text{pred})}$, *яке є «прогнозом» для* $u_{n+1}^{(\text{cor})}$. *Потім* $u_{n+1}^{(\text{pred})}$ *застосовують для обчислення наближеного значення* $f_{n+1}^{(\text{pred})}$, яке, в свою чергу, використовується в інтерполяційній формулі Адамса–Мултона. Таким чином, формула Адамса–Мултона «коректує» наближення, отримане за допомогою формули Адамса–Башфорта. Слід відзначити, що за такої організації обчислень похибка, яка вноситься за рахунок неточності першої формули (9.7), не змінює порядок похибки обчислень за третьою формулою (9.7).

Приклад 9.1

Знайдемо за комбінованим методом Адамса–Башфорта–Мултона наближений розв'язок задачі Коші

$$y' = 0{,}5(t - y), \quad y(0) = 1. \tag{9.8}$$

Екстраполяційна формула Адамса–Башфорта четвертого порядку точності, яка називається формулою прогнозу, має такий вигляд:

$$u_{n+1}^{(\text{pred})} = u_n + \tau(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3})/24.$$

Уточнене значення обчислюють за інтерполяційною формулою Адамса–Мултона (формулою корекції) з використанням знайденого значення $u_{n+1}^{(\text{pred})}$:

$$u_{n+1}^{(\text{cor})} = u_n + \tau(9f_{n+1}^{(\text{pred})} + 19f_n - 5f_{n-1} + f_{n-2})/24.$$

Для того щоб почати обчислення необхідно заздалегідь знайти за формулою прогнозу три перші значення u_n, $n = 1, 2, 3$. Їх, звичайно, обчислюють методом Рунге–Кутта. У розглянутому нижче прикладі ми зробимо так само. Знайдемо спочатку методом Рунге–Кутта (див. приклад 8.14 із підрозділу 8.5) значення (t_n, u_n), $n = 1, 2, 3$. Потім обчислимо значення (t_n, u_n), $n = 4, 5, \ldots$ — спочатку за формулою Адамса–Башфорта, а потім за формулою Адамса–Мултона. Одночасно для порівняння точності двох методів продовжимо розв'язування методом Рунге–Кутта. Графік отриманого розв'язку наведено на рис. 9.1.

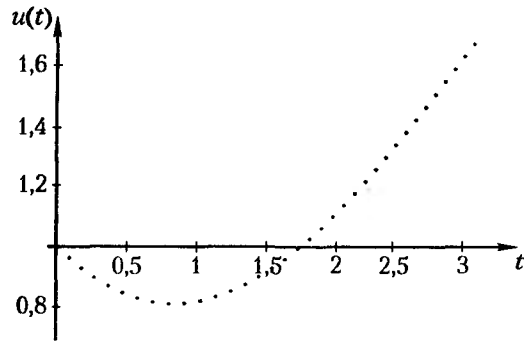


Рис. 9.1. Графік чисельного розв'язку задачі Коші комбінованим методом Адамса–Башфорта і Адамса–Мултона

Точний розв'язок задачі Коші (9.8) легко знайти. Він має такий вигляд: $x = -2 + t + 3e^{-t/2}$. Тому можна оцінити локальну і глобальну похибки наближеного розв'язку, отриманого методами Адамса–Башфорта і Адамса–Мултона. Значення локальних похибок на кожному кроці обчислень e_c і накопиченої глобальної похибки розв'язку e_r наведені в табл. 9.1.

Таблиця 9.1. Значення похибок обчислень

n	t	e_c	e_r
0	0	0	0
1	0,13	0	$2{,}4 \cdot 10^{-8}$
2	0,25	0	$4{,}4 \cdot 10^{-8}$
3	0,38	0	$6{,}2 \cdot 10^{-8}$
4	0,5	$-6{,}8 \cdot 10^{-8}$	$7{,}8 \cdot 10^{-8}$
5	0,63	$-6{,}5 \cdot 10^{-8}$	$9{,}2 \cdot 10^{-8}$
7	0,75	$-6{,}0 \cdot 10^{-8}$	$1{,}0 \cdot 10^{-7}$
8	0,88	$-5{,}7 \cdot 10^{-8}$	$1{,}1 \cdot 10^{-7}$
9	1,0	$-5{,}3 \cdot 10^{-8}$	$1{,}2 \cdot 10^{-7}$
10	1,1	$-5{,}0 \cdot 10^{-8}$	$1{,}3 \cdot 10^{-7}$
11	1,3	$-4{,}7 \cdot 10^{-8}$	$1{,}3 \cdot 10^{-7}$
12	1,4	$-4{,}4 \cdot 10^{-8}$	$1{,}4 \cdot 10^{-7}$
13	1,5	$-4{,}1 \cdot 10^{-8}$	$1{,}4 \cdot 10^{-7}$
14	1,6	$-3{,}9 \cdot 10^{-8}$	$1{,}4 \cdot 10^{-7}$
15	1,8	$-3{,}7 \cdot 10^{-8}$	$1{,}5 \cdot 10^{-7}$
16	1,9	$-3{,}4 \cdot 10^{-8}$	$1{,}5 \cdot 10^{-7}$
17	2,0	$-3{,}2 \cdot 10^{-8}$	$1{,}5 \cdot 10^{-7}$
18	2,1	$-3{,}0 \cdot 10^{-8}$	$1{,}5 \cdot 10^{-7}$
19	2,3	$-2{,}8 \cdot 10^{-8}$	$1{,}5 \cdot 10^{-7}$
20	2,4	$-2{,}7 \cdot 10^{-8}$	$1{,}5 \cdot 10^{-7}$

Дійсна накопичена похибка в разі обчислень за методами Адамса–Башфорта і Рунге–Кутта практично одна й та сама.

9.3. Лінійні багатокрокові різницеві методи

Розглянуті в підрозділах 9.1 і 9.2 методи Адамса–Башфорта і Адамса–Мултона є лінійними багатокроковими різницевими методами, які в загальному випадку можна записати в такому вигляді:

$$\frac{1}{\tau}\sum_{i=0}^{m} a_i u_{n+1-i} = \sum_{i=0}^{m} b_i f(t_{n+1-i}, u_{n+1-i}), \quad n = m-1, m, m+1, \ldots, \tag{9.9}$$

де a_i, b_i — коефіцієнти, що не залежать від n, причому $a_0 \neq 0$. Вираз (9.9) слід розглядати як рекурентне співвідношення, що визначає нове значення u_{n+1} через знайдені раніше значення $u_n, u_{n-1}, u_{n-2}, \ldots, u_{n-m+1}$.

Обчислення починаються з $n = m-1$, тобто з виразу

$$\sum_{i=1}^{m} a_i u_{m-i} = \tau \sum_{i=1}^{m} b_i f(t_{m-i}, u_{m-i}).$$

Звідси випливає, що для початку розрахунку необхідно задати m початкових значень наближеного розв'язку $u_0, u_1, u_2, \ldots, u_{m-1}$. Значення u_0 відоме з початкової умови задачі, а значення $u_1, u_2, \ldots, u_{m-1}$ можна обчислити, наприклад, за допомогою методу Рунге–Кутта. Надалі вважатимемо, що початкові значення $u_0, u_1, u_2, \ldots, u_{m-1}$ вже відомі й за ними обчислюється значення u_{n+1}.

Метод (9.9) називається *явним*, якщо $b_0 = 0$, і, отже, шукане значення u_{n+1}, знаходиться явно через попередні значення $u_n, u_{n-1}, u_{n-2}, \ldots, u_{n-m+1}$. Інакше (тобто, коли $b_0 \neq 0$) метод називається *неявним*. Тоді для знаходження u_{n+1} буде необхідно розв'язувати нелінійне рівняння (9.9), оскільки в нього буде входити також $f(t_m, u_m) = f(t_{n+1}, u_{n+1})$.

Похибкою апроксимації розв'язку диференціального рівняння (8.10) або нев'язкою різницевого методу (9.9) називається функція

$$r_{n+1} = \frac{1}{\tau}\sum_{i=0}^{m} a_i y_{n-i+1} - \sum_{i=0}^{m} b_i f(t_{n-i+1}, y_{n-i+1}), \quad n = m, m+1, \ldots, \tag{9.10}$$

яку отримують у результаті підстановки точного розв'язку $y(t)$ диференціальної задачі (8.8) в різницеве рівняння (9.9).

Оскільки коефіцієнти рівняння (9.9) знаходяться з точністю до множника, то для усунення невизначеності вважатимемо, що виконано умову

$$\sum_{i=0}^{m} b_i = 1, \tag{9.11}$$

тобто права частина різницевого рівняння (9.9) апроксимує праву частину відповідного диференціального рівняння (8.10), або $f(t, y)$.

З'ясуємо питання щодо похибки апроксимації, коли $\tau \to 0$, залежно від вибору коефіцієнтів a_i, b_i, $i = 0, 1, ..., m$. Припустимо при цьому, що всі функції мають необхідну гладкість.

Розкладаючи функції $y(t_{n+1-i}) = y(t_{n+1} - i\tau)$ в точці $t = t_{n+1}$ у ряд Тейлора, отримаємо:

$$y(t_{n+1-i}) = \sum_{j=0}^{p} \frac{(-i\tau)^j y^{(j)}(t_{n+1})}{j!} + O(\tau^{p+1}), \quad i = 1, 2, ..., p$$

і

$$f(t_{n+1-i}, y_{n+1-i}) = y'(t_{n+1} - i\tau) = \sum_{j=1}^{p-1} \frac{(-i\tau)^j y^{(j+1)}(t_{n+1})}{j!} + O(\tau^p).$$

Підставивши ці вирази у формулу (9.10), отримаємо оцінку похибки апроксимації:

$$r_{n+1} = -\frac{1}{\tau}\left(\sum_{i=1}^{m} a_i \sum_{j=0}^{p} \frac{(-i\tau)^j y^{(j)}(t_{n+1})}{j!} + a_0 y(t_{n+1})\right) + b_0 y'(t_{n+1}) + \\ + \sum_{i=1}^{m} b_i \sum_{j=0}^{p-1} \frac{(-i\tau)^j y^{(j+1)}(t_{n+1})}{j!} + O(\tau^p).$$

Виділимо в кожній з сум доданки для випадку $j = 0$ і згрупуємо їх:

$$r_{n+1} = -\frac{1}{\tau}\left(\sum_{i=1}^{m}\sum_{j=1}^{p} a_i \frac{(-i\tau)^{j-1} y^{(j)}(t_{n+1})}{j!} + y(t_{n+1})\sum_{i=0}^{m} a_i\right) + \\ + \sum_{i=1}^{m}\sum_{j=1}^{p-1} b_i \frac{(-i\tau)^j y^{(j+1)}(t_{n+1})}{j!} + y'(t_{n+1})\sum_{i=0}^{m} b_i + O(\tau^p).$$

Скоротимо члени першої суми на τ і змінимо порядок обчислень під знаком подвійних сум:

$$r_{n+1} = -\frac{1}{\tau}\left(\sum_{i=0}^{m} a_i\right) y(t_{n+1}) + \sum_{j=1}^{p}\sum_{i=1}^{m} i a_i \frac{(-i\tau)^j y^{(j)}(t_{n+1})}{j!} + \\ + y'(t_{n+1})\sum_{i=0}^{m} b_i + \sum_{j=1}^{p-1}\sum_{i=1}^{m} b_i \frac{(-i\tau)^j y^{(j)}(t_{n+1})}{j!} + O(\tau^p).$$

У першій подвійній сумі виділимо доданки для $j = 1$ і об'єднаємо їх із подібними членами:

$$r_{n+1} = -\frac{1}{\tau}\left(\sum_{i=0}^{m} a_i\right) y(t_{n+1}) + \left[b_0 + \sum_{i=1}^{m} (i a_i + b_i)\right] y'(t_{n+1}) + \sum_{j=1}^{p}\sum_{i=1}^{m} i a_i \frac{(-i\tau)^j y^{(j)}(t_{n+1})}{j!} + \\ + \sum_{j=1}^{p-1}\sum_{i=1}^{m} b_i \frac{(-i\tau)^j y^{(j)}(t_{n+1})}{j!} + O(\tau^p).$$

Зведемо подвійні суми до однакових інтервалів зміни індексів:

$$r_{n+1} = -\frac{1}{\tau}\left(\sum_{i=0}^{m} a_i\right) y(t_{n+1}) + \left[b_0 + \sum_{i=1}^{m}(ia_i + b_i)\right] y'(t_{n+1}) + \\ + \sum_{j=1}^{p-1} \frac{y^{(j+1)}(t_{n+1})}{j!}(-\tau)^j \sum_{i=1}^{m}\left(\frac{ia_i}{j+1} + b_i\right) i^j + O(\tau^p). \tag{9.12}$$

Звідси видно, що похибка апроксимації має порядок m за таких умов:

$$\sum_{i=0}^{m} a_i = 0, \tag{9.13}$$

$$b_0 + \sum_{i=1}^{m}(ia_i + b_i) = 0, \tag{9.14}$$

$$\sum_{i=1}^{m} i^j \left(\frac{ia_i}{j+1} + b_i\right) = 0, \quad j = 1, 2, ..., p-1. \tag{9.15}$$

Разом із умовою нормування (9.11) рівняння (9.13), (9.14), (9.15) утворюють систему з $m+2$ лінійних алгебраїчних рівнянь відносно $2(m+1)$ невідомих $a_1, ..., a_m, b_1, ..., b_m$.

Можна дещо спростити цю систему, врахувавши в рівнянні (9.14) умову нормування (9.11). Тоді отримаємо систему рівнянь:

$$\sum_{i=1}^{m} ia_i = -1, \quad \sum_{i=1}^{m} i^j \left(\frac{ia_i}{j+1} b_i\right) = 0, \quad j = 1, 2, ..., p-1, \tag{9.16}$$

яка містить p рівнянь і $2m$ невідомих $a_0, a_1, ..., a_m, b_0, b_1, ..., b_m$.

Коефіцієнти a_0, b_0 обчислюються за такими формулами:

$$a_0 = \sum_{i=1}^{m} a_i, \quad b_0 = 1 - \sum_{i=1}^{m} b_i. \tag{9.17}$$

Щоб система (9.16) не була перевизначена, необхідне виконання умови $p \leqslant 2m$. Для цього порядок апроксимації лінійних m-крокових різницевих методів не повинен перевищувати $2m$.

Отже, щонайвищий досяжний порядок апроксимації неявних m-крокових методів дорівнює $2m$, а явних — $2m-1$.

Використовуючи рівняння (9.16) і (9.17), отримаємо формули для деяких лінійних неявних різницевих методів.

♦ Лінійний двокроковий неявний метод четвертого порядку апроксимації:

$$u_{n+1} = u_{n-1} + \frac{\tau}{3}(f_{n-1} + 4f_n + f_{n+1}). \tag{9.18}$$

♦ Лінійний трикроковий неявний метод шостого порядку апроксимації:

$$u_{n+1} = u_{n-2} + \frac{27}{11}u_{n-1} - \frac{27}{11}u_n + \frac{3\tau}{11}(f_{n-2} + 9f_{n-1} + 9f_n + f_{n+1}). \tag{9.19}$$

♦ Лінійний чотирикроковий неявний метод восьмого порядку апроксимації:

$$u_{1+n} = u_{n-3} + \frac{32}{5}u_{n-2} - \frac{32}{5}u_n + \frac{84\tau}{350}(f_{n-3} + 16f_{n-2} + 36f_{n-1} + 16f_n + f_{n+1}). \tag{9.20}$$

Щоб отримати формули для явних лінійних багатокрокових різницевих методів, достатньо в рівнянні (9.15) покласти $b_0 = 0$ і використати таку умову нормування:

$$\sum_{i=1}^{m} b_i = 1. \tag{9.21}$$

Після спрощень рівняння (9.13), (9.14), (9.15) матимуть такий вигляд:

$$\sum_{i=0}^{m} a_i = 0, \quad \sum_{i=1}^{m} i a_i = 0. \tag{9.22}$$

Системи (9.21) та (9.22) містять $p+2$ рівнянь з $2m+1$ невідомими. Щоб система не була перевизначена, необхідне виконання умови $p+2 \leqslant 2m+1$. Ця вимога означає, що порядок апроксимації явних лінійних m-крокових різницевих методів не може перевищувати $2m-1$.

Використовуючи рівняння (9.21), (9.22), отримаємо формули для деяких явних лінійних багатокрокових різницевих методів.

♦ Лінійний двокроковий явний різницевий метод третього порядку:

$$u_{n+1} = 5u_{n-1} - 4u_n + 2\tau(f_{n-1} + 2f_n). \tag{9.23}$$

♦ Лінійний трикроковий явний різницевий метод п'ятого порядку:

$$u_{n+1} = 10u_{n-2} + 9u_{n-1} - 18u_n + 3\tau(f_{n-2} + 6f_{n-1} + 3f_n). \tag{9.24}$$

♦ Лінійний чотирикроковий явний різницевий метод сьомого порядку:

$$\begin{aligned} u_{n+1} = {} & \frac{47}{3}u_{n-3} + 64u_{n-2} - 36u_{n-1} - \\ & - \frac{128}{3}u_n + \frac{140\tau}{35}(f_{n-3} + 12f_{n-2} + 18f_{n-1} + 4f_n). \end{aligned} \tag{9.25}$$

Нагадаємо, що найвищий досяжний порядок апроксимації неявних m-крокових методів дорівнює $2m$, а явних — $2m-1$.

Виявляється, що методи найвищого порядку апроксимації практично непридатні для розрахунків, оскільки вони є нестійкими. Питання стійкості і збіжності різницевих методів розглядатимуться нижче і в наступному розділі.

Відзначимо, що, коли в системі (9.15) відкинути останні k рівнянь, $k = 1, 2, ..., p-1$, отримаємо умови, які забезпечують порядок апроксимації $p-k$.

Щоб отримати методи Адамса–Мултона відповідно до формули (9.6), у загальній формулі (9.9) для лінійних різницевих методів необхідно покласти

$$a_0 = 1, \quad a_1 = -1, \quad a_i = 0, \quad i = 2, ..., m.$$

Тоді умови (9.13), (9.14), (9.15) p-го порядку апроксимації набувають вигляду:

$$(j+1)\sum_{i=1}^{m} i^j b_i = 1, \quad j = 1, 2, ..., p-1, \quad b_0 = 1 - \sum_{i=1}^{m} b_i, \tag{9.26}$$

$$(j+1)\sum_{i=1}^{m} i^j b_i = 1, \quad j = 1, 2, ..., p-1, \quad b_0 = 1. \tag{9.27}$$

Звідси видно, що найвищий порядок апроксимації m-крокового методу Адамса–Мултона дорівнює $m+1$, у той час як найвищий порядок апроксимації методу Адамса–Башфорта, оскільки $b_0 = 0$, дорівнює m.

9.4. Методи розв'язання систем диференціальних рівнянь і рівнянь вищих порядків

Розглянуті в попередньому розділі формули розв'язання задачі Коші для диференціальних рівнянь першого порядку легко переносяться на системи диференціальних рівнянь, аналогічно тому, як це було зроблено в підрозділі 8.7 для методів Рунге–Кутта.

9.4.1. Розв'язання систем рівнянь

У векторних позначеннях, використаних у формулі (8.7), задача Коші має такий вигляд:

$$\mathbf{Y}' = \mathbf{F}(t, \mathbf{Y}), \quad \mathbf{Y}(t_0) = \mathbf{Y}_0. \tag{9.28}$$

Тому для розв'язання векторного рівняння (9.28) можна застосувати будь-який із чисельних методів, описаних у цьому розділі. Наведемо приклад векторних формул розв'язання системи диференціальних рівнянь явним методом Адамса–Башфорта. Введемо такі позначення:

$$\mathbf{U}^n = \begin{pmatrix} u_1^n \\ u_2^n \\ \dots \\ u_p^n \end{pmatrix}, \quad \mathbf{F}^n = \begin{pmatrix} f_1(t_n, \mathbf{U}^n) \\ f_2(t_n, \mathbf{U}^n) \\ \dots \\ f_p(t_n, \mathbf{U}^n) \end{pmatrix}. \tag{9.29}$$

Тоді явний метод Адамса–Башфорта для системи диференціальних рівнянь матиме такий вигляд:

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \tau\sum_{j=1}^{m} b_j \mathbf{F}^{n-j+1}, \quad n = m-1, m, \ldots . \tag{9.30}$$

Формули (9.30) за формою збігаються з явною формулою Адамса–Башфорта для одного рівняння, скажімо, з виразом (9.5). Наведемо приклади їх застосування.

Приклад 9.2

Розв'яжемо задачу Коші, використовуючи пакет Mathematica:

$$\begin{aligned} y_1' &= e^{-(y_1^2+y_2^2)} + 2t, \\ y_2' &= 2y_1^2 + y_2, \\ y_1(0) &= 0{,}5, \quad y_2(0) = 1. \end{aligned} \tag{9.31}$$

Спочатку задамо початкові умови і крок інтегрування τ:

```
In[]:= t0 = 0; y0 = {0.5, 1}; τ = 0.1;
```

Опишемо вектор-функцію правих частин рівнянь:

```
In[]:= F[t_,y_]:= {E ^-(y[[1]]^2 + y[[2]]^2) + 2 t, 2 y[[1]]^2 + y[[2]]};
```

Наближений розв'язок обчислюватимемо за чотирикроковою явною формулою Адамса. Введемо вектор коефіцієнтів формули Адамса.

```
In[]:= b = {55/24, -59/24, 37/24, -9/24};
```

Задамо кількість кроків інтегрування і опишемо вектор наближеного розв'язку **U**:

```
In[]:= n = 10;
       Array[U, n + 1, 0];
```

Значення наближеного розв'язку в перших чотирьох точках сітки, які необхідні для початку розрахунку за формулою Адамса, знайдемо методом Рунге–Кутта (див. підрозділ 8.6, табл. 8.5):

```
In[]:= U[0] = {0.5, 1.};
       U[1] = {0.534027, 1.16116};
       U[2] = {0.579456, 1.34819};
       U[3] = {0.637922, 1.56755};
```

Далі йде процедура розрахунку наближеного розв'язку з кроком τ на n кроках:

```
In[]:= Do [U[i+1] = U[i] + t Sum[b[[j]]F[t0 + τ(i + 1 - j), U[i + 1 - j]], {j, 1, 4}], {i, 3, n}]
```

Виведемо таблицю перших десяти значень наближеного розв'язку задачі (9.31):

```
In[]:= TA = Table[{t0 + (i - 1)*τ, U[i - 1][[1]], U[i - 1][[2]]}, {i, 10}];
       Print["Таблиця 9.2."]:
       TableForm[TA, TableHeadings → {Automatic → {"tn", "u1", "u2"}}]
```

Отримані результати зведені у табл. 9.2.

Таблиця 9.2. Чисельний розв'язок системи диференціальних рівнянь

n	t_n	u_1	u_2
1	0	0,5	1,0
2	0,1	0,534027	1,16116
3	0,2	0,579456	1,34819
4	0,3	0,637922	1,56755
5	0,4	0,71181	1,82747
6	0,5	0,80326	2,1395
7	0,6	0,913684	2,51856
8	0,7	1,04369	2,98382
9	0,8	1,19354	3,55952
10	0,9	1,36344	4,27613

9.4.2. Розв'язання рівнянь вищих порядків

Одним із основних способів чисельного розв'язання задачі Коші для диференціальних рівнянь вищих порядків є їх зведення до системи рівнянь першого порядку. Розглянемо задачу Коші для рівняння другого порядку:

$$y'' = f(t, y', y), \quad y(t_0) = y_0, \; y'(t_0) = y'_0. \tag{9.32}$$

Введемо нову змінну $x = y'$, тоді маємо $x' = y''$. Підставивши їх у рівняння (9.32), отримаємо еквівалентну систему рівнянь:

$$\begin{cases} y' = x, \\ x' = f(t, x, y). \end{cases} \tag{9.33}$$

Початкові умови для неї перепишемо у вигляді

$$y(t_0) = y_0, \quad x(t_0) = y'_0. \tag{9.34}$$

Для розв'язання цієї задачі Коші можна застосувати будь-який із розглянутих у цьому розділі чисельних методів. Для пояснення наведемо приклад чисельного розв'язання задачі Коші для рівняння другого порядку.

Приклад 9.3

Задана задача Коші:

$$y'' + 6y' + 9y = 0, \quad y(0) = 4, \; y'(0) = -4.$$

Замінимо її еквівалентною задачею Коші для системи двох рівнянь першого порядку:

$$\begin{cases} y'_1 = y_2, \\ y'_2 = -6y_2 - 9y_1, \\ y_1(0) = 4, \quad y_2(0) = -4. \end{cases} \tag{9.35}$$

Отриману систему рівнянь розв'яжемо, використовуючи метод Мілна:

$$u_{n+1}^{(\text{pred})} = u_{n-3} + \frac{4}{3}\tau(2f_{n-2} - f_{n-1} + 2f_n);$$
$$u_{n+1}^{(\text{cor})} = u_{n-1} + \frac{1}{3}\tau(f_{n-1} + 4f_n + f_{n+1}^{(\text{pred})}).$$

Як і в попередньому прикладі, перейдемо до векторного запису системи рівнянь. Для цього введемо вектор значень сіткових функцій у формулах прогнозу і корекції

$$\tilde{\mathbf{U}}^{n+1} = \begin{bmatrix} \tilde{u}_1^{n+1} \\ \tilde{u}_2^{n+1} \end{bmatrix}, \quad \mathbf{U}^{n+1} = \begin{bmatrix} u_1^{n+1} \\ u_2^{n+1} \end{bmatrix},$$

вектор правих частин рівнянь

$$\mathbf{F}^n = \begin{bmatrix} u_2^n \\ -9u_1^n - 6u_2^n \end{bmatrix},$$

вектор коефіцієнтів формул Мілна

$$\mathbf{b}^{(\text{pred})} = \left[\frac{8}{3}, -\frac{4}{3}, \frac{8}{3}\right], \quad \mathbf{b}^{(\text{cor})} = \left[\frac{1}{3}, \frac{4}{3}, \frac{1}{3}\right].$$

Тоді векторні формули прогнозу і корекції для системи рівнянь (9.35) матимуть вигляд:

$$\tilde{\mathbf{U}}^{n+1} = \mathbf{U}^{n-3} + \frac{4\tau}{3}(2\mathbf{F}^n - \mathbf{F}^{n-1} + 2\mathbf{F}^{n-2}),$$
$$\mathbf{U}^{n+1} = \mathbf{U}^{n-1} + \frac{\tau}{3}(\mathbf{F}^{n+1} + 4\mathbf{F}^n + \mathbf{F}^{n-1}). \qquad (9.36)$$

Початкові значення наближеного розв'язку в перших чотирьох точках сітки, необхідні для початку розрахунку за формулою прогнозу Мілна, знайдемо методом Рунге–Кутта (див. підрозділ 8.4.)
Задамо початкові умови:

```
In[]:= t0 = 0; y0 = {4, - 4}; τ = 0.1;
```

Визначимо вектор-функцію правих частин рівнянь:

```
In[]:= F[t_, y_]: = {y[[2]], - (9*y[[1]] + 6*y[[2]])};
```

Задамо кількість кроків інтегрування і опишемо вектор наближеного розв'язку **U**:

```
In[]:= n = 10; Array[U, n + 1, 0]; Array[V, n + 1, 0];
       bp = {8/3, -4/3, 8/3}; bc = {1/3, 4/3, 1/3}
```

Нижче наведено програму обчислення наближеного розв'язку методом Рунге–Кутта.

```
In[]:= U[0] = y0; V[0] = y0; t = t0;
       DO [K1 = F[t, U[i]]; K2 = F[t + 0.5τ, U[i] + 0.5τK1];
           K3 = F[t + 0.5τ, U[i] + 0.5τK2]; K4 = F[t + τ, U[i] + τK3];
           U[i + 1] = U[i] + (τ/6)[K1 + 2K2 + 2K3 + K4]; V[i + 1] = U[i + 1];
           t = t + τ, {i, 0, 3} ];
```

Виведемо таблицю перших чотирьох значень наближеного розв'язку задачі (9.36):

```
In[]:= TR = Table[{t0 + (i - 1)*τ, U[i - 1][[1]], U[i - 1][[2]]}, {i, 4}}];
       Print["Таблица 9.3."]; TableForm[TR, TableHeadings → {Automatic, {"i", "t", "u1", "u2"}}]
```

Отримані дані представлені в табл. 9.3.

Таблиця 9.3. Перші чотири значення розв'язку, необхідні для застосування методу Мілна

i	t_n	u_1	u_2
1	0	4	–4
2	0,1	3,55575	–4,74055
3	0,2	3,07311	–4,82859
4	0,3	2,6018	–4,5526

Тепер можна продовжити розв'язування методом Мілна:

```
In[]:= Do[V[i + 1] = U[i - 3] + τ Σ(j=1 to 3) bp[[j]]F[t0 + τ(i + 1 - j), U[i + 1 - j]];
                    U[i + 1] = V[i + 1];
       U[i + 1] = U[i - 1] + τ Σ(j=0 to 2) bc[[j + 1]]F[t0 + τ(i + 1 - j),
                    U[i + 1 - j]] {i, 3, n}];
       TM = Table[{t0 + (i - 1)*τ, U[i - 1][[1]], U[i - 1][[2]]} {i, n}];
```

Для даної задачі можна знайти аналітичний розв'язок:

```
In[]:= EQ = Dsolve[{x'[z] == y[z],
                    y'[z] + 9x[z] + 6y[z] == 0,
                    x[0] == 4, y[0] == 4}
                   {x[z], y[z]}, z]
Out[]= {x[z] → 4e^(-3z)(1 + 2z), y[z] → -4e^(-3z)(1 + 6z)}
```

Запишемо отриманий аналітичний розв'язок у вигляді функцій:

```
In[]:= x1[z_] = EQ[[1, 1, 2]];
       x2[z_] = EQ[[1, 2, 2]];
```

Використовуючи їх, можна обчислити дійсну накопичену (глобальну) похибку розв'язку системи методом Мілна.

У табл. 9.4 наведено значення накопичених похибок для кожної шуканої функції.

Таблиця 9.4. Значення накопиченої похибки чисельного розв'язку задачі методом Мілна

i	t	e_{r1}	e_{r2}
1	0	0	0
2	0,1	–0,00018	0,00069
3	0,2	–0,00024	0,00095
4	0,3	–0,00024	0,00098
5	0,4	0,0004	–0,0015
6	0,5	$-1{,}1 \cdot 10^{-6}$	$5{,}1 \cdot 10^{-6}$
7	0,6	0,00056	–0,0022
8	0,7	0,000019	–0,00012
9	0,8	0,0005	–0,002
10	0,9	–0,000047	0,00011

9.5. Стійкість методів Адамса–Башфорта і Адамса–Мултона

Аналіз стійкості багатокрокових методів розв'язання диференціальних рівнянь відрізняється від аналізу стійкості однокрокових методів, який проводиться з використанням умови (8.58) (див. підрозділ 8.7). Досліджуючи стійкість різницевих методів, звичайно розглядають модельне рівняння

$$\frac{dy}{dt} = \lambda y, \tag{9.37}$$

де λ — довільне комплексне число. Для модельної задачі (9.37) багатокрокові методи легко зводяться до різницевих рівнянь вигляду:

$$\frac{1}{\tau}\sum_{i=0}^{m} a_i u_{n+1-i} - \sum_{i=0}^{m} \tau\lambda u_{n+1-i} = \sum_{i=0}^{m} c_i u_{n+1-i} = 0, \quad n = m-1, m+1, \ldots, \tag{9.38}$$

де $c_i = a_i - \tau\lambda b_i$.

Тому аналіз стійкості багатокрокових методів (9.9) зручно проводити за допомогою добре досліджених методів аналізу стійкості різницевого рівняння (9.38). Стійкість різницевого лінійного однорідного рівняння з постійними коефіцієнтами

$$c_0 u_{n+m} + c_1 u_{n+m-1} + c_2 u_{n+m-2} + \ldots + c_m u_n = 0, \tag{9.39}$$

якому задовольняє послідовність $u_j = ez^j$ $(z \neq 0, e \neq 0)$, визначається коренями його характеристичного рівняння:

$$C(z) = c_0 z^m + c_1 z^{m-1} + c_2 z^{m-2} + \ldots + c_m = 0, \tag{9.40}$$

які повинні задовольняти умову

$$|z_i| < 1, \quad i = 1, 2, \ldots, m. \tag{9.41}$$

Тільки в цьому випадку розв'язком різницевого рівняння (9.39) є величина

$$u_n = e_1 z_1^n + e_2 z_2^n + \ldots + e_m z_m^n,$$

яка за довільних констант e_i наближається до нуля, якщо $n \to \infty$. Цей же розв'язок прямує до деякого граничного значення e_i і тоді, якщо серед коренів (9.41) характеристичного рівняння є один корінь $|z_i| = 1$. Таким чином, для стійкості розв'язку задачі Коші необхідно, щоб корені характеристичного рівняння (9.41) лежали всередині одиничного кола $\{z : |z_i| < 1\}$ і ті з коренів, модулі яких дорівнюють одиниці $\{z : |z_i| = 1\}$, повинні бути простими.

Розглянемо умову стійкості методу Адамса–Башфорта другого порядку (9.4):

$$u_{n+1} = u_n + \frac{\tau}{2}(3f_n - f_{n-1}).$$

Згідно з рівнянням (9.37) маємо значення $f_n = \lambda u_n$ та $f_{n-1} = \lambda u_{n-1}$. Відповідне різницеве рівняння (9.38) і його характеристичне рівняння (9.39) набувають вигляду:

$$u_{n+1} - \left(1 + \frac{3}{2}\tau\lambda\right)u_n + \frac{1}{2}\tau\lambda u_{n-1} = 0,$$

$$z^2 - \left(1 + \frac{3}{2}\tau\lambda\right)z + \frac{1}{2}\tau\lambda = 0,$$

звідки з умови стійкості (9.41) маємо

$$\tau|\lambda| < \frac{2}{2} = 1. \tag{9.42}$$

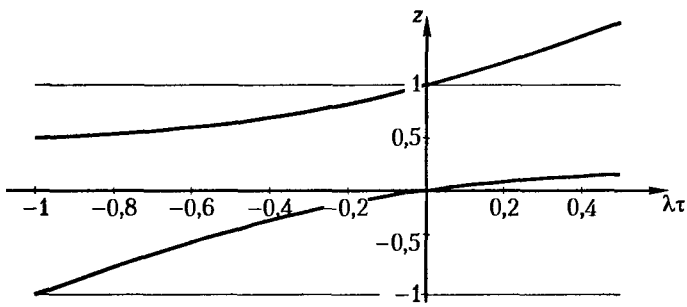


Рис. 9.2. Область стійкості мвтоду Адамса–Башфорта другого порядку

Для дійсних λ стійкість методу Адамса–Башфорта другого порядку в два рази менша, ніж стійкість методу Ейлера першого порядку.

За досить великих за модулем від'ємних значень z поліном $C(z)$ (9.40) має знак $(-1)^m$. Тоді значення полінома $C(-1)$ повинне мати той же знак, інакше інтервалі $(-\infty, -1)$ знайдеться корінь, який не задовольняє умову (9.41). Безпосереднє обчислення значення

$$C(-1) = \sum_{i=0}^{m} c_i(-1)^i = (-1)^m \sum_{i=0}^{m} (-1)^{m+i}$$

призводить до нерівності

$$\sum_{i=0}^{m} c_i(-1)^{m+i} > 0, \tag{9.43}$$

що накладає додаткові обмеження на значення коефіцієнтів c_i, необхідних для виконання умови стійкості (9.41).

Рівняння (9.38) з урахуванням значень коефіцієнтів для явних методів Адамса–Башфорта $a_0 = 1, a_1 = -1, a_i = 0,\ i = 2, ..., m$ набуває вигляду:

$$y_{n+m} - y_{n+m-1} - \tau\lambda \sum_{i=0}^{m-1} b_i y_{n+i} = 0,$$

і йому відповідає характеристичне рівняння

$$z^m - z^{m-1} - \tau\lambda\sum_{i=0}^{m-1} b_i z^i = 0. \tag{9.44}$$

У разі від'ємних і дійсних значень λ для виконання умови (9.41) коефіцієнти характеристичного рівняння (9.44) повинні задовольняти нерівність (9.43), яка набуває вигляду:

$$2 - \tau\lambda\sum_{i=0}^{m-1} b_i(-1)^{m+i} > 0. \tag{9.45}$$

Можна показати [26], що для явного методу Адамса–Башфорта m-го порядку справедливі такі співвідношення:

- значення $b_i(-1)^{m+i}$ від'ємні для всіх i;
- значення $\left|\sum_{i=0}^{m-1} b_i(-1)^{m+i}\right|$ росте зі збільшенням m.

Тому для виконання нерівності (9.41) необхідне виконання умови

$$\tau|\lambda| < \frac{2}{\sum_{i=0}^{m-1}|b_i|}, \tag{9.46}$$

де b_i — коефіцієнти, що необхідні для знаходження середнього значення похідної на декількох кроках обчислень.

Наприклад, для явного методу Адамса–Башфорта третього порядку

$$u_{n+1} = u_n + \frac{\tau}{12}(5f_{n-2} - 16f_{n-1} + 23f_n),$$

умовою стійкості буде

$$\sum_{i=0}^{2}|b_i| = \frac{23+16+5}{12} = 3,3666,$$
$$\tau|\lambda| = \frac{2}{3,3666} = 0,545. \tag{9.47}$$

Аналогічно можна показати, що для явного методу Адамса–Башфорта четвертого порядку умовою стійкості є нерівність

$$\tau|\lambda| \leqslant 0,3. \tag{9.48}$$

Таке зменшення кроку з метою забезпечення стійкості розв'язку є своєрідною платою за зменшення об'єму обчислень. Таким чином, у методі Рунге–Кутта четвертого порядку крок можна вибирати майже в 10 разів більшим, ніж

у відповідному методі Адамса–Башфорта, що ставить під сумнів переваги останнього з точки зору об'єму необхідних обчислень. Із збільшенням порядку методу Адамса–Башфорта допустимий крок обчислень зменшується (рис. 9.3). Нагадаємо, що у разі застосування методів Рунге–Кутта, навпаки, збільшення порядку методу веде до розширення області його стійкості.

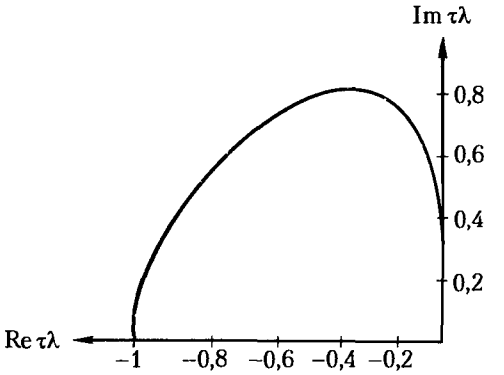


Рис. 9.3. Області стійкості явних мвтодів Адамса–Башфорта

Годографи стійкості, наведені на рис. 9.3, отримані на основі виразу (9.46).

Про стійкість методу певного порядку для вибраного кроку обчислень судять за положенням точки на комплексній площині $\tau\lambda$ відносно відповідної кривої годографа. Якщо точка потрапляє у внутрішню замкнуту область, метод стійкий, якщо в зовнішню — ні. Слід зазначити, що процедури прогнозу-корекції типу (9.9) не збільшують стійкості використаних в них методів, якщо рівняння корекції не розв'язується як нелінійне рівняння відносно u_{n+1}.

Неявні багатокрокові методи Адамса–Мултона, наприклад методи (9.6) і (9.7), для реалізації яких необхідне розв'язання нелінійних рівнянь, мають значно більшу стійкість, властиву в цілому неявним методам.

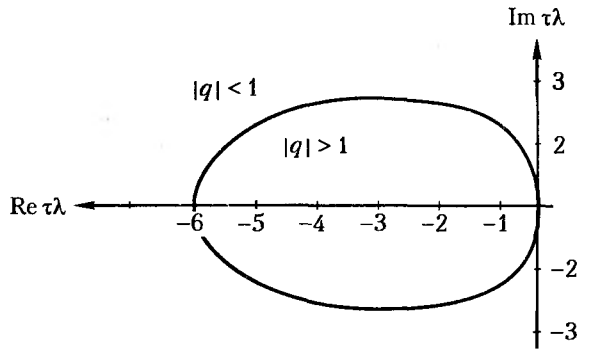


Рис. 9.4. Область стійкості нвявного методу Адамса–Мултона третього порядку

Можна показати, що, коли $\tau\lambda < 0$, для методу третього порядку умовою стійкості є $\tau|\lambda| < 6$ (рис. 9.4), а для методу четвертого порядку — $\tau|\lambda| < 3$. Докладніше стійкість неявних методів розглядається в наступному розділі.

9.6. Розв'язання задачі Коші в пакеті Mathematica багатокроковими різницевими методами

У пакеті Mathematica для розв'язання звичайних диференціальних рівнянь у формі Коші передбачено стандартний оператор NDSolve, що має формат:

```
NDSolve[{eqn1, eqn2, ...}, y, {t, tmin, tmax}]
```

і який можна адаптувати на конкретний метод із трьох можливих: Рунге–Кутта, Адамса і BDF (Backward Differentiation Formula); останній буде розглянуто в розділі 10. Зокрема, для використання методу Адамса в операторі необхідно вказати параметр, що задає вибраний метод:

```
NDSolve[{eqn1, eqn2, ...}, y, {t, tmin, tmax}, Method → Adams];
```

При цьому, як і раніше, здійснюється автоматичний вибір кроку обчислень τ і порядку методу Адамса — від першого до дванадцятого.

Проілюструємо застосування стандартного оператора для розв'язання задачі, розглянутої в прикладі 9.2:

$$y' = e^{-(y^2+x^2)} + 2t, \quad x' = 2y^2 + x, \quad y(0) = 0{,}5, \quad x(0) = 1.$$

```
In[]:= sol = NDSolve[{y'[t] == Exp[-x[t]^2 + y[t]^2] + 2t, x'[t] == 2y[t]^2 + 2x[t],
            y[0] == 0, x[0] == 1.}, {x, y}, {t, 0, 0.9}, Method → Adams]
Out[]= {{x → InterpolatingFunction[{{0., 0.9}}, <>],
         y → InterpolatingFunction[{{0., 0.9}}, <>]}}
```

Визначимо відповідні функції:

```
In[]:= y[t_] = y[t]/.sol; x[t_] = x[t]/.sol
Out[]= {{InterpolatingFunction[{{0., 0.9}}, <>][t]}}
```

Можна одержати таблиці значень наближеного розв'язку системи і довести їх практичну тотожність даним табл. 9.2.

Оскільки умови стійкості багатокрокових різницевих методів Адамса більш жорсткі, ніж методів Рунге–Кутта, то може виявитися, що передбаченої в пакеті кількості кроків ($N = 1000$) не достатньо для розв'язання конкретної задачі з точністю «за замовчуванням», що відповідає 10-му знаку після коми. У таких випадках може виникнути необхідність у доповненні стандартного формату оператора NDSolve параметрами типу MaxSteps і WorkingPrecision, які вводяться так само, як і параметр Method.

Висновки

1. Один із способів побудови багатокрокових методів полягає в заміні функції, що входить до правої частини диференціального рівняння, інтерполяційними поліномами. Формули для чисельного розв'язку отримують шляхом інтегрування цих поліномів на відрізку $[t_n, t_{n+1}]$. Від кількості вузлів, які використовуються для побудови поліномів, залежить порядок і точність методу. Чим більше вузлів, тим вище порядок методу і, як наслідок, вище його точність.

2. Перевага багатокрокових методів над однокроковими полягає в тому, що для знаходження чисельного розв'язку на кожному кроці багатокрокового методу необхідно лише один раз обчислити функцію, що входить до правої частини вихідного диференціального рівняння. В однокрокових методах Рунге–Кутта цю функцію необхідно обчислювати кілька разів.

3. Одним із недоліків багатокрокових методів є те, що для знаходження розв'язку на початку інтервалу інтегрування необхідно застосовувати однокрокові методи.

4. Для знаходження розв'язку диференціального рівняння за допомогою неявних багатокрокових методів на кожному кроці необхідно розв'язувати нелінійне рівняння. Щоб запобігти цьому спільно використовують неявний і явний методи (прогноз та корекція). Спочатку явним методом обчислюється прогнозоване значення розв'язку, яке потім уточнюється за допомогою неявного. Обидва методи повинні бути одного порядку точності.

5. Області стійкості багатокрокових явних методів значно менші, ніж у методів Рунге–Кутта тих самих порядків, при цьому зі зростанням порядку методу його стійкість знижується. Для збереження стійкості розв'язку доводиться істотно зменшувати крок, що призводить до невиправданого зростання обсягу обчислень, навіть у порівнянні з методами Рунге–Кутта. Зростання обсягу обчислень під час розв'язання жорстких задач (розв'язки яких містять як швидкозмінні, так і повільні складові) обмежує застосування методів Адамса (так само, як і методів Рунге–Кутта) для моделювання широкосмугових цифрових систем збору, обробки й передачі інформації.

Контрольні запитання та завдання

1. Які з наведених різницевих схем апроксимують рівняння (9.1):

$$\frac{u_{n+1}-u_{n-2}}{3\tau}=f_n,$$

$$\frac{u_{n+1}-3u_{n-1}+2u_{n-2}}{8\tau}=\frac{f_n+f_{n-1}}{2},$$

$$\frac{3u_{n+1}-4u_n+u_{n-1}}{2\tau}=f_{n+1}.$$

Який їх порядок апроксимації?

2. Для рівняння (9.1) побудуйте різницеву схему з найвищим порядком апроксимації вигляду:

$$\frac{u_{n+1}-u_{n-1}}{2\tau}=b_0f_{n+1}+b_1f_n+b_2f_{n-1}.$$

3. Складіть програму обчислення коефіцієнтів для m-крокових різницевих методів Адамса–Башфорта і Адамса–Мултона.

4. Доведіть наведені нижче явні різницеві формули Ністрема, визначіть порядок їх точності та знайдіть вираз для головного члена локальної похибки:

$$u_{n+1} = u_{n-1} + 3\tau f_n,$$

$$u_{n+1} = u_{n-1} + \frac{1}{3}\tau(7f_n - 2f_{n-1} + f_{n-2}),$$

$$u_{n+1} = u_{n-1} + \frac{1}{3}\tau(8f_n - 5f_{n-1} + 4f_{n-2} - f_{n-3}).$$

5. Покажіть, що необхідною і достатньою умовою апроксимації рівняння (9.1) різницевими рівняннями (9.2) є виконання рівностей

$$\sum_{i=0}^{m} a_i = 0, \quad \sum_{i=1}^{m} i a_i = -1, \quad \sum_{i=0}^{m} b_i = 1.$$

6. Отримайте наведені нижче різницеві формули Мілна, визначіть порядок їх точності та знайдіть вираз головного члена локальної похибки:

явні формули (предиктор)

$$u_{n+1}^{(\text{pred})} = u_{n-3} + \frac{4}{3}\tau(2f_{n-2} - f_{n-1} + 2f_n),$$

$$u_{n+1}^{(\text{pred})} = u_{n-2} + \frac{3}{8}\tau(7f_n - 3f_{n-1} + 5f_{n-2} - f_{n-3}),$$

неявні формули (коректор)

$$u_{n+1}^{(\text{cor})} = u_{n-1} + \frac{1}{3}\tau(f_{n-1} + 4f_n + f_{n+1}),$$

$$u_{n+1}^{(\text{cor})} = u_{n-2} + \frac{3}{8}\tau(f^{(\text{pred})}{}_{n+1} + 3f_n + 3f_{n-1} + f_{n-2}).$$

7. Складіть програму обчислення наближеного розв'язку задачі Коші методом Мілна:

$$y' = t^2 - y, \quad y(0) = 1,$$
$$0 < t \leqslant 5,$$

$$u_{n+1}^{(\text{pred})} = u_{n-3} + \frac{4}{3}\tau(2f_{n-2} - f_{n-1} + 2f_n),$$

$$u_{n+1}^{(\text{cor})} = u_{n-1} + \frac{1}{3}\tau(f_{n-1} + 4f_n + f_{n+1}).$$

8. Для задачі, розглянутої в прикладі 9.2, отримайте розв'язок трикроковим методом Мілна. Побудуйте графіки наближених розв'язків.

9. Різницеве рівняння $x_{n+m} + a_1 x_{n+m-1} + ... + a_m x_m = 0$ після підстановки вектора $\mathbf{X}_j = [x_j, x_{j+1}, ..., x_{j+m-1}]^T$ зводиться до вигляду: $\mathbf{X}_{n+1} = \mathbf{A}\mathbf{X}_n$. Побудуйте відповідну матрицю **A**.

10. Методом Адамса–Башфорта третього порядку розв'яжіть із кроком $\tau = 0,1$ задачу Коші $dy/dt = y^2$ за таких початкових умов: $y_0 = 1,0000$, $y_1 = 1,1111$, $y_2 = 1,2500$. Знайдіть значення y_3 і y_4.

11. Розв'яжіть задачу Коші методом Адамса–Башфорта четвертого порядку: $dy/dt = -y/t$ у діапазоні від $t = 4$ до $t = 5$ із кроком $\tau = 0,5$ і початковими значеннями $y(2,5) = 1,2$, $y(3) = 1$, $y(3,5) = 0,857142857$, $y(4) = 0,75$. Використовуйте для оцінки похибки точні значення $y(4,5) = 0,66666667$ і $y(5) = 0,6$, знайдені аналітично.

Розділ 10

Неявні методи розв'язання жорстких задач

- Жорсткі системи диференціальних рівнянь
- Неявні однокрокові та багатокрокові методи
- Оцінювання похибок розв'язку
- Автоматичний вибір кроку і порядку методу
- Об'єднані явно-неявні методи

У цьому розділі розглядаються так звані жорсткі системи диференціальних рівнянь, що описують процеси, які містять швидкі й повільні складові. Функції розв'язку таких систем, як правило, мають дві області: з великими за модулем значеннями похідних та з малими. Це значно ускладнює обчислення розв'язку жорстких систем явними методами, такими як методи Рунге-Кутта чи Адамса, оскільки вони є умовно стійкими, що накладає певні обмеження на вибір кроку інтегрування. Для розв'язання таких систем застосовують неявні методи, у яких для обчислення розв'язку в кожній точці необхідно розв'язувати систему нелінійних рівнянь. Завдяки цьому крок можна вибирати значно більшим, ніж у разі застосування явних методів, розглянутих у розділах 8 і 9.

10.1. Поняття жорсткості системи диференціальних рівнянь

Для підвищення точності й адекватності математичних моделей складних об'єктів та процесів під час їх побудови потрібно враховувати велику кількість факторів і параметрів. Необхідність врахування в математичній моделі складових із великими і малими значеннями похідних від розв'язку неминуче спричинює так звану жорсткість рівнянь. Добре відомо, що в разі неврахування різного роду «малих величин» у математичних моделях об'єктів і процесів реальна картина явища може бути істотно спотвореною. Тому дослідники змушені включати в моделі велику кількість другорядних, на перший погляд, факторів. У результаті, як правило, підвищується порядок системи рівнянь моделі та її жорсткість. Слід відмітити, що жорсткість є властивістю математичної задачі, а не самого чисельного методу.

Розглянемо спочатку лінійну систему диференціальних рівнянь (10.1) із постійною, тобто не залежною від t, матрицею A:

$$\frac{d\mathbf{y}}{dt} = \mathbf{A}\mathbf{y}. \tag{10.1}$$

Нехай $(\lambda_1, \lambda_2, ..., \lambda_m)$ — множина власних чисел матриці A. Система диференціальних рівнянь (10.1) із постійною матрицею $m \times m$ називається *жорсткою*, якщо система асимптотично стійка за Ляпуновим, тобто

$$\mathrm{Re}(\lambda_k) < 0, \quad k = 1, 2, ..., m,$$

а відношення найбільшої за модулем дійсної частини власного числа до найменшої за модулем дійсної частини власного числа, тобто:

$$K = \frac{\max\limits_k |\mathrm{Re}(\lambda_k)|}{\min\limits_k |\mathrm{Re}(\lambda_k)|} \tag{10.2}$$

є досить великим.

Для лінійної системи, параметри якої змінюються в часі, тобто якщо $\mathbf{A} = \mathbf{A}(t)$, власні значення λ_k і, відповідно, число жорсткості K також є функціями t, і визначення жорсткої системи може бути переформульоване в такий спосіб: система

$$\frac{d\mathbf{y}}{dt} = \mathbf{A}(t)\mathbf{y}, \quad t > 0 \tag{10.3}$$

називається жорсткою в інтервалі $(0, T)$, якщо для всіх $t \in (0, T)$ виконується умова

$$\mathrm{Re}\,\lambda_k(t) < 0, \quad k = 1, 2, ..., m,$$

і число

$$K(t) = \frac{\max\limits_k |\mathrm{Re}(\lambda_k(t))|}{\min\limits_k |\mathrm{Re}(\lambda_k(t))|},$$

велике.

В основу визначення жорсткості нелінійної системи

$$\frac{d\mathbf{y}}{dt} = F(\mathbf{y}, t), \quad t > 0 \tag{10.4}$$

покладено попереднє визначення лінійної системи зі змінними коефіцієнтами типу (10.3), де роль матриці $A(t)$ відіграє матриця Якобі J для правої частини системи рівнянь (10.4).

Якщо об'єкт дослідження фізично стійкий, то

$$\mathrm{Re}(\lambda_k) < 0, \quad k = 1, 2, ..., m \tag{10.5}$$

і модулі розв'язку $e^{\lambda_k t}$ будуть згасати за умови зростання t із швидкістю, пропорційною $1/\mathrm{Re}(-\lambda_k)$. Цю величину часто називають локальною сталою часу системи. Мірою жорсткості задачі Коші є проміжок, в якому лежать її локальні сталі часу. Число жорсткості, як і число обумовленості матриці A, характеризується виразом:

$$K(\mathbf{J}) = \frac{\max\limits_k \mathrm{Re}|\lambda_k|}{\min\limits_k \mathrm{Re}|\lambda_k|}. \tag{10.6}$$

Звичайно кажуть про жорсткість задачі Коші (10.1), якщо $K(\mathbf{J}) \gg 1$. Однак у кожному конкретному випадку різні значення $K(\mathbf{J})$ можуть вважатися великими. Проте максимальний крок обчислень для явних методів обмежений нерівністю

$$\tau < \frac{c}{|\lambda_{\max}|}, \tag{10.7}$$

де c – константа, яка залежить від умов задачі, $\lambda_{\max}$ – максимальне власне число матриці Якобі.

Для використання виразів (10.7) і (10.6) необхідно знати максимальні значення модулів власних чисел λ_k або максимальні та мінімальні значення їх дійсних частин $\mathrm{Re}(\lambda_k)$. Величину λ_k можна оцінити зверху за допомогою норми матриці, а їх дійсну частину – за допомогою сліду матриці Якобі (10.2). Щоб не обчислювати обернену матрицю $\mathbf{J}^{-1}$, малі за модулем λ_k можна також оцінювати, виходячи з інтервалу інтегрування T задачі Коші (10.1), заданого з урахуванням фізичних властивостей досліджуваного об'єкта чи процесу.

Жорсткі системи диференціальних рівнянь є математичними моделями складних об'єктів, великий діапазон зміни часових характеристик яких зумовлений фізичною природою самих об'єктів чи процесів у них. Прикладами моделей таких об'єктів можуть бути моделі, які створюються формальним об'єднанням систем із різними сталими часу (наприклад, моделі інерційного об'єкта і малоінерційного контролера). Однак у загальному випадку жорсткість може проявлятися в результаті об'єднання і взаємодії нежорстких систем. У цьому разі розв'язок об'єднаної системи може істотно відрізнятися від розв'язків окремих систем.

Уперше поняття жорсткості системи диференціальних рівнянь було введено у зв'язку з труднощами, що виникали під час чисельного інтегрування цих рівнянь явними методами. Розв'язування системи з малим кроком, зумовленим співвідношенням (10.7), приводить до того, що після згасання перехідних процесів, обумовлених «швидкими» складовими розв'язку, будь-яка спроба збільшення кроку інтегрування зумовлює експоненціальне зростання похибки розв'язку через втрату його стійкості. Тому були розроблені спеціальні неявні методи чи-

сельного розв'язання жорстких рівнянь; інтерес до них і область їх застосування беззупинно зростають. Основна мета використання таких методів — забезпечити можливість зміни кроку обчислень у широких межах і забезпечити незалежність результатів від вихідних даних на різних відрізках розв'язку.

Жорстку задачу іноді називають задачею з дуже рознесеними сталими часу чи задачею з великою константою Ліпшиця, під якою розуміють норму матриці Якобі для правої частини диференціального рівняння (10.1). Систему можна вважати жорсткою, якщо коефіцієнт жорсткості (10.6) задовольняє нерівність $K(\mathbf{J}) > 10$. Однак у практичних задачах, наприклад під час моделювання процесів керування, електричних і електронних ланцюгів, процесів хімічної кінетики та ін., коефіцієнт жорсткості часто досягає значень $10^6 - 10^8$.

10.2. Неявні методи Ейлера і Рунге–Кутта

Поняття неявних методів було введене під час розгляду методів розв'язання диференціальних рівнянь (див. підрозділ 8.2). Це методи, формули яких містять шукані значення наближення u_{n+1} у лівій і правій частинах.

Наприклад, для неявного методу Ейлера справедлива формула

$$u_{n+1} = u_n + \tau f(u_{n+1}, t_{n+1}), \tag{10.8}$$

де $f(u_{n+1}, t_n)$ — значення правої частини розв'язуваного диференціального рівняння (10.1) у момент часу t_{n+1}, для якого власне і шукається наближення u_{n+1}. Формула (10.8) має наочну графічну інтерпретацію (рис. 10.1), аналогічну до тієї, яка була наведена для явного методу.

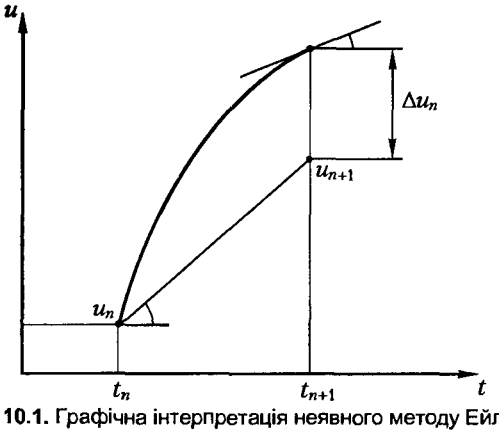


Рис. 10.1. Графічна інтерпретація неявного методу Ейлера

На рис. 10.1 показано процес знаходження чергового наближення. Для цього з початкової точки з координатами (t_n, u_n) проводиться дотична до функції точного розв'язку в точці (t_{n+1}, y_{n+1}), у результаті чого отримують наступну точку наближення (t_{n+1}, u_{n+1}). Слід відмітити, що для опуклої функції точного

розв'язку значення наближення неявним методом Ейлера будуть менше від точних значень, а значення наближення явним методом перевищуватимуть точні значення. Тому природним є бажання усереднити наближення, щоб отримати більш точний результат:

$$u_{n+1} = u_n + \frac{\tau}{2}[f(u_n, t_n) + f(u_{n+1}, t_{n+1})]. \tag{10.9}$$

Формула (10.9) відома як формула неявного методу трапецій, і вона деякою мірою відповідає формулі (8.35) явного методу Хейна, в якому точне значення похідної в кінці інтервалу обчислень замінено на наближене значення k_2.

З скороченого ряду Тейлора для $f(y_{n+1}, t_{n+1} - \tau)$ отримаємо неявні методи Рунге–Кутта:

- другого порядку:

$$u_{n+1} = u_n + \tau k_2^*, \tag{10.10}$$

де

$$k_1^* = f(t_{n+1}, u_{n+1}), \quad k_2^* = f(t_{n+1} - \tau/2, u_{n+1} - (\tau/2)k_1^*);$$

- третього порядку:

$$u_{n+1} = u_n + \frac{\tau}{4}\left(k_1^* + 3k_3^*\right), \tag{10.11}$$

де

$$k_1^* = f(x_{n+1}, t_{n+1}), \quad k_2^* = f(t_{n+1} - \tau/3, u_{n+1} - (1/3)\tau k_1^*),$$
$$k_3^* = f(t_{n+1} - (2/3)\tau, u_{n+1} - (2/3)\tau k_2^*);$$

- четвертого порядку:

$$u_{n+1} = u_n + \frac{\tau}{6}(k_1^* + 2k_2^* + 2k_3^* + k_4^*), \tag{10.12}$$

де

$$k_1^* = f(u_{n+1}, t_{n+1}), \quad k_2^* = f(t_{n+1} - \tau/2, u_{n+1} - (1/2)\tau k_1^*),$$
$$k_3^* = f(t_{n+1} - \tau/2, u_{n+1} - (1/2)\tau k_2^*), \quad k_4^* = f(t_{n+1}, u_{n+1} - \tau k_3^*).$$

Порівнюючи вирази для неявних методів (10.10) – (10.12) із формулами явних методів (8.24) – (8.30), неважко побачити їх подібність, за винятком того, що точки початку і кінця інтервалу помінялися місцями і відлік незалежного параметра t ведеться у зворотному напрямку (назад). Користуючись принципом *дуальності*, за аналогією з формулами (8.23) – (8.30) можна записати формули неявних вбудованих методів. Як і раніше, глобальна похибка неявного методу k-го порядку оцінюється величиною $O(\tau^k)$, а локальна – величиною $O(\tau^{k+1})$.

Застосування неявних методів Ейлера та Рунге–Кутта перетворює задачу інтегрування жорстких диференціальних рівнянь у задачу розв'язання нелінійних алгебраїчних рівнянь (10.8), (10.10) – (10.12) відносно u_{n+1}. У цьому разі складність нелінійного алгебраїчного рівняння зростає з ростом кількості оцінок параметрів k_i, що використовується у різних формулах.

Приклад 10.1

Розв'яжемо явним і неявним методами Ейлера рівняння $y' = f(t, y) = t^2 - 15y^2$, $t_0 = 0$, з початковими умовами $y_0 = 1$ і з кроком $\tau = 0{,}1$. Знайдемо значення $y(0{,}4)$ і порівняємо отримані результати.

Вибране значення кроку $\tau = 0{,}1$ перевищує в 1,5 разу допустимий крок обчислень явним методом Ейлера, який згідно з (10.7) дорівнює $\tau = 0{,}066(6)$, тому що $\lambda = [\partial f/\partial y]_{y_0} = -30$ і $c = 2$.

У розбіжності рішення явним методом Ейлера, якщо $\tau = 0{,}1$, можна переконатися, запрограмувавши цей метод у пакеті Mathematica:

```
In[]:= u[0] = 1;
       t[0] = 0;
       m = 4;
       τ = 0.1;
       Do [{t[n] = t[0] + τ*n;
            f[n] = t[n]^2 - 15*u[n]^2; u[n+1] = u[n] + τ*f[n]}, {n,0,m} ];
       Do [Print["n = ", n , "t[", n, "] = ", t[n], "u[", n, "] = ", u[n]], {n,0,m} ];
```

Отримані результати:

```
n = 0  t[0] = 0    u[0] =  1
n = 1  t[1] = 0.1  u[1] = -0.5
n = 2  t[2] = 0.2  u[2] = -0.874
n = 3  t[3] = 0.3  u[3] = -2.01581
n = 4  t[4] = 0.4  u[4] = -8.10207
```

Далі скористаємось формулою неявного методу Ейлера (10.8) і запишемо:

$$u_{n+1} = u_n + \tau f(t_{n+1}, u_{n+1}),$$

а потім послідовно сформуємо і розв'яжемо відповідні нелінійні алгебраїчні рівняння:

$$t_1 = \tau + t_0 = 0{,}1, \quad u_1 = y_0 + \tau(t_1^2 - 15u_1^2);$$

$$u_1 = 1 + 0{,}1\left(0{,}1^2 - 15u_1^2\right) \Rightarrow 2 = 1{,}5u_1^2 + u_1 - 1{,}001 = 0 \Rightarrow u_1 = 0{,}548962;$$

$$t_2 = \tau + t_1 = 0{,}2, \quad u_2 = u_1 + \tau(t_2^2 - u_2^2);$$

$$u_2 = 0{,}548962 + 0{,}1\left(0{,}2^2 - 15u_2^2\right) \Rightarrow 1{,}5u_2^2 + u_2 - 0{,}552962 = 0 \Rightarrow u_2 = 0{,}359308.$$

Необхідну послідовність обчислень у пакеті Mathematica можна записати в такий спосіб:

```
In[]:= f[t_, τ_, u_]:= Block[{x},
       w = FindRoot[τ*15*x^2 + x - u - τ*t^2 == 0, {x,u}];
       y = x/.w[[1]]];
       t = 0; u = 1;
       τ = 0.1; m = 4;
       Do [Print["n = ", n, "t = ", t, "u = ", u];
           t = t + τ; ul = u; u = f[t, τ, ul], {n, 0, m} ];
```

Отримані результати:

```
n = 0  t = 0    u = 1
n = 1  t = 0.1  u = 0.548962
n = 2  t = 0.2  u = 0.359308
n = 3  t = 0.3  u = 0.263868
n = 4  t = 0.4  u = 0.212276
```

10.3. Неявні лінійні багатокрокові методи

Серед лінійних багатокрокових методів, описаних у розділі 9, можна виділити кілька класів неявних методів, побудованих на сітці з постійним кроком.

10.3.1. Неявні багатокрокові методи Адамса–Мултона

Для повноти викладу тут наводяться також методи Адамса–Мултона:

- неявний метод першого порядку:

$$u_{n+2} = u_{n+1} + \frac{\tau}{2}(f_{n+2} + f_{n+1}),$$

який збігається з неявним методом трапецій (10.9);

- неявний метод другого порядку:

$$u_{n+2} = u_{n+1} + \frac{\tau}{12}(5f_{n+2} + 8f_{n+1} - f_n), \tag{10.14}$$

де f_{n+i}, $i = 0, 1, 2$ — оцінки правої частини розв'язуваного диференціального рівняння в моменти часу t_n, t_{n+1}, t_{n+2};

- неявний метод третього порядку:

$$u_{n+3} = u_{n+2} + \frac{\tau}{24}(9f_{n+3} + 19f_{n+2} - 5f_{n+1} + f_n), \tag{10.15}$$

який (за тієї ж точності) значно простіше реалізувати, ніж неявний метод Рунге–Кутта четвертого порядку (10.12), оскільки лише один член праворуч від f_{n+3} залежить від шуканого значення u_{n+3}.

10.3.2. Неявні багатокрокові методи Ракитського

На відміну від методів Рунге–Кутта методи Ракитського не вимагають багаторазового оцінювання значення похідної:

- метод другого порядку:

$$u_{n+1} = u_n + \tau f\left(t_n + \frac{\tau}{2}, \frac{u_{n+1} + u_n}{2}\right); \tag{10.16}$$

- метод четвертого порядку:

$$u_{n+1} = u_n + \frac{\tau}{6}\left[f_{n+1} + f_n + f\left(t_n + \frac{\tau}{2}, \frac{u_{n+1} + u_n}{2}\right) - \frac{\tau}{8}(f_{n+1} - f_n)\right]. \tag{10.17}$$

Приклад 10.2

Розв'яжемо неявним методом Ракитського другого порядку (10.16) рівняння: $y' = t^2 - 15y^2$, $t_0 = 0$, $y_0 = 1$, (див. приклад 10.1). Знайдемо значення $y(0,4)$ із кроком $\tau = 0,1$.

Тоді модифікована програма для пакета Mathematica набуде такого вигляду:

```
In[]:= u[0] = 1; t[0] = 0; m = 4; τ = 0.1;
       Do [t[n] = t[0] + τ*n; f[n] = (t[n + 1]+ τ/2)^2 - 15*((u[n + 1]+u[n])/2)^2;
         U[n] = u[n + 1] - u[n] - τ*f[n]; Q[n] = U[n] /. u[n + 1] → x[n];
         w = NSolve[Q[n] == 0, x[n]]; u[n + 1] = x[n] /. w[[2]];
         Print["n = ", n, "t[", n, "] = ", t[n], "u[", n, "] = ", u[n]], {n, 0, m} ];
```

Отримані результати:

```
n = 0  t[0] = 0    u[0] = 1
n = 1  t[1] = 0.1  u[1] = 0.334458
n = 2  t[2] = 0.2  u[2] = 0.223827
n = 3  t[3] = 0.3  u[3] = 0.176099
n = 4  t[4] = 0.4  u[4] = 0.155192
```

10.3.3. Неявні багатокрокові методи Гіра

На відміну від неявних методів Адамса–Мултона неявні методи Гіра (диференціювання назад) використовують множину попередніх значень функції й лише одну оцінку похідної в поточній точці t_{n+1}, базуються вони на застосуванні інтерполяційного полінома Лежандра. За постійного кроку обчислень τ отримуємо:

- метод другого порядку:

$$u_{n+2} = \frac{4}{3}u_{n+1} - \frac{1}{3}u_n + \frac{2}{3}\tau f_{n+2}; \tag{10.18}$$

- метод третього порядку:

$$u_{n+3} = \frac{18}{11}u_{n+2} - \frac{9}{11}u_{n+1} + \frac{2}{11}u_n + \frac{6}{11}\tau f_{n+3}; \tag{10.19}$$

- метод четвертого порядку:

$$u_{n+4} = \frac{48}{25}u_{n+3} - \frac{36}{25}u_{n+2} + \frac{16}{25}u_{n+1} - \frac{3}{25}u_n + \frac{12}{25}\tau f_{n+4}. \tag{10.20}$$

Однак із практичної точки зору важливими є, насамперед, неявні методи зі змінним кроком обчислень, які дозволяють реалізувати основну перевагу неявних методів, що полягає в можливості зміни кроку τ в широких межах.

Приклад 10.3

Розв'яжемо неявним методом Гіра другого порядку (10.18) рівняння: $y' = t^2 - 15y^2$, $t_0 = 0$, $y_0 = 1$, яке було розв'язане раніше, в прикладі 10.1, неявним методом Ейлера. Неявний метод Гіра називають також методом Шихмана. Потрібно, як і раніше, з кроком $\tau = 0,1$ знайти значення $y(0,4)$.

Оскільки у формулі (10.18) застосовуються значення розв'язку, отримані в двох попередніх точках, скористаємося результатами прикладу 10.1 — значеннями $u(1) = 0{,}548962$ та $u(0) = 1$. Це не дуже точне значення впливає на точність усіх наступних обчислень. Тоді модифікована програма для пакета Mathematica набуде такого вигляду:

```
In[]:= u[0] = 1; t[0] = 0; u[1] = 0.548962; m = 4; τ = 0.1;
       Do [t[n] = t[0] + τ*n; f[n] = t[n + 1]^2 - 15 u[n + 1]^2;
          U[n] = u[n + 1] -4*u[n]/3 + u[n-1]/3 - τ*2*f[n]/3;
          Q[n] = U[n] /. u[n + 1] → x[n]; w = NSolve[Q[n] == 0, x[n]];
          u[n + 1] = x[n] /. w[[2]];
        Print["n = ", n, "t[", n, "] = ", t[n], "u[", n, "] = ", u[n]], {n, 1, m} ];
```

Отримані результати:

```
n = 1  t[1]= 0.1  u[1]= 0.548962
n = 2  t[2]= 0.2  u[2]= 0.307021
n = 3  t[3]= 0.3  u[3]= 0.194531
n = 4  t[4]= 0.4  u[4]= 0.146298
```

10.4. Багатокрокові неявні методи змінного порядку і змінного кроку

Узагальнюючи формули (10.18)–(10.20), можна записати такий вираз для неявного методу «диференціювання назад», відомого як формула Гіра зі змінним кроком t та змінним порядком k:

$$u_{n+1} = \sum_{i=1}^{k} a_i u_{n+1-i} + \tau b_{-1} \varphi(u_{n+1}), \quad k \leqslant 6, \tag{10.21}$$

звідки легко знаходиться уточнена формула апроксимації похідної:

$$u'_{n+1} = \sum_{i=0}^{k} a_i u_{n+1-i}, \quad k \leqslant 6. \tag{10.22}$$

Коефіцієнти a_i у формулах (10.21) і (10.22), як буде показано нижче, обчислюють з урахуванням зміни кроку в процесі обчислень:

$$a_i = \left(\frac{t_n - t_{n-1}}{t_n - t_{n-i}}\right) \prod_{\substack{j=1, \\ j \neq i}}^{k} \left(\frac{t_n - t_{n-j}}{t_{n-1} - t_{n-j}}\right), \quad j = 1, 2, \ldots, k. \tag{10.23}$$

Оскільки вираз (10.21) відповідає нелінійному алгебраїчному рівнянню (невідома величина u_{n+1} присутня в його лівій і правій частинах одночасно), то для покращення збіжності методу Ньютона–Рафсона знаходять початкове наближення у вигляді прогнозованого значення:

$$u_{n+1}^{(0)} = \sum_{i=1}^{k+1} \gamma_i u_{n+1-i}, \tag{10.24}$$

де коефіцієнти

$$\gamma_i = \prod_{\substack{j=1 \\ j \neq i}}^{k+1} \left(\frac{t_n - t_{n-j}}{t_{n-i} - t_{n-j}} \right), \quad i = 1, ..., k+1. \tag{10.25}$$

Відома формула диференціювання назад Брайтона (BDF) відрізняється від формули Гіра (10.21) тим, що в ній замість значень функції використовують лише їх скінченні різниці:

$$\Delta u_{n-i} = u_{n+1-i} - u_{n-i}, \quad i = 1, 2, ..., k. \tag{10.26}$$

Тоді замість виразів (10.22) і (10.24) отримуємо еквівалентні співвідношення:

$$\begin{aligned} u'_{n+1} &= -\frac{1}{h} \sum_{i=0}^{k-1} \hat{a}_i \Delta u_{n-i}, \\ u^{(0)}_{n+1} &= u_n + \sum_{i=1}^{k} \hat{\gamma}_i \Delta u_{n-i}, \end{aligned} \tag{10.27}$$

в яких коефіцієнти взаємозалежні:

$$\begin{aligned} \hat{a}_i &= \sum_{j=0}^{i} a_j, \\ \hat{\gamma}_i &= -1 + \sum_{j=0}^{k+1} \gamma_j. \end{aligned} \tag{10.28}$$

Якщо від перших скінченних різниць функції перейти до різниць вищих порядків, то отримаємо співвідношення:

$$\tau u'_{n+1} = a_0 u_{n+1} - \sum_{i=0}^{k-1} a_i \nabla_n^i u_n, \tag{10.29}$$

$$u^{(0)}_{n+1} = \sum_{i=0}^{k} \rho_i \nabla_n^i u_n, \tag{10.30}$$

в яких коефіцієнти й скінченні різниці високих порядків обчислюються за такими рекурентними формулами:

$$\begin{aligned} a_i &= \rho_i \sum_{j=i}^{k-1} \frac{\tau}{t_{n+1} - t_{n-j}}, \\ \rho_i &= \rho_{i-1} \frac{t_{n+1} - t_{n+1-i}}{t_n - t_{n-i}}, \quad \rho_0 = 1, \\ \nabla_{n+1}^{i+1} u_{n+1} &= \nabla_{n+1}^{i} u_{n+1} - \rho_i \nabla_n^i u_n. \end{aligned} \tag{10.31}$$

10.5. Обчислення коефіцієнтів неявних формул наближення

Для ілюстрації методики обчислення значень коефіцієнтів використаємо формулу (10.22):

$$u'_{n+1} = -\frac{1}{\tau}\sum_{i=0}^{k} a_i u_{n+1-i}.$$

На її основі побудуємо систему лінійних рівнянь для обчислення значень a_i, по черзі застосовуючи цю формулу до послідовності поліномів:

$$y(t) = \frac{1}{\tau^m}(t_{n+1} - t)^m, \quad m = 0, 1, 2, ..., k.$$

Формула (10.22) дозволяє знайти точні значення похідних від цих поліномів

$$y'_{n+1} = y'(t_{n+1}),$$

поки їх порядок не перевищує порядок самої формули k. Для поліномів зростаючого порядку послідовно обчислюємо значення функції та її похідної:

$$\begin{aligned}
&m = 0, && y(t) = \frac{1}{\tau^0}[t_{n+1} - t]^0, && y'(t) = 0;\\
&m = 1, && y(t) = \frac{1}{\tau^1}[t_{n+1} - t]^1, && y'(t) = -\frac{1}{\tau};\\
&m = 2, && y(t) = \frac{1}{\tau^2}[t_{n+1} - t]^2, && y'(t) = -\frac{2}{\tau^2}[t_{n+1} - t];\\
&... && ... \quad ... \quad ... && ... \quad ... \quad ...\\
&m = k, && y(t) = \frac{1}{\tau^k}[t_{n+1} - t]^k, && y'(t) = -\frac{k}{\tau^k}[t_{n+1} - t]^{k-1}.
\end{aligned} \tag{10.32}$$

Підставляючи знайдені значення (10.32) в розгорнуту формулу (10.22)

$$u'_{n+1} = -\frac{1}{\tau}(a_0 u_{n+1} + a_1 u_n + a_2 u_{n-1} + ... + a_k u_{n+1-k}),$$

будуємо шукану систему лінійних рівнянь відносно невідомих коефіцієнтів:

$$\begin{aligned}
&m = 0, && 0 = -\frac{1}{\tau}(a_0 + a_1 + a_2 + ... + a_k);\\
&m = 1, && -\frac{1}{\tau} = \frac{1}{\tau}\left(a_0 0 + a_1\left[\frac{t_{n+1} - t_n}{\tau}\right]^k + ... + a_k\left[\frac{t_{n+1} - t_{n+1-k}}{\tau}\right]\right);\\
&... && ... \quad ... \quad ...\\
&m = k, && 0 = -\frac{1}{\tau}\left(a_0 0 + a_1\left[\frac{t_{n+1} - t_n}{\tau}\right]^k + ... + a_k\left[\frac{t_{n+1} - t_{n+1-k}}{\tau}\right]^k\right).
\end{aligned}$$

Застосовуючи матричну форму запису з визначником Ван-дер-Монда, остаточно отримуємо:

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ 0 & \left[\frac{t_{n+1}-t_n}{\tau}\right] & \dots & \left[\frac{t_{n+1}-t_{n+1-k}}{\tau}\right] \\ 0 & \left[\frac{t_{n+1}-t_n}{\tau}\right]^2 & \dots & \left[\frac{t_{n+1}-t_{n+1-k}}{\tau}\right]^2 \\ \dots & \dots & \dots & \dots \\ 0 & \left[\frac{t_{n+1}-t_n}{\tau}\right]^k & \dots & \left[\frac{t_{n+1}-t_{n+1-k}}{\tau}\right]^k \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \dots \\ a_k \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \dots \\ 0 \end{bmatrix}, \tag{10.33}$$

звідки

$$a_i = \left(\frac{t_{n+1}-t_n}{t_{n+1}-t_{n+1-i}}\right)\prod_{\substack{j=1 \\ j\neq i}}^{k}\left(\frac{t_{n+1}-t_{n+1-j}}{t_{n+1-i}-t_{n+1-j}}\right), \tag{10.34}$$

$$i = 1, 2, \dots, k.$$

Аналогічно для обчислення коефіцієнтів формули прогнозу побудуємо систему лінійних рівнянь (10.24):

$$\begin{bmatrix} 1 & 1 & \dots & 1 \\ 0 & \left[\frac{t_{n+1}-t_n}{\tau}\right] & \dots & \left[\frac{t_{n+1}-t_{n+1-k}}{\tau}\right] \\ \dots & \dots & \dots & \dots \\ 0 & \left[\frac{t_{n+1}-t_n}{\tau}\right]^{k+1} & \dots & \left[\frac{t_{n+1}-t_{n+1-k}}{\tau}\right]^{k+1} \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \dots \\ \gamma_{k+1} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \dots \\ 0 \end{bmatrix}, \tag{10.35}$$

з якої отримаємо значення

$$\gamma_i = \prod_{\substack{j=1 \\ j\neq i}}^{k+1}\left(\frac{t_{n+1}-t_{n+1-j}}{t_{n+1-i}-t_{n+1-j}}\right), \tag{10.36}$$

що збігаються з (10.25).

Приклад 10.4

Знайдемо коефіцієнти формули неявного багатокрокового методу Гіра, яка відповідно до виразу (10.22) має вигляд:

$$u'_{n+1} = -\frac{1}{\tau}\left(a_0 u_{n+1} + a_1 u_n + a_2 u_{n-1}\right).$$

Оскільки порядок формули дорівнює 2, то шукана система лінійних рівнянь відносно невідомих коефіцієнтів за постійного кроку формується у вигляді:

$$m = 0, \qquad 0 = -\frac{1}{\tau}(a_0 + a_1 + a_2);$$

$$m = 1, \qquad -\frac{1}{\tau} = -\frac{1}{\tau}\left(a_0 0 + a_1\left[\frac{\tau}{\tau}\right] + a_2\left[\frac{2\tau}{\tau}\right]\right);$$

$$m = 2, \qquad 0 = -\frac{1}{\tau}\left(a_0 0 + a_1\left[\frac{\tau}{\tau}\right]^2 + a_2\left[\frac{2\tau}{\tau}\right]^2\right).$$

Вона легко зводиться до такої системи:

$$a_0 + a_1 + a_2 = 0,$$
$$a_1 + 2a_2 = 1,$$
$$a_1 + 4a_2 = 0,$$

з якої отримуємо $a_2 = -1/2$, $a_1 = 2$, $a_0 = -3/2$.
Отже, для розглянутого методу за постійного кроку

$$u'_{n+1} = -\frac{1}{\tau}\left(-\frac{3}{2}u_{n+1} + 2u_n - \frac{1}{2}u_{n-1}\right) \quad \text{або} \quad u_{n+1} = \frac{4}{3}u_n - \frac{1}{3}u_{n-1} + \frac{2}{3}\tau u'_{n+1},$$

що збігається з виразом (10.18), відомим як неявний метод Гіра–Шихмана другого порядку. Розглянута методика обчислення коефіцієнтів формул наближення є універсальною і дозволяє конструювати неявні методи, які не входять у множину методів Гіра (10.22), що підтверджується наведеним нижче прикладом.

Приклад 10.5

Знайдемо значення коефіцієнтів формули наближення неявного багатокрокового методу, який має вигляд

$$u_{n+1} = a_1 u_{n-1} + b_1 \tau u'_{n+1} + b_2 \tau u'_n.$$

Порядок формули p дорівнює 2, і шукана система лінійних рівнянь для невідомих коефіцієнтів за постійногоу кроку обчислень набуває вигляду:

$$m = 0, \qquad 1 = -\frac{1}{\tau}(a_1 + b_1 0 + b_2 0);$$

$$m = 1, \qquad 0 = a_1\left[\frac{2\tau}{\tau}\right] - b_1 - b_2;$$

$$m = 2, \qquad 0 = a_1\left[\frac{4\tau^2}{\tau^2}\right] - b_1 0 - b_2\left[\frac{4\tau^2}{\tau^2}\right].$$

Із першого рівняння знаходимо коефіцієнт $a_1 = 1$. Друге і третє рівняння перетворяться в систему рівнянь:

$$2a_1 - b_1 - b_2 = 0,$$
$$4a_1 - 4b_2 = 0,$$

з розв'язку якої знаходимо $b_1 = b_2 = 1$, тому шукана формула наближення матиме такий вигляд:

$$u_{n+1} = u_{n-1} + \tau(u'_{n+1} + u'_n).$$

10.6. Стійкість неявних методів

Визначимо умови стійкості неявних методів і порівняємо їх з умовами стійкості явних методів, розглянутих у розділах 8 і 9. Почнемо з неявних методів Ейлера і трапецій, скориставшись методикою аналізу стійкості різницевих рівнянь, яка базується на визначенні коренів характеристичного рівняння (9.40).

10.6.1. Неявний метод Ейлера

Скориставшись виразом (10.9) із урахуванням тестового рівняння $y' = \lambda y$, формуємо різницеве рівняння методу $u_{n+1} = x_n + \tau\lambda u_{n+1}$, звідки $(1-\tau\lambda)u_{n+1} - u_n = 0$. Відповідне йому характеристичне рівняння набуває вигляду $(1-\tau\lambda)z - 1 = 0$, корінь якого для $\mathrm{Re}\,\lambda < 0$ і довільного значення τ завжди задовольняє умову стійкості:

$$z = \left|\frac{1}{1-\tau\lambda}\right| < 1. \qquad (10.37)$$

10.6.2. Неявний метод трапецій

На основі виразу (10.10) з урахуванням тестового рівняння $y' = \lambda y$ формуємо різницеве рівняння методу:

$$u_{n+1} = u_n + \frac{1}{2}\tau\lambda(u_n + u_{n+1}) \quad \text{чи} \quad \left(1-\frac{1}{2}\tau\lambda\right)u_{n+1} - \left(1+\frac{1}{2}\tau\lambda\right)u_n = 0.$$

Відповідне характеристичне рівняння отримуємо у вигляді:

$$\left(1-\frac{1}{2}\tau\lambda\right)z - \left(1+\frac{1}{2}\tau\lambda\right) = 0,$$

із якого, як і раніше для неявного методу Ейлера, випливає, що для $\mathrm{Re}\,\lambda < 0$ і довільного значення τ умова стійкості завжди задовольняється:

$$z = \frac{\left(1+\frac{1}{2}\tau\lambda\right)}{\left(1-\frac{1}{2}\tau\lambda\right)} < 1. \qquad (10.38)$$

10.6.3. Неявний метод Гіра–Шихмана другого порядку

Виходячи з виразу (10.18)

$$u_{n+2} = \frac{4}{3}u_{n+1} - \frac{1}{3}u_n + \frac{2}{3}\tau f_{n+2},$$

з урахуванням тестового рівняння $y' = \lambda y$ формуємо різницеве рівняння методу:

$$\left(1 - \frac{2}{3}\tau\lambda\right)u_{n+2} - \frac{4}{3}u_{n+1} + \frac{1}{3}u_n = 0,$$

характеристичне рівняння для якого:

$$\left(1 - \frac{2}{3}\tau\lambda\right)z^2 - \frac{4}{3}z + \frac{1}{3} = 0.$$

Знайдемо корені цього рівняння:

$$z_{1,2} = \frac{1}{(2 \pm \sqrt{1 + 2\tau\lambda})} < 1, \qquad (10.39)$$

тому що, коли $\tau\lambda < 0$, знаменник кореня завжди більше одиниці за значенням (для $|\tau\lambda| \leqslant 1/2$) або за модулем (для $|\tau\lambda| > 1/2$).

Запропоновано називати *A-стійкими* розглянуті тут та інші неявні методи, в яких область стійкості у разі розв'язування тестового модельного рівняння $y' = \lambda y$ містить всю ліву комплексну півплощину $\tau\lambda$, тобто, якщо $\mathrm{Re}\,\lambda < 0$, розв'язок буде асимптотично стійким за будь-якого додатного τ. Показано також, що ніякий явний лінійний багатокроковий метод не може бути A-стійким і не існує A-стійкого неявного лінійного багатокрокового методу з порядком $k > 2$.

Дослідження стійкості методу Гіра–Шихмана третього порядку зручніше проводити графічно побудовою годографа стійкості аналогічно визначенню областей стійкості багатокрокових методів (див. рис. 9.2).

Записуючи характеристичне рівняння (10.19) у вигляді $(11 - 6\tau\lambda) - 18z^{-1} + 9z^{-2} - 2z^{-3} = 0$, знаходимо з нього $\mu = \tau\lambda = (11 - 18z^{-1} + 9z^{-2} - 2z^{-3})/6$. Підставивши $z = q = e^{i\varphi}$, отримаємо рівняння межі області стійкості.

Годограф стійкості будується за допомогою пакета Mathematica:

```
In[]:= μ[φ_]:= (11 - 18*E^(I*φ) + 9*E^(2I*φ) - 2*E^(3I*φ))/6; μx[φ_]:= Re[μ[φ]]; μy[φ_]:= Im[μ[φ]];
       Tx4 = Graphics[{ Text[|q| < 1, {0.5, 0.7}], Text[|q| > 1, {3, 0.8}]}];
       rg4 = ParametricPlot [{μx[φ], μy[φ]}, {φ, 0, 2*Pi},
       AxesLabel → {Re τλ, Im τλ}, DisplayFunction → Identity];
       Show[rg4, Tx4, DisplayFunction → $DisplayFunction];
```

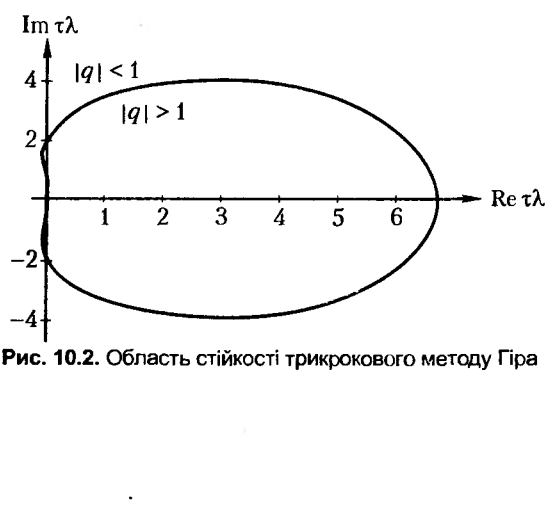


Рис. 10.2. Область стійкості трикрокового методу Гіра

На відміну від попереднього прикладу, існують точки межі області стійкості, розташовані в лівій півплощині. Тому метод третього порядку не є A-стійким.

Для неявних методів з порядком $k > 2$ вимоги A-стійкості послаблюються. Вводиться поняття жорстко-стійких методів (рис. 10.3), для яких їх область стійкості у разі розв'язання тестового рівняння містить не всю ліву півплощину $\tau\lambda$, а тільки області R_1 і R_2, де R_1 визначається умовою

$$-a \leqslant \mathrm{Re}(\tau\lambda) \leqslant b, \quad \|\mathrm{Im}(\tau\lambda)\| \leqslant d, \tag{10.40}$$

а R_2 — умовою

$$\mathrm{Re}(\tau\lambda) < -a, \quad a > 0, \quad d > 0, \quad b \geqslant 0. \tag{10.41}$$

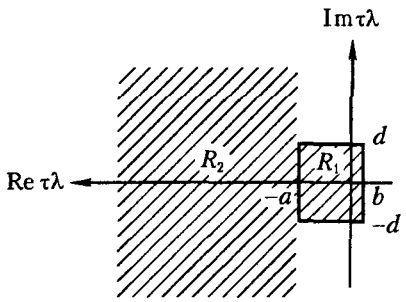


Рис. 10.3. Область стійкості жорстко-стійких методів

Крім того, для $\tau\lambda \in R_1$, забезпечується задана точність. Суть вимоги $\tau\lambda \in \in R_1 \cup R_2$ полягає в тому, що за великих $\tau|\lambda|$ забезпечується згасання «швидких» складових розв'язку, а за малих $\tau|\lambda|$, якщо $\tau\lambda \in R_1$, зберігається необхідна точність «повільних» складових розв'язку.

До жорстко-стійких методів належать методи Гіра і Брайтона, неявні методи Ракитського і неявні методи Рунге–Кутта, коли $k > 2$.

Наведемо умови стійкості неявних методів Рунге–Кутта:

- другого порядку:

$$\left|1 - \tau\lambda + \frac{\tau^2\lambda^2}{2}\right| > 1; \tag{10.42}$$

- третього порядку:

$$\left|1 - \tau\lambda + \frac{\tau^2\lambda^2}{2} - \frac{\tau^3\lambda^3}{6}\right| > 1; \tag{10.43}$$

- четвертого порядку:

$$\left|1 - \tau\lambda + \frac{\tau^2\lambda^2}{2} - \frac{\tau^3\lambda^3}{6} + \frac{\tau^4\lambda^4}{24}\right| > 1. \tag{10.44}$$

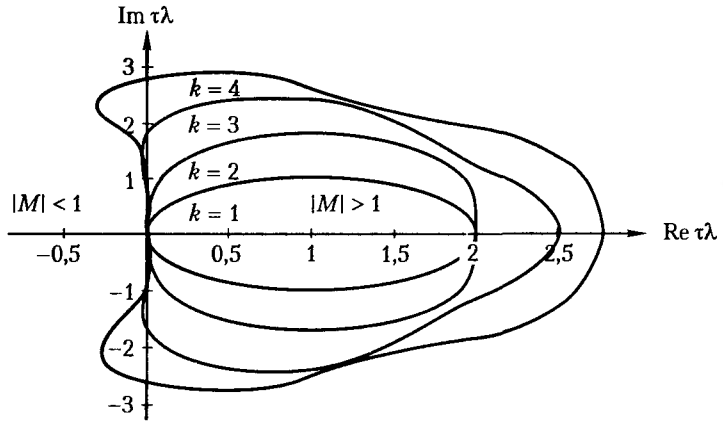


Рис. 10.4. Області стійкості неявних методів Рунге–Кутта

Порівнюючи умови (10.42) – (10.44) з умовами стійкості явних методів тих же порядків (8.59), можна зробити висновок, що області стійкості неявних методів Рунге–Кутта можуть бути побудовані з областей стійкості явних методів (рис. 8.7) заміною $\tau\lambda$ на $-\tau\lambda$ і зміною знака нерівності. Однак при цьому змінюється і правило оцінки стійкості розв'язків: тепер точки всередині годографа характеризують нестійкі розв'язки, а поза годографом — стійкі. Області стійкості багатокрокових неявних методів змінного порядку Гіра, отримані на підставі виразу (10.21), наведені на рис. 10.5.

Тут, як і на рис. 10.4, точки всередині годографа характеризують нестійкі розв'язки, а поза годографом — стійкі. На рисунку видно, що методи першого і другого порядків ($k \leqslant 2$) є A-стійкими, тобто області їх абсолютної стійкості включають всю ліву півплощину $\mathrm{Re}(\tau\lambda) \leqslant 0$, де λ — власні значення матриці Якобі для правої частини розв'язуваного рівняння. Для $k = 3, \ldots, 6$ вони з одного боку $A(\alpha)$-стійкі, тому що області їх абсолютної стійкості включають необмежений клин $\arg(-\tau\lambda) < \alpha$, а з іншого — жорстко стійкі, оскільки області їх стійкості обмежені частиною лівої півплощини, при цьому в околі початку координат забезпечується не лише стійкість, але й точність.

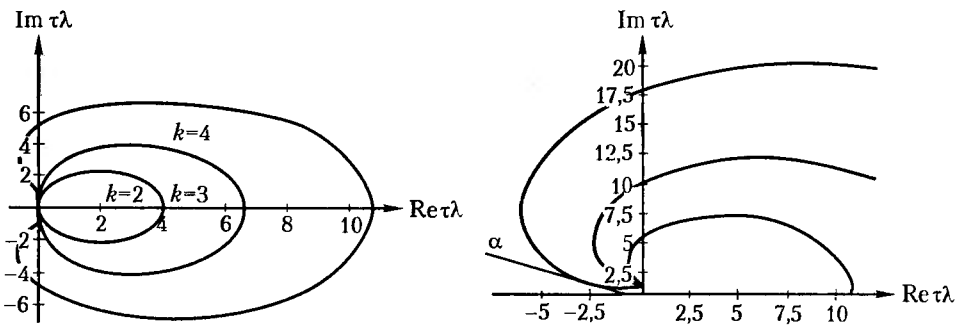


Рис. 10.5. Області стійкості k-крокових методів Гіра:
а — годографи методів першого — четвертого порядків;
б — годографи методів четвертого — шостого порядків

Жорстка стійкість характеризується максимальним кутом дотичної α до кривої годографа в лівій півплощині (для $k \geqslant 2$). У табл. 10.1 наведені значення кута для формул Гіра різних порядків $k \leqslant 6$ за відносної зміни кроку обчислень $q = \tau_j / \tau_{j-1}$.

Таблиця 10.1. Значення кута нахилу дотичної α для годографів методів Гіра різних порядків

Зміна кроку q	Порядок методу k						
	1	2	3	4	5	6	7
1	90°	90°	86°	74°	52°	18°	0°
1,01	90°	90°	85°	72°	48°	10°	0°
1,02	90°	90°	85°	71°	45°	1°	0°
1,05	90°	90°	83°	65°	32°	0°	0°
1,1	90°	89°	81°	55°	1°	0°	0°
1,2	90°	87°	70°	25°	0°	0°	0°
1,5	90°	75°	25°	0°	0°	0°	0°
2,0	90°	50°	0°	0°	0°	0°	0°

Виходячи з даних табл. 10.1, можна зробити висновок, що в процесі обчислень часовий крок потрібно змінювати *поступово* (а не збільшувати або зменшувати вдвічі, як це прийнято), якщо необхідно використовувати неявні методи високих порядків. Підвищити порядок $A(\alpha)$-стійких методів можна, якщо у формулу наближення ввести другу похідну, як це зроблено в методах Енрайта:

$$u_{n+1} = \sum_{i=0}^{k} a_i u_{n-i} + \tau \sum_{r=-1}^{k+1} \beta_i u'_{n-r} + \gamma \tau^2 u''_{n+1}, \qquad (10.45)$$

при цьому жорстка стійкість зберігається аж до методу з порядком $k = 12$ (рис. 10.6). Друга похідна функції у формулі (10.45) обчислюється за допомогою матриці Якобі для правої частини рівняння так:

$$u''_{n+1} = \varphi'(u_{n+1}, t_{n+1}) = \left(\frac{\partial \varphi}{\partial y} \frac{\partial y}{\partial t} \right)_{t=t_{n+1}} = \mathbf{J}\, \varphi(u_{n+1}, t_{n+1}).$$

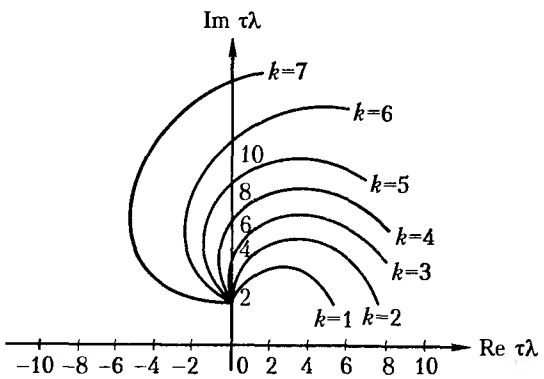


Рис. 10.6. Області стійкості методів Енрайта, які використовують другу похідну

10.7. Оцінка локальної похибки і організація обчислень

Як було відмічено в розділі 9, головний член локальної похибки наближення багатокрокового лінійного методу k-го порядку визначається першим неврахованим членом відповідного еквівалентного ряду Тейлора:

$$\eta_{n+1} = [C_{k+1} y^{(k+1)}(t_{n+1}) \tau^{k+1}].$$

Цей вираз, що містить похідні високих порядків від розв'язку, можна безпосередньо використовувати у випадках, коли застосовуються формули методів Гіра змінного порядку і змінного кроку, (10.29) і (10.30), виражені через скінченні різниці високих порядків, при цьому:

$$\eta_{n+1} \approx \frac{\tau}{t_{n+1} - t_{n-k}} \nabla^{k+1}_{n+1} u_{n+1} = \frac{1}{t_{n+1} - t_{n-k}} \left(u_{n+1} - \sum_{i=1}^{k} \rho_i \nabla^i_n u_n \right). \qquad (10.46)$$

У таких випадках принципово можливе одночасне паралельне оцінювання похибки обчислень для кожного із стійких методів із порядками $k = 1, 2, ..., 6$, що полегшує вибір найбільш ефективного з них на кожному кроці обчислень.

Якщо ж застосовуються формули методів Гіра (10.22) або (10.27), в яких використовуються значення функції або її перші скінченні різниці, то доречніше оцінювати похибку вбудованими методами (див. підрозділ 8.4). При цьому більш зручним, з одного боку, є використання формул прогнозу (10.24), (10.27) і (10.30), а з іншого, — формул наближень (10.22), (10.27) і (10.30). У результаті похибка виражається через різницю прогнозованого і обчисленого значень:

$$\eta_{n+1} = \tau \frac{u^{(0)}_{n+1} - u^{(k)}_{n+1}}{t_{n+1} - t_{n-k}}. \qquad (10.47)$$

Оцінка похибки з використанням розв'язків із різними кроками, яка описана в розділі 8, для методу змінного кроку принципово неможлива.

Нагадаємо, що прогнозоване значення $u^{(0)}_{n+1}$ використовується також у разі застосування методу Ньютона для обчислення наближення $u^{(k)}_{n+1}$ під час розв'язання нелінійних алгебраїчних рівнянь (10.22), (10.27) і (10.30). З метою економії часу часто застосовують модифіковану процедуру Ньютона, для якої обчислення матриці Якобі та розв'язання отриманої лінійної системи методом LU-розкладання виконується лише один раз на кожному кроці. Це, звичайно, призводить до збільшення значення локальної похибки і, як наслідок, до зростання кількості виконуваних кроків на заданому інтервалі, однак загальний час розрахунку може виявитися меншим, тому що істотно зменшуються витрати на кожен із таких кроків.

Звичайно, така організація обчислень можлива за умов, які практично збігаються з загальними умовами розв'язання диференціального рівняння (див. під-

розділ 8.1) і полягають в тому, що права частина диференціального рівняння повинна бути принаймні двічі диференційована і мати обмежені значення. До того ж для всіх t функція

$$\phi(t) = \left[\frac{\partial f}{\partial t_{n+1}}\right]^{-1} f(t) \tag{10.48}$$

повинна бути монотонною.

У комерційних програмах, які реалізують неявні методи змінного порядку і кроку Гіра, звичайно передбачаються засоби для розв'язування конкретної задачі з заданою точністю шляхом вибору деяких констант (опцій), які визначають кількість допустимих ітерацій Ньютона (чи оцінок матриці Якобі) на кожному кроці (від однієї до десятків) і кількості кроків, на яких матриця Якобі не змінюється (від одного до кількох).

10.8. Автоматичний вибір кроку і порядку методу

Суть методу змінного порядку і змінного кроку полягає в тому, що ці величини, залежно від виду функції розв'язку, *погоджено вибираються автоматично* на кожному кроці з метою мінімізації загальних витрат комп'ютерних ресурсів на розв'язання задачі з заданою користувачем точністю.

Оскільки порядок методу безпосередньо залежить від кількості визначених і збережених величин, то змінювати його не складно, прогнозуючи локальні похибки для методів різних порядків і вибираючи той, для якого оцінка похибки мінімальна. Потім для обраного порядку методу можна оцінити максимальний крок, виходячи з умов забезпечення заданої точності розв'язку і дотримання умов його стійкості. Є принаймні дві можливі реалізації описаного вище підходу.

10.8.1. Алгоритм із довільним вибором порядку методу

Застосовується у випадках, коли використовуються формули методів Гіра змінного порядку і змінного кроку (10.29) і (10.30), виражені через скінченні різниці високих порядків. Цей алгоритм передбачає виконання таких кроків.

1. На кожному кроці за формулами (10.46) прогнозується похибка розв'язку для методів $1, 2, k+1$ порядків:

$$E_D^1, E_D^2, ..., E_D^{k+1}, \quad 1 \leqslant k \leqslant 6. \tag{10.49}$$

2. Вибирається найменша з можливих похибок:

$$E_i = \min\left(E_D^i\right), \quad i = 1, ..., k+1, \tag{10.50}$$

яка визначає вибір порядку методу.

3. Встановлюється нове значення часового кроку на основі заданої користувачем похибки розв'язку ε (8.56) з урахуванням обраного прогнозованого мінімального значення локальної похибки E_i (10.50) і залежності похибки розв'язку від порядку методу k (8.9):

$$q = \frac{\tau_{j+1}}{\tau_j} = \left[\frac{\varepsilon\tau_j}{E_i T}\right]^{1/k}, \tag{10.51}$$

де ε — задана користувачем відносна похибка обчислень, T — часовий інтервал розв'язку, τ_j — попередній крок.

10.8.2. Алгоритм з обмеженим вибором порядку методу

Застосовується у випадках, коли використовуються формули методів Гіра змінного порядку і змінного кроку (10.22) і (10.27), а похибка визначається через різницю прогнозованого і обчисленого значень (10.47). Цей алгоритм передбачає виконання таких кроків.

1. Прогнозуються похибки для трьох методів із різними порядками точності:

$$E_D^{k-1}, E_D^k, E_D^{k+1}.$$

Після k кроків порядок методу змінюється відповідно до значення, що забезпечує мінімальну похибку E_D^i.

2. Обчислюється новий крок за формулою (10.51).

Більшу функціональну гнучкість першого алгоритму порівняно з другим забезпечує зменшення кількості кроків на заданому інтервалі розв'язку і можливість досить стійкого керування обчисленнями.

Нагадаємо, що коефіцієнти формул наближення методів змінного порядку і змінного кроку обчислюються на кожному кроці за формулами (10.23), (10.25), (10.28) і (10.31), що збільшує обсяг обчислень. Тому була запропонована спрощена схема організації обчислень зі змінним кроком, коли використовуються формули наближення з постійними коефіцієнтами, які перераховуються з використанням інтерполяційних поліномів, побудованих на основі раніше обчислених значень. Це досягається збереженням попередніх значень у вигляді вектора скінченних різниць вищих порядків

$$\mathbf{Z}_n = \left[y_n, y_{n-1}, ..., y_{n-k}, \nabla\tau y_n, \nabla^2(\tau y_n), \nabla^2(\tau y_n), ..., \nabla^k(\tau y_n)\right] \tag{10.52}$$

або вектора Нордсика

$$\mathbf{N}_n = \left[y_n, \tau y Y_n, \frac{\tau^2 yY}{2}, ..., \frac{\tau^{2k+1} y_n^{(2k+1)}}{(2k+1)!}\right]. \tag{10.53}$$

У першому випадку старі й нові значення пов'язані через коефіцієнт q, обумовлений виразом (10.51):

$$\begin{bmatrix} \tilde{y}'_n \\ \tau\tilde{y}'_n \\ \nabla^1(\tau\tilde{y}'_n) \\ \nabla^2(\tau\tilde{y}'_n) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & q & 0 & 0 \\ 0 & 0 & q^2 & q^2(1-q) \\ 0 & 0 & 0 & q^3 \end{bmatrix} \begin{bmatrix} Y'_n \\ \tau Y'_n \\ \nabla^1(\tau Y'_n) \\ \nabla^2(\tau Y'_n) \end{bmatrix}$$

або в матрично-векторній формі

$$\tilde{\mathbf{Z}}_n = \mathbf{T}(q)\mathbf{Z}_n. \qquad (10.54)$$

У другому випадку для вектора Нордсика обчислення збережених значень для нового значення кроку спрощується:

$$\tilde{\mathbf{N}}_n = \mathbf{Q}(q)\mathbf{N}_n, \qquad (10.55)$$

тому що матриця обчислення значень $\mathbf{Q}(q)$ на відміну від матриці $\mathbf{T}(q)$, є чисто діагональною $\mathbf{Q}(q) = \mathrm{diag}(q^i)$, ненульові елементи в ній визначаються коефіцієнтом q в i-му степені.

10.9. Об'єднані явно-неявні процедури

Як відзначалося вище, неявні методи ефективно використовуються для моделювання динамічних процесів у широкосмугових інформаційних системах, оскільки автоматичний вибір їх порядку і кроку визначається характером розв'язку. На рис. 10.7 показана типова перехідна характеристика системи, на якій можна чітко виділити дві ділянки: швидко наростаючий передній фронт, на якому обчислення виконуються з малим кроком, і згасаючий процес, де крок обчислень істотно збільшується через повільну зміну значень змінної.

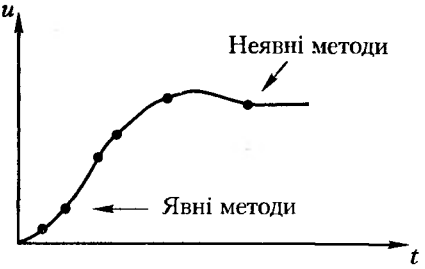


Рис. 10.7. Типова перехідна характеристика з виділеними ділянками використання явних і неявних методів

Тому доцільно побудувати такі об'єднані явно-неявні процедури, у яких на ділянках швидкої зміни розв'язку обчислення виконувалися б за допомогою явних методів, а на ділянках повільної зміни розв'язку (малоінформативні) — за допомогою неявних методів, які дозволяють довільно змінювати (і збільшувати)

крок. У таких процедурах необхідно автоматично розпізнавати ділянки розв'язку і автоматично вибирати відповідний метод розв'язування.

Одним із можливих шляхів розпізнавання локальної «жорсткості» розв'язку в околі точки $t_n \in [t_0, T]$ є перевірка виконання умови

$$P_n = L_n(T - t_n) \geqslant B \geqslant 1, \tag{10.56}$$

де

$$R(\mathbf{J}_n) \leqslant L_n \leqslant \|\mathbf{J}_n\|.$$

Тут $R(\mathbf{J}_n)$ – спектральний радіус матриці Якобі $\mathbf{J}_n$, зумовлений її найбільшим власним значенням λ_{max}, яке знаходиться степеневим методом (див. розділ 4).

Умова переходу від неявного методу до явного визначається співвідношенням

$$\tau_{\text{неявн}} < \tau_{\text{явн}} = \frac{c}{\lambda_{max}}. \tag{10.57}$$

Умова зворотного переходу – від явного методу до неявного – набуває вигляду

$$\tau_{\text{неявн}} > \gamma\tau_{\text{явн}}, \tag{10.58}$$

де γ – співвідношення обчислювальних витрат на виконання кроку неявним і явним методами ($\gamma = 2$).

Застосування такого підходу дозволяє прискорити обчислення на 20–30 % у разі збереження похибки розв'язку [22].

10.10. Явні нелінійні методи з А-стійкістю

Останнім часом ведуться інтенсивні дослідження з метою побудови явних методів розв'язання жорстких задач із підвищеною стійкістю, стійкість яких наближалася б до стійкості неявних методів. Оскільки, як було показано, для лінійних методів таких можливостей не існує, пошук ведеться серед нелінійних. Певні успіхи можна продемонструвати на прикладах явних нелінійних методів першого і другого порядків.

10.10.1. Явний нелінійний метод першого порядку

Відмовимося від експонентної апроксимації розв'язку диференціального рівняння, обумовленого власними значеннями матриці Якобі від його правої частини (10.4), і перейдемо до часової апроксимації на інтервалі $[t_n, t_{n-1}]$

$$y_k(t) = \frac{A}{1 + \sum_{i=1}^{k} a_i t^i}. \tag{10.59}$$

Тоді, коли $k = 1$, маємо гіперболічну апроксимуючу функцію:

$$y_1(t) = \frac{A}{1 + a_1 t},$$

що відповідає нелінійній формулі наближення, запропонованій Федулою:

$$u_{n+1} = \frac{u_n^2}{u_n - \tau f_n}. \tag{10.60}$$

Покажемо, користуючись тестовим рівнянням $y' = \lambda y$, що, коли $\operatorname{Re}\lambda < 0$, формула (10.60) має A-стійкість, яка збігається зі стійкістю неявного методу Ейлера. Дійсно, на підставі співвідношення (10.60) можна записати:

$$u_{n+1} = \left(\frac{u_n}{u_n - \tau f_n}\right) u_n = \left(\frac{1}{1 - \tau\lambda}\right) u_n = K(\tau\lambda) u_n. \tag{10.61}$$

Із виразу (10.61) випливає, що, коли $\operatorname{Re}\lambda < 0$, за будь-якого довільного кроку τ

$$K(\tau\lambda) = \frac{1}{1 - \tau\lambda} < 1. \tag{10.62}$$

Приклад 10.6

Розв'яжемо явним нелінійним методом Федули рівняння $y' = t^2 - 15y^2$, $t_0 = 0$, $y_0 = 1$, розв'язане раніше, в прикладі 10.1, неявним методом Ейлера. Потрібно, як і раніше, з кроком $\tau = 0{,}1$ знайти значення $y(0{,}4)$.

Користуючись формулою (10.60), запишемо програму розв'язання задачі засобами пакета Mathematica:

```
In[]:= u[0] = 1;
       t[0] = 0;
       m = 4;
       τ = 0.1;
       Do [t[n] = t[0] + τ*n; f[n] = t[n]^2 - 15*u[n]^2;
           u[n+1] = N[(u[n])^2 /( u[n] - τ*f[n])], {n, 0, m} ];
       Do [Print["n = ", n, "t[", n, "] = ", t[n], "u[", n, "] = ", u[n]], {n, 0, m} ];
```

Отримані результати дещо відрізняються від результатів прикладу 10.1:

```
n = 0  t[0] = 0    u[0]= 1
n = 1  t[1] = 0.1  u[1]= 0.4
n = 2  t[2] = 0.2  u[2]= 0.250391
n = 3  t[3] = 0.3  u[3]= 0.184164
n = 4  t[4] = 0.4  u[4]= 0.150047
```

Цікаво порівняти результати розв'язання одного й того ж самого рівняння, отримані в прикладах 10.1, 10.2, 10.3 і 10.6 різними неявними методами (табл. 10.2). Тут же наведено розв'язання неявним методом BDF п'ятого порядку як найбільш точне.

Таблиця 10.2. Порівняльні результати обчислення

Метод	Значення $y(0,4)$
Явний метод Ейлера першого порядку	–8,10207
Неявний метод Ейлера першого порядку	0,212276
Неявний метод Ракитського другого порядку	0,155192
Неявний метод Гіра–Шихмана другого порядку зі значенням $y[0,1]$, отриманим неявним методом Ейлера	0,146298
Явний нелінійний метод Федули першого порядку	0,150047
Неявний метод BDF п'ятого порядку	0,156484

Як свідчать дані табл. 10.2, явний нелінійний метод Федули більш точний, ніж неявний метод Ейлера. У разі розв'язування нелінійного рівняння його точність наближається до точності неявних методів другого порядку.

10.10.2. Явний нелінійний метод другого порядку

Існує явна формула Глинського другого порядку для розв'язання жорсткого диференціального рівняння:

$$u_{n+1} = \begin{cases} \dfrac{u_n}{1 - \dfrac{\tau}{2u_n}(3f_n - f_{n-1}) + \left[\dfrac{\tau}{2u_n}(3f_n - f_{n-1})\right]^2}, & \text{якщо} \quad u_n \neq 0; \\ \dfrac{\tau}{2}(3f_n - f_{n-1}), & \text{якщо} \quad u_n = 0. \end{cases} \tag{10.63}$$

Стійкість цієї формули можна довести, розглядаючи дві сусідні ітерації, на яких для модельного рівняння $y' = \lambda y$ має виконуватися така умова:

$$K_{n+1}(\tau\lambda)\,K_n(\tau\lambda) < 1. \tag{10.64}$$

Використовуючи модельне рівняння, перетворимо вираз (10.63):

$$K_{n+1}(\tau\lambda) = \frac{1}{1 - \dfrac{\tau\lambda}{2}\left(3 - \dfrac{1}{K_n(\tau\lambda)}\right) + \left(\dfrac{\tau\lambda}{2}\left(3 - \dfrac{1}{K_n(\tau\lambda)}\right)\right)^2}. \tag{10.65}$$

Проаналізувавши знаменник формули (10.65), переконаємося:

- якщо $1/3 < K_n(\tau\lambda) < 1$, то $K_{n+1}(\tau\lambda) < 1$ і умова (10.64) виконується;
- якщо $K_n(\tau\lambda) = 1/3$, то $K_{n+1}(\tau\lambda) = 1$ і умова (10.64) все ж виконується;
- якщо $0 < K_n(\tau\lambda) < 1/3$, то, коли $\tau\lambda/2(3 - 1/K_n(\tau\lambda)) = 1/2$, досягається мінімальне значення знаменника виразу (10.65), рівне $3/4$, у результаті чого $K_{n+1}^{\max} = 4/3$ і умова (10.64), як і раніше, виконується.

Приклад 10.7

Розв'яжіть систему рівнянь

$$\begin{cases} x' = -0,1x - 199,9y, \\ y' = -200y - 0,1x, \\ x(0) = 1, \quad y(0) = -1, \quad 0 \leqslant t \leqslant 0,5 \end{cases}$$

методом Глинського в разі обчислень із кроком $\tau = 0,05$.

Неважко бачити, що власні значення матриці Якобі відповідно $\lambda_1 = -0,1$, $\lambda_2 = -200$. Тому за обраного кроку явні методи не забезпечують збіжності розв'язку цієї системи, оскільки згідно з виразом (10.2) коефіцієнт жорсткості $K = 2000$.

Пакет Mathematica дозволяє знайти точний розв'язок системи диференціальних рівнянь за допомогою оператора DSolve:

```
In[]:= Eq1 = x'[t] == -0.1*x[t] - 1.999*y[t];
       Eq2 = y'[t] == -200*y[t] - 0.1*x[t];
       sol = DSolve[{Eq1, Eq2, x[0] == 1, y[0] == -1}, {x[t], y[t]}, t]
```

Отримаємо розв'язок у вигляді:

```
Out[] ={{x[t] → -0.0099949 Exp[-200.001 t] + 1.00999 Exp[-0.099 t],
         y[t] → -0.999495 Exp[-200.001 t] - 0.000505248 Exp[-0.099 t]}}
```

і обчислимо значення змінних для двох моментів часу (табл. 10.3):

Таблиця 10.3. *Результати оцінки точних значень змінних*

Момент часу	$x(t)$	$y(t)$
$t = 0,05$	1,005	–0,000548128
$t = 0,5$	0,961213	–0,000480847

Враховуючи початкове значення $y(0) = -1$, на основі даних таблиці робимо висновок, що вже протягом першого кроку $t = 0,05$ «швидка» змінна $y(t)$ зменшується майже до нуля, в той час як «повільна» змінна $x(t)$ майже не змінюється.

Програмна реалізація методу Глинського в пакеті Mathematica набуває такого вигляду:

```
In[]:= f[u1_,u2_][1]:= -0.1*u1 - 1.999*u2;    (*праві частини рівнянь*)
       f[u1_,u2_][2]:= -200*u2 - 0.1*u1;
       t0 = 0; tm = 0.5; u10 = 1; u20 = -1;   (*початкові умови*)
       u11 = 1.005; u21 = -0.000548128;       (*додаткові умови з табл. 10.3*)
       τ = 0.05;
       m = IntegerPart[(tm - t0)/τ];
       t[0] = t0;
       U[1, 0] = u10; U[1, 1] = u11;
       U[2, 0] = u20; U[2, 1] = u21;
       Do [t[n] = t[0] + n*τ;
         Do [If[U[j, n] = 0,
             U[j, n + 1] = (τ/2)*(3*f[U[1, n],
             U[2, n]][j] - f[U[1, n - 1],
             U[2, n - 1]][j]),
             U[j, n + 1] = U[j, n]/(1 - (τ/(2*U[j, n]))*(3*f[U[1, n],
             U[2, n]][j] - f[U[1, n - 1],
             U[2, n - 1]][ j]) + ((τ/(2*U[j, n]))*(3*f[U[1, n], U[2, n]][j] -
             f[U[1, n - 1], U[2, n - 1]][j]))^2)];
```

```
Print["n = ", n, "t[", n, "] = ", t[n], "U[", j, ",", n, "] = ",
U[j, n]], {j, 1, 2}], {n, 1, m}]
```

Отримані результати зведені в табл. 10.4.

Таблиця 10.4. Оцінки значень змінних

Момент часу	$x(t)$	$y(t)$
$t = 0{,}05$	1,005	–0,000548128
$t = 0{,}1$	0,950225	$-6{,}5964 \times 10^{-12}$
$t = 0{,}15$	0,945583	0
$t = 0{,}2$	0,940867	0
$t = 0{,}25$	0,936175	0
$t = 0{,}3$	0,931506	0
$t = 0{,}35$	0,92686	0
$t = 0{,}4$	0,922237	0
$t = 0{,}45$	0,917638	0
$t = 0{,}5$	0,913061	0

Результати в таблиці свідчать про те, що коли явні лінійній методи зовсім непрацездатні, явний нелінійний метод Глинського працює бездоганно і забезпечує збіжність розв'язку, хоча й з невеликою точністю (кілька відсотків) через високе значення вибраного кроку.

На жаль, спроби побудувати явні нелінійні методи більш високого порядку не дали вражаючих результатів, оскільки необхідний для них обсяг обчислень перевищує той, що потрібний для реалізації звичайних неявних методів, розглянутих у цьому розділі.

10.11. Засоби розв'язання жорстких задач у пакеті Mathematica

У пакеті Mathematica існує можливість настроїти стандартну операцію NDSolve на використання неявних методів BDF Брайтона (10.27) і (10.28) з автоматично змінюваними порядком методу — від першого до п'ятого — і кроком обчислення. Наприклад, розв'язання задачі з прикладу 10.4 $y' = t^2 - 15y^2$, $t_0 = 0$, $y_0 = 1$ в інтервалі $[0, 0{,}5]$ і знаходження значення $y(0{,}4)$ виконується у такий спосіб:

```
In[]:= Y1 = NDSolve[{y'[t] == -15*y[t]^2 + t^2, y[0] == 1}, y, {t, 0, 0.5},
       Method → BDF]
Out[]= {{y → InterpolatingFunction[{{0., 0.5}}, <>]}}

In[]:= y[t_] = y[t]/.Y1;
       y[0.4]
Out[]= {0.156484}
```

Це значення $y(0{,}4) = 0{,}156484$ було використане раніше в табл. 10.2.

Знайдемо також розв'язок жорсткої задачі з прикладу 10.5:

$$\begin{cases} x' = -0,1x - 1,999y, \\ y' = -200y - 0,1x, \\ x(0) = 1, \quad y(0) = -1, \quad 0 \leqslant t \leqslant 0,5 \end{cases}$$

методом BDF і знайдемо значення $x(0,5)$ і $y(0,5)$.
Для цього запишемо і виконаємо таку послідовність операцій:

```
In[]:= sol = NDSolve[{y'[t] == -200*y[t] - 0.1*x[t],
           x'[t] == -0.1*x[t] - 1.999*y[t],
           y[0] == -1, x[0] == 1.}, {x, y}, {t, 0, 0.6},
       Method → BDF]
Out[]= {{x → InterpolatingFunction[{{0., 0.6}}, <>],
         y → InterpolatingFunction[{{0., 0.6}}, <>]}}

In[]:= x[t_]=x[t]/.sol;
       y[t_]=y[t]/.sol
Out[]= {{InterpolatingFunction[{{0., 0.6}}, <>] [t],
```

Знайдемо значення $x(0,5)$ і $y(0,5)$.

```
In[]:= x[0.5]
Out[]= {0.961217}

In[]:= y[0.5]
Out[]= {-0.000480847}
```

Отримані значення $x(0,5) = 0,961217$ і $y(0,5) = -0,000480847$ з точністю 10^{-6} збігаються з точними значеннями $x^*(0,5) = 0,961213$ і $y^*(0,5) = -0,000480847$, наведеними в табл. 10.3.

Розробники пакета Mathematica передбачили в разі використання стандартного оператора NDSolve можливість автоматичного переходу від явних методів Адамса (від першого до дванадцятого порядку) до неявних методів BDF (від першого до п'ятого порядку) і навпаки. Це досягається за допомогою оператора NDSolve без вказівки на необхідний метод чи введенням у формат оператора опції Method → Automatic. Тобто під час виконання оператора NDSolve відповідна програма, яка досить велика (на мові Mathematica вона містила б кілька тисяч рядків коду!) спочатку аналізує жорсткість задачі, а потім на основі оцінки жорсткості приймається рішення щодо використання явного чи неявного методу. Значення кроку і порядок методу змінюється, виходячи з умови забезпечення потрібної точності (див. підрозділ 10.8). Результати обчислень зі змінними кроком і порядком потім використовуються для побудови інтерполяційного полінома Лагранжа чи Ерміта (оператор InterpolatingFunction), за допомогою якого будуються відповідні таблиці та графіки.

Однак спільні явно-неявні процедури, розглянуті в підрозділі 10.9, під час розв'язання однієї й тієї ж самої задачі в пакеті Mathematica поки що не реалізовані.

Висновки

1. Моделями широкосмугових імпульсних систем є переважно жорсткі системи диференціальних рівнянь, шо визначає для їх розв'язання пріоритетне використання неявних методів.
2. Серед існуючих неявних методів найбільш ефективними з погляду мінімізації кількості кроків на заданому інтервалі (і як наслідок зменшення обсягу обчислень) є неявні методи змінного порядку і змінного кроку BDF і Гіра, які зберігають стійкість до шостого порядку включно. Ці методи передбачають зміну коефіцієнтів формул після кожної зміни значення кроку чи порядку методу. Процедура зміни порядку і кроку за заданими користувачем загальною похибкою і загальним часом обчислення у таких методах виконується автоматично.
3. Автоматична процедура вибору порядку методу і кроку враховує неоднаковий ступінь стійкості неявних методів різного порядку, яка залежить від A-стійкості, властивої неявним методам першого і другого порядків, і $A(\alpha)$-стійкості, характерної для методів третього–шостого порядків.
4. Для моделювання автоколивальних систем зі змінною частотою, в яких власні значення матриць Якобі розташовані на уявній осі комплексної площини, рекомендується використовувати неявні методи першого і другого порядків, області стійкості яких включають уявну вісь.
5. Можлива модифікація автоматичної процедури зміни кроку і порядку методу, яка передбачає використання методів з постійними коефіцієнтами, але з обов'язковим перерахунком інтерполяційних поліномів, побудованим на основі збережених значень коефіцієнтів.
6. Під час розв'язання жорстких задач перспективним є використання явно-неявних процедур, які забезпечують автоматичний перехід від явного методу до неявного (і навпаки), в залежності від поточних результатів обчислень.

Контрольні запитання та завдання

1. Розв'яжіть неявним методом Гіра третього порядку (10.19) рівняння $y' = t^2 - 15y^2$, $t_0 = 0$, $y_0 = 1$, яке було розв'язане раніше, в прикладі 10.1, неявним методом Ейлера. Необхідно, як і раніше, з кроком $\tau = 0{,}1$ знайти значення $y(0{,}2)$.
2. Найдіть умови стійкості формули наближення методу середньої точки $u_{n+1} = u_{n-1} + 2\tau f_n$ для випадку $\mathrm{Re}\,\lambda < 0$.
3. Визначіть, чи є серед наведених різницевих рівнянь такі, що забезпечують стійкі розв'язки:

   $y_{n+2} - 2y_{n+1} - 3y_n = 0;$

   $y_{n+2} - 4y_{n+1} - 5y_n = 0;$

   $y_{n+2} + 2y_{n+1} + y_n = 0.$

4. Покажіть, що всі розв'язки різницевого рівняння $y_{n+2} - 2\lambda y_{n+1} - y_n = 0$ стійкі для $n \to \infty$, якщо $-1 < \lambda < 1$. Для будь-яких інших значень λ у комплексній площині існує, хоча би один нестійкий розв'язок.

5. Побудуйте годограф стійкості в комплексній площині для неявного методу Гіра, користуючись пакетом Mathematica і характеристичним рівнянням (10.39):

$$\left(1 - \frac{2}{3}\tau\lambda\right)z^2 - \frac{4}{3}z + \frac{1}{3} = 0.$$

6. Знайдіть значення коефіцієнтів β_1 и β_2, яких бракує у формулі третього порядку $u_n = -4u_{n-1} + 5u_{n-2} + \tau(\beta_1 f(u_n) + \beta_2 f(u_{n-2}))$, користуючись методикою, описаною в підрозділі 10.5.

7. Отримайте значення коефіцієнтів у формулах методів Гіра третього, четвертого й п'ятого порядків для постійного часового кроку.

Розділ 11

Крайові задачі для звичайних диференціальних рівнянь

- Розв'язання лінійної крайової задачі комбінуванням двох задач Коші
- Метод прицілювання
- Метод скінченних різниць
- Власні значення однорідної крайової задачі
- Метод колокацій
- Метод Гальоркіна
- Метод найменших квадратів
- Метод скінченних елементів

У цьому розділі розглядаються наближені методи розв'язання двоточкових лінійних крайових задач для диференціальних рівнянь другого порядку [1, 4]. Наводиться різницева апроксимація крайової задачі і методи чисельного розв'язання отриманих рівнянь. Розглянуто також наближені аналітичні методи: колокацій, найменших квадратів і Гальоркіна.

11.1. Постановка задачі

Щоб знайти єдиний розв'язок звичайного диференціального рівняння, необхідно задати деякі допоміжні умови, що використовуються для обчислення інтегрування. Для рівняння n-го порядку потрібно n таких умов. Якщо ці умови задаються для одного значення незалежної змінної (зокрема, для одного кінця інтервалу, на якому необхідно знайти розв'язок), то говорять про *початкові умови* для задачі Коші. Ця задача докладно розглянута в розділах 8–10. Якщо ж додаткові умови задаються для значень незалежної змінної на різних кінцях інтервалу, то мають на увазі *крайову* задачу і граничні умови для неї.

Двоточкова крайова задача для рівняння другого порядку має такий вигляд:

$$\begin{gathered} y'' = f(x, y, y'), \\ a < x < b \end{gathered} \tag{11.1}$$

із граничними умовами

$$y(a) = \gamma_0, \quad y(b) = \gamma_1. \tag{11.2}$$

Перш ніж застосовувати будь-який чисельний метод, варто перевірити умови, що гарантують існування розв'язку задачі (11.1), (11.2). Наведена нижче теорема визначає загальні умови, що забезпечують існування й одиничність розв'язку [25].

Теорема. Припустимо, що $f(x, y, z)$ неперервна в області

$$D = \{(x, y, z): a \leqslant x \leqslant b, \ -\infty < y < \infty, \ -\infty < z < \infty\}$$

і що

$$\frac{\partial f}{\partial y} = f_y(x, y, z) \quad \text{і} \quad \frac{\partial f}{\partial z} = f_z(x, y, z)$$

теж неперервні на D. Якщо існує постійна $M > 0$, для якої виконуються умови

$$\begin{aligned} f_y(x, y, z) &> 0 \quad \text{для всіх} \quad (x, y, z) \in D, \\ f_z(x, y, z) &\leqslant M \quad \text{для всіх} \quad (x, y, z) \in D, \end{aligned} \tag{11.3}$$

то крайова задача (11.1), (11.2) має єдиний розв'язок $y(x)$ для $a \leqslant x \leqslant b$.

Найчастіше зустрічаються і найкраще вивчені двоточкові лінійні крайові задачі виду

$$L[y] = y'' + p(x)y' + q(x)y = f(x), \quad a < x < b, \tag{11.4}$$

$$l_a[y] = \alpha_0 y(a) + \beta_0 y'(a) = \gamma_0, \quad l_b[y] = \alpha_1 y(b) + \beta_1 y'(b) = \gamma_1, \tag{11.5}$$

де

$$\alpha_0^2 + \beta_0^2 \neq 0, \quad \alpha_1^2 + \beta_1^2 \neq 0.$$

Умови, які повинні задовольняти функції $p(x)$, $q(x)$ і $f(x)$, для того щоб задача (11.4), (11.5) мала єдиний розв'язок, випливають із теореми як наслідок.

Наслідок. Якщо $p(x)$ і $q(x)$ неперервні на D і $q(x) < 0$, то задача (11.4), (11.5) має єдиний розв'язок на $a \leqslant x \leqslant b$.

Граничні умови (11.5) визначають третю крайову задачу для рівняння (11.4). Якщо припустити, що $\beta_0 = \beta_1 = 0$, то умови (11.5) визначають першу крайову задачу, а коли $\alpha_0 = \alpha_1 = 0$ — другу.

Точне (аналітичне) розв'язання крайових задач — більш складна процедура, ніж знаходження розв'язку задачі Коші. Це спричинило появу великої кількості наближених методів. Ці методи можна розділити на дві групи: наближено-аналітичні методи, що дають наближений розв'язок крайової задачі на відрізку $[a, b]$ у вигляді конкретної аналітичної функції, і чисельні методи, що визначають розв'язок у вигляді табличної функції, заданої на сітці відрізка $[a, b]$. Нижче будуть розглянуті підходи до побудови і реалізації методів обох типів.

11.2. Розв'язання лінійної крайової задачі комбінуванням двох задач Коші

Припустимо, що розв'язок задачі (11.4), (11.5) будемо шукати у вигляді

$$y(x) = Aw(x) + v(x), \tag{11.6}$$

де A — деяка константа, $w(x)$ — функція, що задовольняє однорідне рівняння

$$L[w] = 0, \tag{11.7}$$

а $v(x)$ — функція, яка задовольняє неоднорідне рівняння

$$L[v] = f(x). \tag{11.8}$$

Через те, що рівняння (11.4) є лінійним, функція $y(x)$ буде його розв'язком для будь-якого A. Справді,

$$L[y] = L[Aw + v] = AL[w] + L[v] = f(x).$$

Якщо припустити, що розв'язок (11.6) задовольняє першу граничну умову (11.5) для будь-якого A, то отримаємо рівняння

$$l_a[y] = l_a[Aw + v] = Al_a[u] + l_a[v].$$

Ця гранична умова задовольняється, якщо покласти

$$l_a[w] = \alpha_0 w(a) + \beta_0 w'(a) = 0 \tag{11.9}$$

і

$$l_a[v] = \alpha_0 v(a) + \beta_0 v'(a) = \gamma_0. \tag{11.10}$$

Рівність (11.9) справедлива, коли прийняти, наприклад, що

$$w(a) = \beta_0, \quad w'(a) = -a_0. \tag{11.11}$$

Щоб задовольнити рівність (11.10), можна покласти:

$$v(a) = \frac{\gamma_0}{\alpha_0}, \quad v'(a) = 0, \quad \text{якщо} \quad \alpha_0 \neq 0 \tag{11.12}$$

чи

$$v(a) = 0, \quad v'(a) = \frac{\gamma_0}{\beta_0}, \quad \text{якщо} \quad \beta_0 \neq 0. \tag{11.13}$$

Врахуємо, що одночасно α_0 і β_0 на нуль не перетворюються через умову (11.5).

Таким чином, для розв'язання крайової задачі (11.4), (11.5) необхідно знайти розв'язок задач

$$w'' + p(x)w' + q(x)w = 0, \quad w(a) = \beta_0, \quad w'(a) = -\alpha_0 \tag{11.14}$$

та

$$v'' + p(x)v' + q(x)v = f(x) \tag{11.15}$$

з початковими умовами (11.12) чи (11.13). Для цього можна використати будь-який чисельний метод розв'язання задачі Коші для рівнянь другого порядку. Наближений розв'язок цих рівнянь отримуємо на відрізку $[a, b]$, у результаті чого стають відомими значення $w(b), w'(a), v(b), v'(a)$. Це дозволяє вибрати таку константу A, щоб функція (11.6) задовольняла не тільки рівняння (11.12) і першу граничну умову, але і другу граничну умову (11.5). Маємо

$$l_b[y] = l_b[Aw + v] = Al_b[w] + l_b[v] = \gamma_1,$$

звідки

$$A = \frac{\gamma_1 - l_b[v]}{l_b[w]} = \frac{\gamma_1 - \alpha_1 v(b) - \beta_1 v'(b)}{\alpha_1 w(b) + \beta_1 w'(b)}, \text{ якщо } \alpha_1 w(b) + \beta_1 w'(b) \neq 0. \tag{11.16}$$

Коли $\alpha_1 w(b) + \beta_1 w'(b) = 0$, то однорідна крайова задача

$$L[w] = 0, \quad l_a[w] = 0, \quad l_b[w] = 0$$

має нетривіальний розв'язок $w(x)$, який є ознакою виродженості початкової задачі (11.4) , (11.5).

Приклад 11.1

Розв'яжемо крайову задачу

$$\begin{aligned} y'' - (1 + x^2)y &= x^2 + 2x, \\ y(-1) = y(1) &= 0. \end{aligned} \tag{11.17}$$

Порівнюючи умови (11.17) із загальними граничними умовами, можна помітити, що

$$a = -1, \quad b = 1, \quad \alpha_0 = 1, \quad \beta_0 = 0, \quad \alpha_1 = 1, \quad \beta_1 = 0, \quad \gamma_0 = 0, \quad \gamma_1 = 0.$$

Будемо шукати розв'язок у вигляді (11.6). Тоді для функції $w(x)$, відповідно до (11.7), маємо задачу Коші

$$w'' - (1 + x^2)w = 0, \quad w(-1) = 0, \quad w'(-1) = -1.$$

Розв'язуючи будь-яким чисельним методом останню задачу, використовуючи Mathematica:

```
In[]:= a = -1; b = 1, α0 = 1; β0 = 0; γ0 = 0; α1 = 1; β1 = 0; γ1 = 0;
       S1 = NDSolve[{w''[x] - (1 + x^2)w[x] == 0, w[-1] == 0, w'[-1] == -1},
       w[x], {x, -1, 1}]; W[x_] = w[x]/. S1;
```

отримуємо розв'язок $w(x)$.

Задача Коші (11.15) з початковими умовами (11.12) для розглянутого випадку має такий вигляд:

$$v'' - (1 + x^2)v = x^2 + 2x, \quad v(-1) = 0, v'(-1) = 0.$$

Розв'язуючи цю задачу, отримаємо функцію $v(x)$:

```
In[]:= S2 = NDSolve[{v''[x] - (1 + x^2)v[x] == x^2 + 2x, v[-1] == 0, v'[-1] == 0},
       v[x], {x, -1, 1}]; V[x_] = v[x]/. S2;
```

Для визначення константи A за формулою (11.16) знайдемо значення похідних $w'(b)$, $w''(b)$ і обчислимо значення A:

```
In[]:= dW[x_] = D[W[x], x]; dV[x_] = D[V[x], x];
       A = (γ1 - α1V[b] - β1dV[b])/(α1W[b] + β1dW[b])
Out[]= {-0.334994}
```

Відповідно до рівняння (11.6) розв'язок отримаємо у вигляді

$$y(x) = -0{,}334994w(x) + v(x),$$

графік якого показано на рис. 11.1.

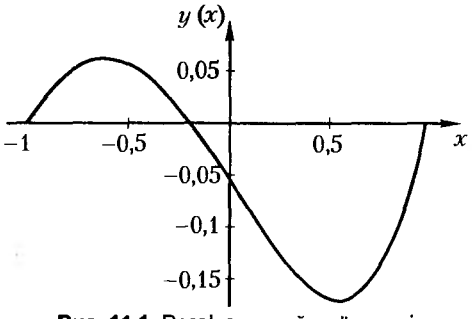


Рис. 11.1. Розв'язок крайової задачі

11.3. Метод прицілювання

Викладений вище метод редукції крайової задачі до задачі Коші має певні недоліки.

- Він не дозволяє використовувати методи розв'язання задачі Коші зі змінним порядком і змінним кроком. Розв'язки $w(x)$ і $v(x)$ повинні обчислюватись на сітці з однаковим кроком, інакше знайти їх комбінацію (11.6) буде неможливо.
- Використання методу, як правило, обмежується лише одновимірною лінійною задачею. Причина полягає в тому, що під час розв'язання системи рівнянь потрібно обчислювати не одне значення константи A (11.16), а матрицю **A**, що є далеко не простою задачею.
- Метод не придатний для розв'язання нелінійних крайових задач.

Ці недоліки спричинилися до появи нових методів. На практиці двоточкова крайова задача (лінійна чи нелінійна) звичайно розв'язується методом *прицілювання* (*стрільби*), назва якого запозичена із теорії артилерійської стрільби. Відповідно до цього методу розв'язок шуканого рівняння другого порядку

$$y'' = f(x, y, y')$$

із заданими граничними умовами

$$y(a) = \gamma_0, \quad y(b) = \gamma_1, \quad x \in [a, b]$$

знаходять у такий спосіб: ітераційним розв'язанням задачі Коші

$$\begin{gathered} y'' = f(x, y, y'), \\ y(a) = \gamma_0 \quad \text{і} \quad y'(a) = \sigma \end{gathered} \tag{11.18}$$

підбирається значення першої похідної $y'(a)$, для якої виконується друга крайова умова $y(b) = \gamma_1$.

Спочатку вибирається довільне значення $\sigma = \sigma_0$ і розв'язується задача Коші (11.18). Значення σ_0 бажано вибирати так, щоб наближений розв'язок на кінці інтервалу задовольняв умову $y_0(b) < \gamma_1$ (рис. 11.2). Потім вибирається $\sigma = \sigma_1$, і розв'язання задачі Коші повторюється. Тепер бажано вибрати його так, щоб виконувалась умова $y_1(b) > \gamma_1$ (рис. 11.2).

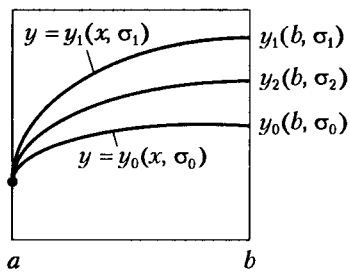


Рис. 11.2. Ілюстрація методу прицілювання

Після цього шляхом інтерполяції уточнюється значення σ_i для задач Коші з початковими умовами:

$$\begin{aligned} \sigma_2 &= \sigma_1 - (u(\sigma_1) - \gamma_1)\frac{\sigma_1 - \sigma_0}{u(\sigma_1) - u(\sigma_0)}, \\ \sigma_3 &= \sigma_2 - (u(\sigma_2 - \gamma_1))\frac{\sigma_2 - \sigma_1}{u(\sigma_2) - u(\sigma_1)}, \\ &\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots \end{aligned} \tag{11.19}$$

де $u(\sigma_i)$ — наближений розв'язок задачі Коші в точці b для вибраного значення σ_i.

Метод прицілювання є універсальним і використовується для розв'язання нелінійних диференціальних рівнянь n-го порядку. Слід зазначити, що довільний вибір початкового наближення σ_0 може привести до того, що задача (11.18) виявиться жорсткою (розділ 10) навіть у випадку, коли задача (11.1), (11.2) є добре обумовленою.

11.4. Метод скінченних різниць

Ідея методу скінченних різниць полягає в тому, що похідні в диференціальному рівнянні (11.4) і граничних умовах (11.5) заміняються їх скінченними різницями. Для цього спочатку введемо на відрізку $[a, b]$ сітку з кроком h:

$$h = \frac{b-a}{n} : \omega_h = \{x_i \mid x_i = x_0 + ih;\ i = 0, 1, ..., n;\ x_0 = a;\ x_n = b\}.$$

Позначимо через $y_i = y(x)$ точний розв'язок задачі (11.1) у i-му вузлі сітки, а через u_i — наближений розв'язок у цій точці. Заміняючи в кожному внутрішньому вузлі сітки похідні різницями, отримаємо різницеві рівняння:

$$\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + p_i \frac{u_{i+1} - u_{i-1}}{2h} + q_i u_i = f_i, \quad i = 1, ..., n. \tag{11.20}$$

Як відмічалось у розділі 5, симетричні різницеві апроксимації похідних першого і другого порядків мають похибку другого порядку відносно h, тобто $O(h^2)$. Це легко довести на основі розкладання в ряд Тейлора точного розв'язку рівняння. Дійсно, для вузлів x_{i-1} та x_{i+1} маємо

$$y_{i+1} = y_i + y_i' h + \frac{y_i''}{2} h^2 + \frac{y_i^{(3)}}{6} h^3 + ...,$$
$$y_{i-1} = y_i - y_i' h + \frac{y_i''}{2} h^2 - \frac{y_i^{(3)}}{6} h^3 + ...$$

з різниці яких отримуємо шуканий результат:

$$\frac{y_{i+1} - y_{i-1}}{2h} = y_i' + \frac{y_i^{(3)}}{3} h^2 + ... = y_i' + O(h^2),$$
$$\frac{y_{i+1} - 2y + y_{i-1}}{h^2} = y_i'' + \frac{y_i^{(4)}}{4!} h^2 + ... = y_i'' + O(h^2). \tag{11.21}$$

Знайдемо нев'язку різницевого рівняння

$$r_i = \frac{y_{i+1} - 2y + y_{i-1}}{h^2} - p_i \frac{y_{i+1} - y_{i-1}}{2h} - q_i y_i + f_i = -y_i'' - p_i y_i' - q_i y_i + f_i + O(h^2).$$

Оскільки $y(x)$ є точним розв'язком рівняння (11.4),

$$-y_i'' - p_i y_i' - q_i y_i + f_i = 0 \quad \text{та} \quad r_i = O(h^2). \tag{11.22}$$

Тому різницеве рівняння (11.21) апроксимує вихідне диференціальне рівняння (11.4) також із другим порядком відносно h.
Тепер апроксимуємо граничні умови скінченними різницями:

$$\alpha_0 u_0 + \beta_0 \frac{u_1 - u_0}{h} = \gamma_0, \quad \alpha_1 u_n + \beta_1 \frac{u_n - u_{n-1}}{h} = \gamma_1. \tag{11.23}$$

Знайдемо похибку апроксимації граничних умов. Нев'язки граничних умов (11.23) мають вигляд:

$$r_0 = -\alpha_0 y_0 - \beta_0 \frac{y_1 - y_0}{h} + \gamma_0, \quad r_n = -\alpha_1 y_n - \beta_1 \frac{y_n - y_{n-1}}{h} + \gamma_1.$$

Асиметрична апроксимація першої похідної на відміну від симетричної має глобальну похибку першого порядку відносно h, тобто $O(h)$. Це безпосередньо випливає з розкладання в ряд Тейлора

$$r_0 = -\alpha_0 y_0 - -\beta_0 y_0' - \beta_0 y_0'' h + O(h^2) = O(h),$$

із якого отримуємо

$$r_0 = O(h), \quad r_n = O(h).$$

Отже, граничні умови (11.23) апроксимуються з першим порядком за h. Порядок їх апроксимації можна підвищити до другого, наприклад, використовуючи співвідношення

$$\alpha_0 u_0 + \beta_0 \frac{-3u_0 + 4u_1 - u_2}{2h} = \gamma_0, \quad \alpha_1 u_n + \beta_1 \frac{u_{n-2} - 4u_{n-1} + 3u_n}{2h} = \gamma_1, \tag{11.24}$$

похибка апроксимації яких також пропорційна $O(h^2)$, як і для випадку симетричної апроксимації похідних. Це випливає із порівняння двох рядів Тейлора:

$$y_1 = y(x_0 + h) = y_0 + y_0' h + \frac{y_0''}{2} h^2 + \frac{y_0^{(3)}}{6} h^3 + \ldots,$$

$$y_2 = y(x_0 + 2h) = y_0 + 2h y_0' + \frac{y_0''}{2} (2h)^2 + \frac{y_0^{(3)}}{6} (2h)^3 + \ldots.$$

Якщо перший вираз помножити на 4 і відняти його від другого, отримаємо:

$$-3y_0 - 4y_1 + y_2 = 2y_0' h + \frac{4y_0^{(3)}}{6} h^3 + \ldots = 2y_0' h + O(h^3).$$

Після його підстановки у формулу (11.24) знаходимо нев'язку у вигляді:

$$r_0 = \alpha_0 y_0 + \beta_0 y_0' - \gamma_0 + O(h^2) = O(h^2),$$

тобто крайова умова апроксимується з другим порядком відносно h.
У такий же спосіб доводиться, що і друга гранична умова (11.23) апроксимується з другим порядком відносно h.

Розглянемо ще одну можливість апроксимації крайових умов типу (11.5) на прикладі умови

$$l_b[y] = \alpha_1 y(b) + \beta_1 y'(b) = \gamma_1.$$

Для цього за межами інтервалу $[a, b]$ вводиться додаткова точка $x_{n+1} = b + h$, за допомогою якої обчислюється перша похідна за симетричною формулою апроксимації:

$$\alpha_1 y_n + \frac{\beta_1 (u_{n+1} - u_{n-1})}{2h} = \gamma_1. \tag{11.25}$$

Точку u_{n+1} можна виключити, скориставшись співвідношенням (11.25) і різницевою апроксимацією диференціального рівняння (11.4) в кінцевій точці інтервалу x_n:

$$\frac{u_{n+1} - 2u_n + u_{n-1}}{h^2} + p_n \frac{u_{n+1} - u_{n-1}}{2h} + q_n u_n = f_n. \tag{11.26}$$

Отримуємо рівняння для граничної умови в точці n із порядком $O(h^2)$, яким можна замінити останнє рівняння в системі алгебраїчних рівнянь, одержаній у разі кусочно-різницевої апроксимації похідних у рівнянні (11.4).

Те ж саме можна зробити з першою умовою (11.5) і першим апроксимуючим рівнянням для u_1. Варто підкреслити, що врахування граничних умов різних типів впливає тільки на перше й останнє рівняння цієї системи.

Зведемо подібні члени в рівнянні (11.21) і отримаємо стандартне триточкове різницеве рівняння:

$$(1 - 0,5hp_i)u_{i-1} - (2 - h^2 q_i)u_i + (1 + 0,5hp_i)u_{i+1} = h^2 f_i, \quad i = 1, \ldots, n-1. \tag{11.27}$$

Включивши до системи рівнянь (11.25) різницеве рівняння (11.23) чи (11.24), отримаємо систему рівнянь, що містить $n+1$ рівняння з $n+1$ невідомими u_i.

Порівняємо ці два варіанти апроксимації крайової задачі. У першому з них система лінійних алгебраїчних рівнянь, утворена рівняннями (11.21) і (11.23), має тридіагональну матрицю коефіцієнтів, і її можна розв'язати методом прогону (розділ 3). Щоб застосувати метод прогону в другому випадку, слід створити

відповідну тридіагональну матрицю. Для цього потрібно з першого рівняння (11.27) для $i = 1$

$$(1 - 0,5hp_1)u_0 - (2 - h^2q_1)u_1 + (1 + 0,5hp_1)u_2 = h^2f_1$$

і першого рівняння (11.24) виключити u_2. Виконавши це, отримаємо:

$$h(2 + hp_1)\gamma_0 = (-h^2f_1 + (2 + 2hp_1 + h^2q_1)u_1)\beta_0 + \\ + u_0(h(2 + hp_1)\alpha_0 + (-2 - 2hp_1)\beta_0).$$

Маємо рівняння з двома невідомими — u_0 і u_1. Замінимо ним перше рівняння (11.24). Виконаємо такі ж перетворення з другим (11.24) і останнім рівнянням (11.27) для $i = n - 1$:

$$(1 - 0,5hp_{n-1})u_{n-2} - (2 - h^2q_{n-1})u_{n-1} - (1 - 0,5hp_{n-1})u_n = h^2f_{n-1}.$$

Виключивши з них u_{n-2}, знаходимо:

$$h(-2 + hp_{-1+n})\gamma_1 = (-h^2f_{-1+n} + (2 - 2hp_{-1+n} + h^2q_{-1+n})u_{-1+n})\beta_1 + \\ + u_n(h(-2 + hp_{-1+n})\alpha_1 + (-2 + 2hp_{-1+n})\beta_1).$$

Це рівняння містить дві невідомі — u_n і u_{n-1}. Замінимо ним друге рівняння (11.27). Два останні рівняння разом із (11.27) утворюють систему рівнянь із тридіагональною матрицею, що апроксимує вихідну крайову задачу (11.4), (11.5) з порядком $O(h^2)$. Цю систему також можна розв'язати методом прогону. В розділі 3 показано, що метод прогону є стійким, якщо матриця коефіцієнтів діагонально домінантна. Забезпечити діагональну домінантність можна обранням кроку h. Для цього необхідно, щоб для системи рівнянь (11.27) виконувались умови:

$$1 - 0,5p_ih > 0, 1 + 0,5p_ih > 0 \quad \text{і} \quad q_i < 0, \ i = 1, ..., n - 1.$$

Підсилюючи останні нерівності, маємо такі обмеження на величину кроку:

$$q(x) < 0, x \in [a, b] \quad \text{і} \quad h \leqslant \frac{2}{\max|p(x)|}, \quad x \in [a, b]. \tag{11.28}$$

Щоб задовольнялись умови (11.23), мають виконуватись нерівності

$$\alpha_0\beta_0 < 0 \quad \text{і} \quad \alpha_1\beta_1 > 0. \tag{11.29}$$

Наявність обмежень (11.28) і (11.29) свідчить про умовну стійкість розглянутого методу апроксимації.

Приклад 11.2

Розв'яжемо різницевим методом крайову задачу

$$y'' - \ln(x)y' - 2y = 1, \quad y(0,5) - y'(0,5) = 1, \quad y(1,5) + y'(1,5) = 0. \tag{11.30}$$

Відповідна різницева задача має вигляд:

$$(h+1)u_0 - u_1 = h,$$
$$(2 + h\ln(x_i))u_{i-1} - 4(1+h^2)u_i + (2 - h\ln(x_i))u_{i+1} = 2h^2, \quad i = 1, ..., n-1,$$
$$u_{n-1} - (1+h)u_n = 0.$$

Виберемо крок h, який забезпечує стійкість різницевої схеми (11.26):

$$h \leqslant \frac{2}{|\ln(x)|}, \quad x \in [0,5; 1,5].$$

Функція $\ln(x)$ монотонна, тому вона досягає найбільшого за модулем значення на одному з кінців інтервалу. Це значення

$$|\ln(0,5)| = 0,693147, \quad |\ln(1,5)| = 0,405465.$$

Таким чином, $h \leqslant 0,30103$ і умови (11.29) виконані. Щоб отримати розв'язок із достатньою точністю. Виберемо крок рівним $h = 0,2$. Формули методу прогону (3.13) для $\beta_i = 2 + h\ln(x_i)$, $\alpha_i = -4(1+h^2)$, $\gamma_i = 2 - h\ln(x_i)$, $b_i = 2h^2$ мають такий вигляд:

$$w_1 = \frac{1}{1+h}, \qquad v_1 = \frac{h}{1+h},$$
$$w_i = -\frac{2 - h\ln x_i}{(2 + h\ln(x_i))w_{i-1} - 4(1+h^2)}, \quad v_i = \frac{2h^2 - (2 + h\ln(x_i))v_{i-1}}{(2 + h\ln(x_i))w_{i-1} - 4(1+h^2)}.$$

Наведемо програму розв'язання задачі за допомогою пакета Mathematica:

```
In[]:= n = 50;
       h = (xn - x0)/n;
       x0 = 0.5;
       xn = 1.5;
       Array[W, n, 0]; Array[V, n, 0]; Array[u, {n+1, 2}, 0];
       W[0] = 1/(1 + h); V[0] = h/(1 + h); b = 4(1 + h^2); u[0,0] = x0; u[n,0] = xn;
       Do [le = Log[x0 + h*i]; a = 2 + h*le; c = 2 - h*le;
          W[i] = c/(b - a*W[i - 1]); V[i] = -(2*h^2 - a*V[i - 1])/b - a*W[i-1]), {i, n-1}];
```

Прямий прогін завершено. Обчислимо значення u_n:

```
In[]:= u[n,1] = V[n-1]/(1 + h - W[n-1])
Out[]= -0.120967
```

Виконаємо зворотний прогін:

```
In[]:= Do [u[n-i, 0] = xn - h*i; u[n-i, 1] = W[n-i]*u[n-i+1, 1] + V[n-i], {i, 1, n}];
```

Отримаємо розв'язок тієї ж задачі (10.26) за допомогою стандартного оператора пакета Mathematica з використанням методу прицілювання (підрозділ 11.8):

```
In[]:= EQ = NDSolve[{y''[x] - Log[x]*y'[x] - 2*y[x] == 1, y[0.5] - y'[0.5] == 1,
       y[1.5] + y'[1.5] == 0}, y, {x, 0.5, 1.5}];
```

Побудуємо графіки обох розв'язків на одному рисунку (11.3):

```
In[]:= Array[z, {n+1, 2}, 0];
       s[x_] = y[x]/.EQ;
       Do [z[i,0] = x0 + i*h; z[i,1] = s[x0 + i*h][[1]], {i,0,n}];
In[]:= <<Graphics`MultipleListPlot`
In[]:= <<Graphics`Legend`
In[]:= MultipleListPlot[Array[z, {n+1, 2}, 0], Array[u, {n+1, 2}, 0],
       PlotLegend → {"Mathematica","Різницевий метод"}, PlotJoined → {False, True},
       PlotPosition → {0.3, -0.5}]
```

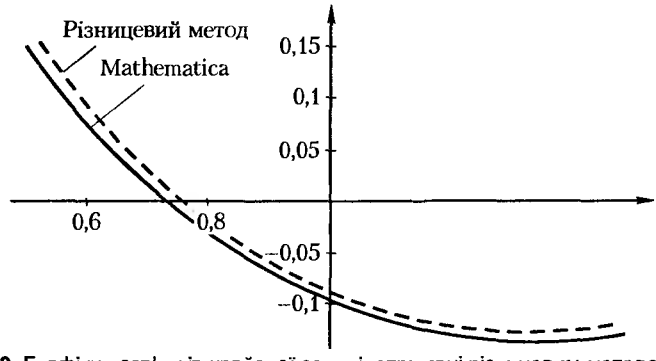


Рис. 11.3. Графіки розв'язків крайової задачі, отримані різницевим методом (11.26) і за допомогою стандартного оператора пакета Mathematica

Приклад 11.3

Проілюструємо ефективність застосування екстраполяції Річардсона для розв'язання крайових задач на прикладі рівняння

$$y'' + y = x$$

з граничними умовами $y(0) = 1$, $y(\pi/2) = \pi/2 - 1$.
Диференціальне рівняння апроксимується лінійною системою рівнянь виду:

$$\mathbf{Au} = h^2(\mathbf{x} - \mathbf{u}) - \mathbf{r}, \tag{11.31}$$

де

$$\mathbf{A} = \begin{bmatrix} -2 & 1 & \dots & \dots & \dots & 0 \\ 1 & -2 & 1 & \dots & \dots & 0 \\ 0 & 1 & -2 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 & 1 & -2 \end{bmatrix}, \quad \mathbf{r} = \begin{bmatrix} 1 \\ 0 \\ \dots \\ \dots \\ \pi/2 - 1 \end{bmatrix}.$$

Результати розв'язання системи (11.31) на інтервалі $[1, \pi/2]$ для $n = 5$ і $n = 10$ відрізків систематизовані в табл. 11.1. Там же наведені уточнення, отримані екстраполяцією Річардсона (8.20) з поправками $\Delta/(q_1^p - 1) = \Delta/3$ ($p = 2$ і $q_1 = 2$). Оскільки відомий точний розв'язок крайової задачі $y(x) = \cos x - \sin x + x$, останній стовпець таблиці містить значення похибки ε.

Таблиця 11.1. Результати застосування екстраполяції Річардсона для уточнення розв'язку крайової задачі

$2x/\pi$	$m = 5$ $u(x, 0{,}1\pi)$	$m = 10$ $u(x, 0{,}05\pi)$	$10^6\Delta/3$	Значення за Річардсоном	$10^6\varepsilon$
0	1,000000	1,000000	0	1,000000	0
0,1		0,988402			
0,2	0,956572	0,956291	–94	0,956197	–2
0,3		0,908337			
0,4	0,849741	0,849597	–48	0,849549	–1
0,5		0,785398			
0,6	0,721056	0,721199	48	0,721247	1
0,7		0,662460			
0,8	0,614224	0,614505	94	0,614599	1
0,9		0,582395			
1,0	0,570796	0,570796	0	0,570796	0

11.5. Власні значення однорідної крайової задачі

Однорідна крайова задача

$$y'' + \lambda y = 0, \quad y(0) = 0, \quad y(1) = 0 \tag{11.32}$$

завжди має тривіальний розв'язок $y(x) = 0$. Однак для практичного застосування важливе значення мають *нетривіальні* розв'язки, існування і вигляд яких залежать від параметра λ. Ці особливі значення параметра λ називають *власними значеннями крайової задачі*, а відповідні до них нетривіальні розв'язки — *власними функціями*. Наприклад, задача (11.32) має розв'язок

$$y(x) = a\cos(x\sqrt{\lambda}) + b\sin(x\sqrt{\lambda}),$$

з якого випливає:

$$y(0) = 0 \Rightarrow a = 0,$$
$$y(1) = 0 \Rightarrow \sqrt{\lambda} = \pi n, \quad n = 0, \pm 1, \pm 2, \ldots.$$

Отже, власні значення задачі (11.32):

$$\lambda = n^2\pi^2, \tag{11.33}$$

власні функції:

$$y(x) = b\sin(nx\pi). \tag{11.34}$$

Приклад 11.4

Знайдемо власні значення різницевого аналога крайової задачі (11.32)

$$y'' + \lambda y = 0, \quad y(0) = 0, \quad y(1) = 0.$$

Для цього на інтервалі $[0, 1]$ виділимо три відрізки ($n = 3$) і з кроком $h = 1/3$ введемо різницеву апроксимацію другої похідної:

$$y_n'' \approx \frac{u_{n+1} - 2u_n + u_{n-1}}{h^2}.$$

Тоді для рівняння (11.32) можна записати систему рівнянь виду:

$$\begin{cases} \dfrac{-2u_1 + u_2}{h^2} + \lambda u_1 = 0, \\ \dfrac{u_1 - 2u_2}{h^2} + \lambda u_2 = 0. \end{cases} \tag{11.35}$$

Якщо для задачі існує нетривіальний розв'язок ($u_1 \neq 0, u_2 \neq 0$), то визначник системи рівнянь (11.35) повинен дорівнювати нулю (11.35):

$$\begin{vmatrix} -2 + \lambda h^2 & 1 \\ 1 & -2 + \lambda h^2 \end{vmatrix} = 0,$$

звідки

$$\lambda h^2 - 2 = \pm 1.$$

Для $h = 1/3$ отримуємо значення $\lambda_1 = 9$ (точне значення $\pi^2 \approx 9{,}8696$) і $\lambda_2 = 27$ (точне значення $4\pi^2 \approx 39{,}48$).

Похибку обчислень власних значень можна записати за допомогою виразу

$$\lambda(h) = \lambda + C_2 h^2 + C_3 h^3 + C_4 h^4 + \ldots.$$

Точність обчислення власних значень задачі можна істотно підвищити, якщо застосувати екстраполяцію Річардсона (8.20). Для цього додатково розіб'ємо інтервал і знайдемо визначник системи рівнянь, побудованої для кроку $h = 1/4$. Результати оцінювання значення λ_1 зведені в табл. 11.2. Бачимо, що оцінка $\lambda_1 = 9{,}8517$ досить близька до точного значення $\lambda_1 = \pi^2 \approx 9{,}8696$.

Таблиця 11.2. Уточнені власні значення крайової задачі

h	λ_i	$\Delta/(7/9)$	Значення за Річардсоном	ε
1/3	9	0,4791	9,8517	–0,0176
1/4	9,3726			

Дані табл. 11.2 свідчать про те, що застосування екстраполяції Річардсона дозволяє отримати оцінку λ_1 із малою похибкою, навіть коли на інтервалі невелика кількість точок, що сприяє зменшенню обсягу обчислень, необхідних для знаходження визначників.

11.6. Метод колокацій

У методі колокацій розв'язок крайової задачі (11.4), (11.5) шукається у вигляді функції

$$u(x) = \varphi_0(x) + \sum_{i=1}^{n} c_i \varphi_i(x). \tag{11.36}$$

де $\varphi_i(x)$, $i = 0, 1, ..., n$ — лінійно незалежні, двічі диференційовані базисні функції, визначені на відрізку $[a, b]$. Функція $\varphi_0(x)$ повинна задовольняти задані граничні умови (11.5):

$$\begin{aligned} l_a[\varphi_0] &= \alpha_0\varphi_0(a) + \beta_0\varphi_0'(a) = \gamma_0, \\ l_b[y] &= \alpha_1\varphi_0(b) + \beta_1\varphi_0'(b) = \gamma_1, \end{aligned} \tag{11.37, а}$$

а функції $\varphi_i(x)$, $i = 0, 1, ..., n$ — відповідні однорідні граничні умови, тобто

$$\begin{aligned} l_a[\varphi_i] &= \alpha_0\varphi_i(a) + \beta_0\varphi_i'(a) = 0, \\ l_b[\varphi_i] &= \alpha_1\varphi_i(b) + \beta_1\varphi_i'(b) = 0, \\ & i = 0, 1, ..., n. \end{aligned} \tag{11.37, б}$$

Через лінійність граничних умов функція $u(x)$ у (11.36) задовольняє граничним умовам (11.24) для будь-яких значень c_i. Наприклад, у точці $x = a$ маємо

$$\begin{aligned} l_a[u] &= l_a\left[\varphi_0(a) + \sum_{i=1}^{n} c_i\varphi_i(a)\right] = \\ &= l_a[\varphi_0(a)] + l_a\left[\sum_{i=1}^{n} c_i\varphi_i(a)\right] = \gamma_0 + \sum_{i=1}^{n} c_i l_a[\varphi_i(a)] = \gamma_0. \end{aligned}$$

Аналогічно для $x = b$ отримаємо

$$l_b[u] = \gamma_1.$$

Суть методу колокацій полягає в тому, що для заданих n точок на відрізку $[a, b]$, названих вузлами колокації, підбирають значення c_i так, щоб отримана при цьому функція $u(x)$ (11.36) задовольняла рівняння (11.4) у кожному з вузлів колокації:

$$\begin{aligned} L[u(x_j)] &= L\left[\varphi_0(x_j) + \sum_{i=1}^{n} c_i\varphi_i(x_j)\right] = \\ &= L[\varphi_0(x_j)] + \sum_{i=1}^{n} c_i L[\varphi_i(x_j)] = f(x_j), \\ & 0 < j < n, \end{aligned} \tag{11.38}$$

де

$$L[\varphi_i(x_j)] = \varphi_i''(x_j) + p(x_j)\varphi_i'(x_j) + q(x_j)\varphi_i(x_j), \quad j = 0, 1, ..., n.$$

Покладемо

$$\begin{aligned} a_{ij} &= L[\varphi_i(x_j)], \\ b_j &= f(x_j) - L[\varphi_0(x_j)], \end{aligned} \tag{11.39}$$

тоді (11.39) матиме стандартний вигляд системи лінійних алгебраїчних рівнянь:

$$\sum_{i=1}^{n} a_{ij}c_i = b_j, \quad j = 0, 1, ..., n, \tag{11.40}$$

відносно коефіцієнтів c_i. Якщо розв'язати цю систему і підставити отримані значення коефіцієнтів у вираз (11.36), отримаємо наближений розв'язок $u(x)$.

Точність розв'язку крайової задачі методом колокацій залежить від типу базисних функцій $\varphi_i(x)$. У конкретних задачах вибір цих функцій слід здійснювати з урахуванням апріорної інформації про розв'язки задачі або на основі емпіричних даних. Якщо така інформація відсутня, можна використати запропонований в [4] метод. Нехай $\varphi_0(x)$ — це лінійна функція

$$\varphi_0(x) = dx + g, \tag{11.41}$$

параметри якої визначимо таким чином, щоб вона задовольняла неоднорідні граничні умови (11.5), тобто з системи рівнянь

$$\begin{aligned} d(a\alpha_0 + \beta_0) + g\alpha_0 &= \gamma_0, \\ d(b\alpha_1 + \beta_1) + g\alpha_1 &= \gamma_1. \end{aligned} \tag{11.42}$$

Функції $\varphi_i(x)$ можна задати у вигляді:

$$\varphi_i(x) = s_i(x-a)^{i+1} + (x-a)^{i+2}, \quad i = 1, 2, ..., n. \tag{11.43}$$

Очевидно, що за будь-яких s_i функція (11.43) задовольняє умову (11.37, *а*). Значення s_i, за якого буде задовольнятися друга умова (11.37, *б*), таке:

$$s_i = -\frac{\alpha_1(b-a)^2 + \beta_1(i+2)(b-a)}{\alpha_1(b-a) + \beta_1(i+1)}. \tag{11.44}$$

Якщо в умовах (11.37, *а*, *б*) $\beta_0 = 0$, то можливий інший вибір, а саме:

$$\begin{aligned} \varphi_i(x) &= s_i(x-a)^i + (x-a)^{i+1}, \\ s_i &= -\frac{\alpha_1(b-a)^2 + \beta_1(i+1)(b-a)}{\alpha_1(b-a) + i\beta_1}. \end{aligned} \tag{11.45}$$

Приклад 11.5

Розв'яжемо задачу:

$$L[y(x)] = x^4y'' + x^6y' - x^5y = 6 - 3x^2, \quad 1 < x < 2, \tag{11.46}$$

$$y(1) = 1, \quad 3y(2) + y'(2) = 0,5. \tag{11.47}$$

Порівнюючи умови (11.47) із загальними граничними умовами, бачимо, що:

$$a = 1, \quad b = 2, \quad \alpha_0 = 0,$$
$$\beta_0 = 0, \quad \gamma_0 = 1,$$
$$\alpha_1 = 3, \quad \beta_1 = 1, \quad \gamma_1 = 0,5.$$

Розглянемо, як необхідно вибирати систему базисних функцій. Спочатку знайдемо коефіцієнти функції (11.42). Розв'язавши систему рівнянь (11.43) для умов (11.47), отримаємо

$$\varphi_0(x) = 0,625x + 1,625.$$

Сформуємо базисні функції $\varphi_i(x)$ у вигляді (11.43):

$$\varphi_1(x) = 1,25(x-1) + (x-1)^2,$$
$$\varphi_2(x) = -1,2(x-1)^2 + (x-1)^3.$$

Визначимо диференціальний оператор $L[u(x)]$, що відповідає лівій частині рівняння (11.4) для наближеного розв'язку:

$$u(x) = \varphi_0(x) + c_1\varphi_1(x) + c_2\varphi_2(x).$$

Виразимо нев'язку диференціального рівняння для наближеного розв'язку $u(x)$:

$$r(x) = f(x) - L[u(x)] = 6 - 3x^2 + (-2c_1 + 8,4c_2)x^4 + (1,625 + 2,25 - 8,2c_2)x^5 +$$
$$+ (0 + 0c_1 + 0c_2)x^6 + (-c1 + 4,2c_2)x^7 - 2c_2x^8.$$

Виберемо дві точки колокації з координатами $x_1 = 1,33$, $x_2 = 1,667$. Задамо вимогу, щоб нев'язки в точках колокацій були рівними нулю, й отримаємо систему рівнянь для обчислення c_1, c_2:

$$5,7334 - 4,32347c_1 + 3,48198c_2 = 0,$$
$$13,0213 - 22,2528c_1 - 9,71251c_2 = 0.$$

Розв'яжемо отримані рівняння і запишемо розв'язок задачі:

$$u(x) = \varphi_0(x) + 0,8457\varphi_1(x) - 0,5967\varphi_2(x) =$$
$$= -0,597(-2,47684 + x)(3,27523 - 3,14045x + x^2).$$

Тепер розв'яжемо цю крайову задачу (11.46), (11.47) за допомогою пакета Mathematica:

```
In[]:= a = 1; b = 2; f[x_] = 6 - 3x^2;
       DSolve[{-x^5y[x] + x^6y'[x] + x^4y''[x] == f[x], y[a] == 1, y'[b] + 3y[b] == 0.5}, y[x], x]
Out[]= {{y[x] → 1/x^2}}
```

Наведемо графіки наближеного і точного розв'язків, що дорівнюють $y = x^{-2}$.

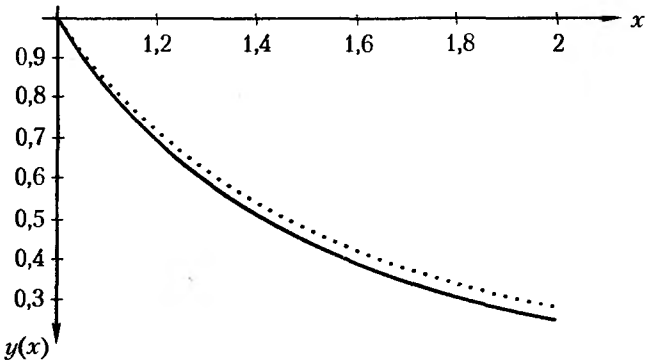


Рис. 11.4. Точний і наближений роза'язки задачі (11.46), (11.47)

На рис. 11.4 видно, що, використовуючи три базисних функції, отримали досить точний наближений розв'язок крайової задачі (11.46), (11.47). Максимальна помилка досягається, коли $x = 2$, і дорівнює ϵ = U[2] - 0.25 = -0.330594.

11.7. Метод Гальоркіна

Як і в методі колокацій, у методі Гальоркіна наближений розв'язок крайової задачі (11.4), (11.5) шукаємо у вигляді

$$u(x) = \varphi_0(x) + \sum_{i=1}^{n} c_i \varphi_i(x), \tag{11.48}$$

де $\varphi_i(x)$, $i = 0, 1, 2, \ldots$ — лінійно незалежні, двічі диференційовані базисні функції, визначені на відрізку $[a, b]$. Функція $\varphi_0(x)$ повинна задовольняти задані граничні умови (11.37, *а*), а функції $\varphi_i(x)$, $i = 0, 1, 2, \ldots$ — відповідні однорідні граничні умови (11.37, *б*).

Необхідно, щоб система базисних функцій $\varphi_i(x)$, $i = 0, 1, 2, \ldots$ була ортогональною на відрізку $[a, b]$, тобто

$$\int_a^b \varphi_i(x)\varphi_j(x)dx = 0 \quad \text{при} \quad i \neq j \quad \text{і} \quad \int_a^b \varphi_i^2(x)dx \neq 0,$$

і повною. Остання вимога означає, що не повинно існувати ніякої іншої відмінної від нуля функції, яка ортогональна до всіх функцій $\varphi_i(x)$, $i = 0, 1, 2, \ldots$.

Використуючи наближений розв'язок (11.48) знайдемо нев'язку:

$$r(x, c_1, c_2, \ldots, c_n) = L[\varphi_0(x)] + \sum_{i=1}^{n} c_i L[\varphi_i(x)] - f(x). \tag{11.49}$$

Підставимо наближений розв'язок (11.54) у рівняння (11.4) і знайдемо нев'язку:

$$r(x, c_1, c_2, ..., c_n) = L[\varphi_0(x)] + \sum_{i=1}^{n} L[\varphi_i(x)] - f(x), \tag{11.55}$$

абсолютна величина якої для $a \leqslant x \leqslant b$ повинна бути якомога меншою. Тому вимагатимемо, щоб виконувалася умова

$$I = \int_a^u r^2(x, c_1, c_2, ..., c_n)\, dx \to \min. \tag{11.56}$$

Значення інтегралу будуть мінімальними за умов:

$$\begin{aligned}
\partial I_{c_1} &= 2\int_a^b r\frac{\partial r}{\partial c_1}dx = 0,\\
\partial I_{c_2} &= 2\int_a^b r\frac{\partial r}{\partial c_2}dx = 0,\\
\partial I_{c_3} &= 2\int_a^b r\frac{\partial r}{\partial c_3}dx = 0,\\
&\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots\\
\partial I_{c_n} &= 2\int_a^b r\frac{\partial r}{\partial c_n}dx = 0.
\end{aligned}$$

На основі цих умов формується система лінійних рівнянь для обчислення коефіцієнтів $c_1, c_2, ..., c_n$.

Приклад 11.7

Розглянемо розв'язання задачі

$$x^4y'' + x^6y' - x^5y = 6 - 3x^2, \quad 1 < x < 2, \tag{11.57}$$

$$y(1) = 1, \quad 3y(2) + y'(2) = 0{,}5 \tag{11.58}$$

наведеної в прикладі 11.5.

У даному випадку будемо використовувати ті ж самі базисні функції і ту ж форму зображення розв'язку, що і в прикладі 10.2:

$$\begin{aligned}
\varphi_0(x) &= 1{,}625 - 0{,}625x,\\
\varphi_1(x) &= -1{,}25(-1+x) + (-1+x)^2,\\
\varphi_2(x) &= -1{,}2(-1+x)^2 + (-1+x)^3,\\
u(x) &= \varphi_0(x) + c_1\varphi_1(x) + c_2\varphi_2(x).
\end{aligned}$$

Підставимо $u(x)$ в (11.55) і знайдемо нев'язку:

$$\begin{aligned}
r(x) = 6 - 3x^2 + (-2c_1 + 8{,}4c_2)x^4 + (1{,}625 + 2{,}25 - 8{,}2c_2)x^5 +\\
+ (0 + 0c_1 + 0c_2)x^6 + (-c_1 + 4{,}2c_2)x^7 - 2c_2x^8.
\end{aligned}$$

Отримаємо систему рівнянь для обчислення коефіцієнтів c_1, c_2:

$$\int_1^2 r\frac{\partial r}{\partial c_1}dx = -433,582 + 962,617c_1 + 883.262c_2 = 0,$$

$$\int_1^2 r\frac{\partial r}{\partial c_2}dx = -372,153 + 883,262c_1 + 879,967c_2 = 0.$$

Розв'яжемо цю систему і знайдемо

$$c_1 = 0,789449, \quad c_2 = -0,369488.$$

Сформуємо наближений розв'язок:

$$u(x) = -0,369488(-2,71191 + x)(4,20565 - 3,6247x + x^2).$$

Наведемо графіки розв'язку, отриманого методом найменших квадратів $u(x)$, і точне значення x^{-2} (рис. 11.6).

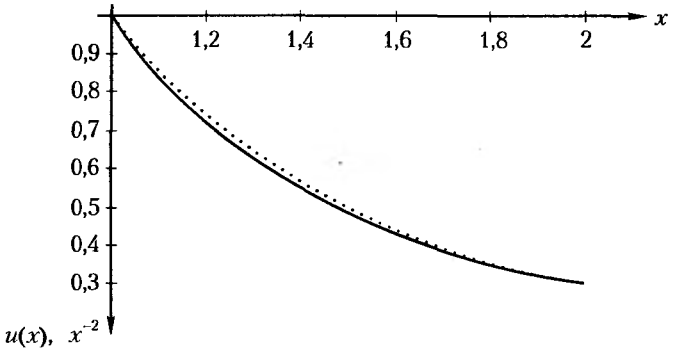


Рис. 11.6. Графіки точного розв'язку крайової задачі (11.57), (11.58), рівного $y = x^{-2}$, і наближеного роза'язку $u(x)$, отриманого методом найменших квадратів у Mathematica

Із використанням трьох базисних функцій отримано досить точний наближений розв'язок крайової задачі (11.57), (11.58), бо криві розв'язків практично збіглися.

11.9. Метод скінченних елементів

Метод Гальоркіна накладає певні обмеження на вибір системи базисних функцій, які залежать від граничних умов крайової задачі. Це обмеження значно ускладнює реалізацію методу, особливо під час розв'язання задач математичної фізики. Це обмеження можна подолати, якщо для апроксимації розв'язку використовувати систему простих базисних функцій, які залежать від координат вузлів на відрізку $[a, b]$. У цьому випадку розв'язання крайової задачі зводиться до формування і розв'язання системи лінійних алгебраїчних рівнянь, тому метод отримав назву методу скінченних елементів. Його часто використовують для розв'язання дво- та тривимірних диференціальних рівнянь із частинними похідними. У цьому розділі розглянемо основні ідеї методу на прикладі розв'язання крайових задач для ЗДР.

Шукатимемо наближений розв'язок задачі

$$L[y] = y'' + p(x)y' + q(x)y = f(x), \quad a < x < b \tag{11.59}$$

як лінійну комбінацію простих однотипних функцій

$$u(x) = \sum_{i=1}^{n-1} c_i \varphi_i(x), \tag{11.60}$$

що мають вигляд

$$\varphi_i(x) = \begin{cases} \dfrac{x - x_{i-1}}{h}, & \text{якщо} \quad x \in [x_{i-1}, x_i], \\ -\dfrac{x - x_{i+1}}{h}, & \text{якщо} \quad x \in [x_i, x_{i+1}], \\ 0, & \text{якщо} \quad x \notin [x_{i-1}, x_{i+1}], \end{cases} \tag{11.61}$$

і, як правило, називаються фінітними. Графік однієї з таких функцій наведено на рис. 11.7, де видно, що функція не дорівнює нулю тільки на інтервалі (x_{i-1}, x_{i+1}). Щодо множини фінітних функцій, які задаються на відрізку $[a, b]$, відомо, що вони лінійно незалежні (більш того, ортогональні в спеціальній енергетичній нормі) і утворюють повну систему в просторі $L_1[a, b]$. Це дає підставу використати їх як базисні функції в методі Гальоркіна.

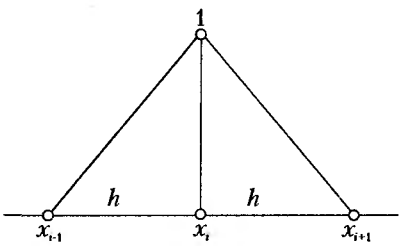


Рис. 11.7. Графік фінітної функції $\varphi_i(x)$

Запишемо умову ортогональності (11.50):

$$\sum_{i=1}^{n-1} c_i \int_a^b \varphi_k(x) L[\varphi_i(x)]\,dx = \int_a^b \varphi_k(x) f(x)\,dx, \quad k = 1, 2, \ldots, n-1 \tag{11.62}$$

і отримаємо систему лінійних алгебраїчних рівнянь для знаходження невідомих c_i. Праві частини цих рівнянь позначимо через d_k і отримаємо для їх обчислення вираз

$$\begin{aligned} d_k &= \int_a^b \varphi_k(x) f(x)\,dx = \sum_{k=1}^{n-1} \int_{x_{k-1}}^{x_{k+1}} \varphi_k(x) f(x)\,dx = \\ &= \int_{x_{k-1}}^{x_k} \frac{x - x_{k-1}}{h} f(x)\,dx + \int_{x_k}^{x_{k+1}} \frac{x - x_{k+1}}{h} f(x)\,dx. \end{aligned} \tag{11.63}$$

Коефіцієнти системи рівнянь (11.62) позначимо через

$$a_{ki} = \int_a^b \varphi_k(x) L[\varphi_i(x)]\,dx.$$

Знайдемо вирази для коефіцієнтів системи рівнянь a_{ki} з невідомими c_i. Підставляючи в останній вираз $L[\varphi_i(x)]$, отримаємо

$$a_{ki} = \int_a^b \varphi_k(x)(\varphi_i''(x) + p(x)\varphi_i'(x) + q(x))\,dx.$$

Перший з інтегралів у цьому виразі обчислимо по частинах:

$$\int_a^b \varphi_k(x)\varphi_i''(x)\,dx = \varphi_k(x)\varphi_i'(x)\Big|_a^b - \int_a^b \varphi_k'(x)\varphi_i'(x)\,dx.$$

Оскільки за граничних умов (11.60) використовуються $n-1$ базисних функцій від $\varphi_1(x)$ до $\varphi_{n-1}(x)$ і всі вони в точках a і b дорівнюють 0, то

$$\int_a^b \varphi_k(x)\varphi_i''(x)\,dx = -\int_a^b \varphi_k'(x)\varphi_i'(x)dx.$$

Тоді вираз для обчислення a_{ki} набуває вигляду:

$$\begin{aligned} a_{ki} &= -\int_a^b \varphi_k'(x)\varphi_i'(x)\,dx + \int_a^b p(x)\varphi_k'(x)\varphi_i(x)\,dx + \int_a^b q(x)\varphi_k(x)\varphi_i(x)\,dx = \\ &= \sum_{j=0}^{n} \int_{x_j}^{x_{j+1}} (-\varphi_k'(x)\varphi_i'(x) + p(x)\varphi_k'(x)\varphi_i(x) + q(x)\varphi_k(x)\varphi_i(x))\,dx. \end{aligned} \tag{11.64}$$

Для обчислення a_{ki} треба знайти значення похідних від фінітних функцій. Із цією метою диференціюємо (11.61) і отримуємо:

$$\varphi'(x) = \begin{cases} \dfrac{1}{h}, & \text{якщо} \quad x \in [x_{i-1}, x_i], \\ -\dfrac{1}{h}, & \text{якщо} \quad x \in [x_i, x_{i+1}], \\ 0, & \text{якщо} \quad x \notin [x_{i-1}, x_{i+1}]. \end{cases} \tag{11.65}$$

Функція відмінна від нуля тільки на інтервалі (x_{i-1}, x_{i+1}). Крім того, на одному і тому ж інтервалі ненульовими є дві базисні функції і їх похідні з сусідніми індексами (рис. 11.8), тобто на інтервалі (x_i, x_{i+1}) відмінні від нуля $\varphi_i(x)$, $\varphi_{i+1}(x)$, $\varphi_i'(x)$, $\varphi_{i+1}'(x)$ і т. д.

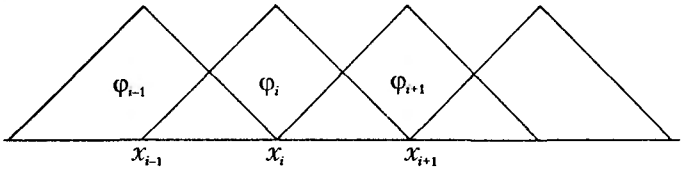


Рис.11.8. Система фінітних функцій

У виразі для a_{ki} (11.64) добутки $\varphi_k'(x)\varphi_i'(x)$, $\varphi_k'(x)\varphi_i(x)$, $\varphi_k(x)\varphi_i(x)$ можна вважати відмінними від нуля тому, що на елементарному інтервалі не дорівнюють нулю фінітні функції та їх похідні, які мають сусідні індекси у випадках, коли $i-1 \leqslant k \leqslant i+1$. А це означає, що

$$a_{ki} = 0 \quad \text{для} \quad |i-k| > 1, \tag{11.66}$$

тобто матриця системи $\mathbf{A} = \{a_{ki}\}$ (11.62) є тридіагональною матрицею. Її ненульові елементи обчислюються таким чином. Формули для діагональних елементів отримаємо, приймаючи $i = k$ у виразі (11.64):

$$\begin{aligned} a_{ii} = &\int_{x_{i-1}}^{x_i}\left[-\frac{1}{h^2} + p(x)\frac{x-x_{i-1}}{h^2} + q(x)\frac{(x-x_{i-1})^2}{h^2}\right]dx + \\ &+ \int_{x_i}^{x_{i+1}}\left[-\frac{1}{h^2} + p(x)\frac{x-x_{i+1}}{h^2} + q(x)\frac{(x-x_{i+1})^2}{h^2}\right]dx. \end{aligned} \tag{11.67}$$

Для $i = k+1$, отримаємо формули для елементів правої бічної діагоналі матриці **A**:

$$a_{ii+1} = \int_{x_i}^{x_{i+1}}\left[\frac{1}{h^2} - p(x)\frac{(x-x_{i+1})}{h^2} - q(x)\frac{(x-x_{i+1})(x-x_i)}{h^2}\right]dx, \tag{11.68}$$

а для $i = k-1$ — лівої:

$$a_{ii-1} = \int_{x_{i-1}}^{x_i}\left[\frac{1}{h^2} - p(x)\frac{(x-x_{i-1})}{h^2} - q(x)\frac{(x-x_{i-1})(x-x_i)}{h^2}\right]dx, \tag{11.69}$$

Три останні вирази визначають систему алгебраїчних рівнянь (11.62) для невідомих коефіцієнтів c_i.

Розглянемо розв'язання задачі (11.59) у випадку неоднорідних граничних умов

$$l_a[y] = y(a) = \alpha, \quad l_b[y] = y(b) = \beta \tag{11.70}$$

і зведемо її до розв'язання задачі з однорідними граничними умовами. Для цього введемо заміну:

$$y = v + w, \quad \text{де} \quad w = \alpha + \frac{\beta-\alpha}{b-a}(x-a).$$

Двічі диференціюючи цю функцію і підставляючи вирази для похідних у рівняння (11.59), отримаємо крайову задачу з однорідними граничними умовами:

$$L[v] = v'' + p(x)v' + q(x)v = f(x) - \frac{\beta - \alpha}{b - a}p(x) - q(x)w(x), \qquad (11.71)$$
$$v(a) = 0, \quad v(b) = 0.$$

Приклад 11.8

Розв'яжемо задачу

$$y''(x) + x^2 y'(x) - xy(x) = e^{2,5x}$$

з граничними умовами

$$y[0] = 0, \quad y[1] = 2$$

методом скінчених елементів. Спочатку зведемо її до розв'язання задачі з однорідними граничними умовами. Для цього виконаємо заміну

$$y = v + w, \quad \text{де} \quad w = 2x.$$

Відповідно до формули (11.71) отримаємо крайову задачу з однорідними граничними умовами:

$$v''(x) + x^2 v'(x) - xv(x) = F(x), \quad v(0) = 0, v(1) = 0,$$
$$F(x) = e^{2,5x} - 2x^2 - xw(x).$$

Розіб'ємо відрізок [0,1] на n частин і розв'яжемо цю задачу методом скінченних елементів. Для цього визначимо коефіцієнти системи алгебраїчних рівнянь (11.67), (11.68), (11.69), склавши програму для пакета Mathematica. Опишемо змінні, що використовуються, і введемо початкові дані:

```
In[]:= w[x_] = 2 x; F[x_]:= E^2.5 x - 2 x^2 - x w[x];
       n = 10;
       Array[a, {n-1, n-1}]; Array[b, n-1]; Array[x, n+1, 0];
       x[0] = 0; x[n] = 1; h = (x[n] - x[0])/n;
       Do [x[i] = x[0] + i*h, {i,1,n}];
       p[x_]:= x^2; q[x_]:= x;
```

Визначимо функції, які обчислюють інтеграли у формулах (11.67), (11.68) (11.69):

```
In[]:= P[l_]:= Integrate[p[x] (x - x[l - 1]) + q[x] (x - x[l - 1])^2,
              {x, x[l-1], x[l]}]/h/h;
       H[l_]:= Integrate[p[x] (x - x[l + 1]) + q[x] (x - x[l + 1])^2,
              {x, x[l], x[l+1]}]/h/h;
```

Обчислимо коефіцієнти системи алгебраїчних рівнянь:

```
In[]:= Do [Do [If [Abs[(i - k)] > 1, a[i,k] = 0,
                 If [i == k, a[i,k] = N[-2/h + H[i] + P[i]],
                  If[i == k - 1, a[i,k] = N[1/h - P[i]],
                      a[i,k] = N[1/h - H[i] ] ] ] ], {i,1,n-1} ],
              {k,1,n-1} ]
```

Виведемо матрицю коефіцієнтів

```
In[]:= A = [Array[a, {n-1,n-1}];  MatrixForm[Array[a, {n-1,n-1}]]
Out[]//MatrixForm=
```

$$\begin{pmatrix} -20 & 9.995 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 10.02 & -20 & 9.98 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10.045 & -20 & 9.955 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10.08 & -20 & 9.92 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10.125 & -20 & 9.875 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10.18 & -20 & 9.82 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10.245 & -20 & 9.755 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 10.32 & -20 & 9.68 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10.405 & -20 \end{pmatrix}$$

Тепер обчислимо праві частини рівнянь:

```
In[]:= Do [b[i] = 1/h (Integrate[F[x] (x - x[i - 1]), {x, x[i - 1], x[i]}]-
                       Integrate[F[x] (x - x[i + 1]), {x, x[i], x[i + 1]}]), {i, 1, n-1}];
```

Виведемо їх значення:

```
In[]:= B = Array[b, n-1]
Out[]= {0.117158, 0.226983, 0.328808, 0.422633, 0.508458,
        0.586283, 0.656108, 0.717933, 0.771758}
```

Розв'яжемо систему алгебраїчних рівнянь для обчислення значень v:

```
In[]:= V = LinearSolve[A, B]
Out[]= {-0.170396, -0.32924, -0.465978, -0.570922, -0.634955,
        -0.649119, -0.6041, -0.489561, -0.293282}
```

Тепер можна записати наближений розв'язок початкової неоднорідної крайової задачі як $y = v + w$:

```
In[]:= Array[u, n+1, 0]; u[0] = w[x[0]]; u[n] = w[x[n]];
       Do [u[i] = V[[i]] + w[x[i]], {i,1,n-1} ]; Y = Array[y, {n+1,2}];
       Do [y[i,1] = N[x[I - 1]]; y[i,2] = u[i-1], {i,1,n+1} ]; Y = Array[y, {n+1,2}]
Out[]= {{0., 0.},       {0.1, 0.0296042}, {0.2, 0.0707596}, {0.3, 0.134022},
        {0.4, 0.229078} {0.5, 0.365045},  {0.6, 0.550881},  {0.7, 0.7959},
        {0.8, 1.11044}, {0.9, 1.50672},   {1., 2}}
```

І нарешті побудуємо графік розв'язку (рис. 11.9):

```
In[]:= G2 = ListPlot[Y, AxesLabel → {"x", "u(x)"}, PlotStyle → PointSize[0.02]]
```

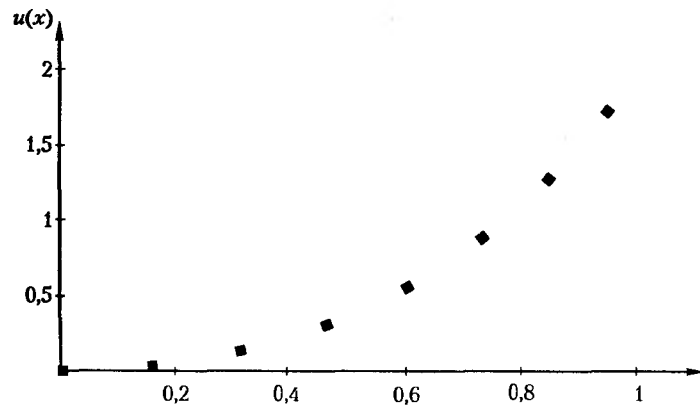


Рис. 11.9. Наближений розв'язок крайової задачі

У кінці розділу покажемо, як отримати розв'язок прикладу 11.9 за допомогою стандартного оператора Mathematica:

```
In[]:= NDSolve[{y''[x] + x^2y'[x] - x*y[x] == E^2.5x, y[0] == 0, y[1] == 2}, y, {x, 0, 1}]
Out[]= {{y → InterpolatingFunction[{{0.,1.}}, <>]}}
```

і покажемо графіки розв'язків, отриманих методом кінцевих елементів і за допомогою оператора NDSolve пакета Mathematica (рис. 11.10).

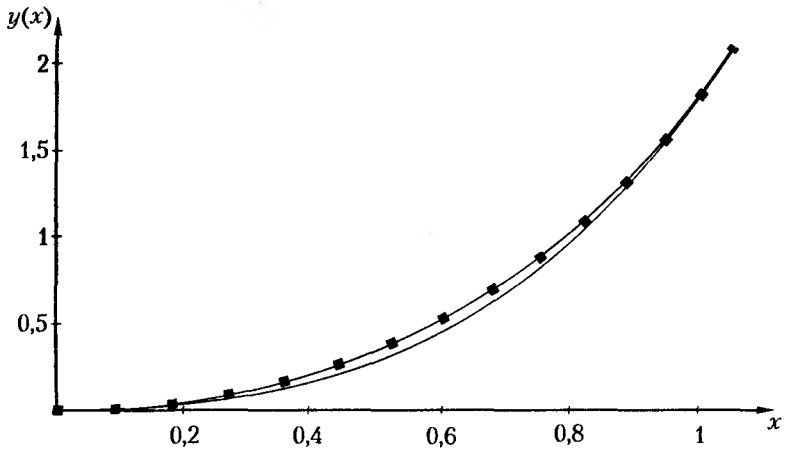


Рис. 11.10. Наближені розв'язки крайової задачі, отримані за допомогою стандартного оператора пакета Mathematica і методом скінченних елементів

Висновки

1. Крайова задача для звичайних диференціальних рівнянь є набагато складнішою, ніж задача Коші. Одним із підходів до розв'язання цієї задачі є зведення її до задачі Коші зі змінними початковими умовами. Розв'язок задачі отримують багаторазовим розв'язанням задачі Коші.
2. У загальному випадку для розв'язання двоточкової крайової задачі (одно- чи багатовимірної, лінійної чи нелінійної) доцільно застосовувати метод прицілювання, а для розв'язання окремих лінійних одновимірних задач — метод композиції двох розв'язків задачі Коші з різними початковими умовами.
3. Ефективним методом розв'язання лінійної крайової задачі для диференціального рівняння другого порядку є метод скінченних різниць, у якому використовуються різницеві схеми апроксимації для похідних першого і другого порядків. У результаті крайова задача перетворюється на задачу розв'язання системи лінійних рівнянь із тридіагональною матрицею. Цю систему можна розв'язати методом прогону. Використання екстраполяції Річардсона дозволяє отримати розв'язок із малою похибкою навіть у випадку, коли кількість точок на інтервалі розв'язку невелика.

4. Метод скінченних різниць дозволяє також обчислювати власні значення і власні функції крайової задачі, які визначають нетривіальні розв'язки однорідної крайової задачі.
5. Метод скінченних різниць можна застосовувати і для розв'язання нелінійних крайових задач, але в цьому випадку необхідно лінеаризовувати нелінійні функції, що входять в умову задачі. Процес розв'язання передбачає для уточнення значень розв'язку в кожній точці інтервалу виконання ітерацій Ньютона–Рафсона.
6. Розв'язок крайової задачі у вигляді апроксимуючого аналітичного виразу отримують методами колокацій, Гальоркіна і найменших квадратів введенням базисних функцій, які враховують граничні умови.
7. Коефіцієнти для базисних функцій та їх композиції, які апроксимують розв'язок крайової задачі, у методі колокацій вибирають з умови нульової нев'язки в обраних вузлах інтервалу розв'язку, у методі найменших квадратів — з умови мінімуму квадрату нев'язки, а в методі Гальоркіна — з умови ортогональності нев'язки до обраних базисних функцій.
8. У сучасних математичних пакетах розв'язання крайових задач для рівнянь з частинними похідними конкуренцію розглянутим методам складає метод скінчених елементів, що базується на концепціях метода Гальоркіна за умови спеціального вибору базисних функцій.

Контрольні запитання та завдання

1. Переконайтеся, що функція $y(x) = x^2 - 0,2525826491x - 2,528442297x\ln(x)$ є розв'язком крайової задачі $y'' = (1/x)y' - (1/x^2)y = 1$ з умовами $y(0,5) = 1$ і $y(3) = -1$.
2. Розв'яжіть крайову задачу $y'' = -(2x/1 + x^2)y' + (2/1 + x^2)y = 1$ з умовами $y(0) = 1,25$, $y(4) = -0,95$ на інтервалі $0 < x < 4$ зведенням до задач Коші. Порівняйте отриманий розв'язок із чисельним розв'язком, визначеним у пакеті Mathematica. Побудуйте графіки розв'язків.
3. Розв'яжіть крайову задачу $y'' + 2y' - 2y = e^{-x} + \cos 2x$ з умовами $y(0) = 1$, $y(1) = 0,1$ на інтервалі $0 < x < 1$ різницевим методом. За допомогою пакета Mathematica знайдіть точний розв'язок і визначте похибку наближеного розв'язку.
4. Розв'яжіть крайову задачу $y'' - 4y' - 4y = xe^{-x}\cos x$ з граничними умовами $y(0) = 1$, $y(1) = -0,5$ на інтервалі $0 < x < 1$ різницевим методом. За допомогою пакета Mathematica знайдіть точний розв'язок і визначте похибку наближеного розв'язку.
5. Розв'яжіть крайову задачу $y'' + x^2y' - xy = 15xe^{-x}\cos x$ з граничними умовами $y(0) = 0$, $y(1) = 0$ на інтервалі $0 < x < 1$ методом колокацій. За допомогою Mathematica знайдіть точний розв'язок і порівняйте обидва розв'язки.

6. Розв'яжіть крайову задачу $y'' + xy' + y = 15x$ з граничними умовами $y(0) = 1$, $y(1) = 0$ на інтервалі $0 < x < 1$ методом Гальоркіна. За допомогою пакета Mathematica знайдіть наближений розв'язок і порівняйте обидва розв'язки.

7. Розв'яжіть крайову задачу $y'' + x^2y' + xy = x\sin 2x$ з граничними умовами $y(1) = 0$, $y(2) = 1$ на інтервалі $1 < x < 2$ методом найменших квадратів. За допомогою пакета Mathematica знайдіть наближений розв'язок і порівняйте обидва розв'язки.

8. Розв'яжіть крайову задачу $y'' + x^2y' + xy = x\sin 2x$ з граничними умовами $y(1) = 0$, $y(2) = 1$ на інтервалі $1 < x < 2$ методом колокацій і Гальоркіна. Порівняйте обидва розв'язки.

9. Розв'яжіть крайову задачу $y'' + x^2y' + y = x\sin x$ з граничними умовами $y(1) = 0$, $y(2) = 1$ на інтервалі $1 < x < 2$ методом найменших квадратів і колокацій. Порівняйте обидва розв'язки.

10. Знайдіть мінімальне власне значення $\lambda_{\min}$ задачі $\frac{d}{dx}\left[(1+x^2)\frac{dy}{dx}\right] + \lambda y = 0$, $y(-1) = y(1) = 0$, застосовуючи метод скінченних різниць, для $h = 2/3$ і $h = 2/5$, і апроксимацію типу $\frac{1}{h^2}\left[1+\left(x_n + \frac{h}{2}\right)^2 (y_{n+1} - y_n) - \left(1+\left(x_n - \frac{h}{2}\right)^2 (y_n - y_{n-1})\right)\right]$ для лівої частини рівняння $\frac{d}{dx}\left[(1+x^2)\frac{dy}{dx}\right]$.

Розділ 12

Розв'язання рівнянь із частинними похідними

- Типи диференціальних рівнянь із частинними похідними
- Метод скінченних різниць
- Умови розв'язання різницевих рівнянь
- Ітераційні методи
- Прямі методи
- Метод скінченних елементів

Рівняння із частинними похідними описують багато фізичних процесів у таких галузях, як механіка суцільних середовищ, термодинаміка, квантова механіка, електродинаміка, теорія пружності й багато інших. Тому розділ математики, що вивчає властивості можливих розв'язків рівнянь з частинними похідними, називається *математичною фізикою*, а самі рівняння найчастіше називають *рівняннями математичної фізики*.

Аналітичний розв'язок рівнянь із частинними похідними вдається отримати лише в окремих практично важливих випадках, і тому значення чисельних методів для розв'язання задач, які описуються за допомогою цих рівнянь, дуже важливе. Математичними моделями багатьох фізичних процесів є лінійні диференціальні рівняння другого порядку. Тому основна увага у цьому розділі приділяється саме цім рівнянням.

12.1. Рівняння математичної фізики

Постановку задач для рівнянь математичної фізики наведемо для класичних рівнянь параболічного, гіперболічного й еліптичного типів. У цих рівняннях найчастіше незалежними змінними є час і просторові координати, від яких залежить функція розв'язку. Такі рівняння описують велику кількість реальних фізичних процесів, властивості яких змінюються не тільки в часі, але й у просторі.

Задача називається стаціонарною, якщо її розв'язок не залежить від часу, і нестаціонарною — якщо така залежність існує. Задачі з однією просторовою змінною називаються одновимірними, з двома змінними — двовимірними, з трьома – тривимірними.

Наведемо рівняння канонічної форми, що має дві просторові змінні і є лінійним відносно других похідних:

$$a(x_1, x_2)\frac{\partial^2 u}{\partial x_1^2} + 2b(x_1, x_2)\frac{\partial^2 u}{\partial x_1 \partial x_2} + c(x_1, x_2)\frac{\partial^2 u}{\partial x_2^2} = f\left(x_1, x_2, u, \frac{\partial u}{\partial x_1}, \frac{\partial u}{\partial x_2}\right).$$

Коефіцієнти a, b, c — це функції, які двічі неперервно-диференційовані і не дорівнюють нулю одночасно. Залежно від значень цих функцій розрізняють кілька типів квазілінійних диференціальних рівнянь з частинними похідними другого порядку. Щоб визначити тип рівняння у заданій точці (x_1, x_2) області простору, обчислимо значення $D = b^2 - ac$. Диференціальне рівняння є *параболічним*, якщо $D = 0$, *гіперболічним* — коли $D > 0$ і *еліптичним* — коли $D < 0$. Слід зазначити, що тип певного рівняння може змінюватись залежно від значень (x_1, x_2) координат точки.

Класичним рівнянням параболічного типу є рівняння теплопровідності, гіперболічного типу — хвильове рівняння, а еліптичного типу — рівняння Лапласа.

Рівняння параболічного типу

До рівнянь параболічного типу належать рівняння теплопровідності чи дифузії виду

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} + f(x,t), \quad 0 < x < l, \tag{12.1}$$

де a — коефіцієнт теплопровідності (якщо u — температура) і масоперенесення (якщо u — концентрація, тиск у задачах фільтрації). Оскільки рівняння містить похідну за часом, для його розв'язання необхідно додатково задавати як початкові (для $t = 0$), так і граничні умови (для $x = 0$, $x = l$, $t > 0$) (рис. 12.1).

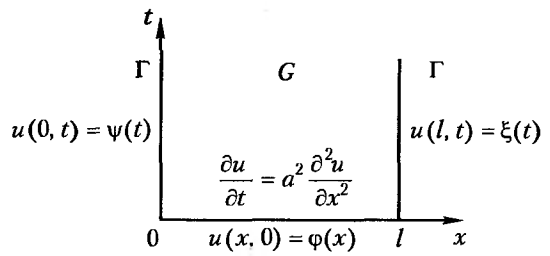


Рис. 12.1. Область визначення розв'язку одновимірного рівняння твплопровідності

Для рівняння теплопровідності за граничних умов

$$\begin{aligned} u(0, t) &= \psi(t), \quad x = 0, t > 0, \\ u(l, t) &= \xi(t), \quad x = l, t > 0 \end{aligned} \tag{12.2}$$

і початкової умови

$$u(x, 0) = \varphi(t), \quad 0 \leqslant x \leqslant l, t = 0 \tag{12.3}$$

маємо *першу* мішану крайову задачу, інакше задачу Коші з початковими умовами.

Рівняння теплопровідності за граничних умов

$$\begin{aligned} \frac{\partial u(0,t)}{\partial x} &= \psi(t), \quad x = 0, t > 0, \\ \frac{\partial u(l,t)}{\partial x} &= x(t), \quad x = l, t > 0 \end{aligned} \tag{12.4}$$

і початкової умови (12.3) визначає *другу* мішану крайову задачу.

Рівняння теплопровідності за граничних умов

$$\begin{aligned} \alpha_0 u(0,t) + \beta_0 \frac{\partial u(0,t)}{\partial x} &= \psi(t), \quad x = 0, t > 0, \\ \alpha_1 u(l,t) + \beta_1 \frac{\partial u(l,t)}{\partial x} &= \xi(t), \quad x = l, t > 0 \end{aligned} \tag{12.5}$$

і початкової умови (12.3) визначає *третю* мішану крайову задачу.

Для нескінченної області ставлять таку задачу Коші для рівняння теплопровідності:

$$u(x,0) = \varphi(t), \quad 0 \leqslant x \leqslant \infty, t = 0.$$

Рівняння гіперболічного типу

Прикладом рівнянь гіперболічного типу є хвильове рівняння

$$\frac{\partial^2 u}{\partial t^2} = a \frac{\partial^2 u}{\partial x^2} + f(x,t), \quad 0 < x < l, t > 0, \tag{12.6}$$

що описує малі подовжні коливання стрижня і поперечні коливання струни, де u — відхилення від положення рівноваги і a — швидкість розповсюдження збурення. Хвильове рівняння описує процес розповсюдження малих акустичних коливань.

Початкові умови для хвильового рівняння мають такий вигляд:

$$u(x,0) = \varphi_1(t), \quad \frac{\partial u(x,0)}{\partial t} = \varphi_2(t), \quad 0 \leqslant x \leqslant l, t = 0. \tag{12.7}$$

Граничні умови першого, другого і третього родів для хвильового рівняння задаються виразами (12.2), (12.4) і (12.5).

Задача Коші у нескінченній області для хвильового рівняння формулюється у такий спосіб:

$$u(x,0) = \varphi_1(t), \quad \frac{\partial u(x,0)}{\partial t} = \varphi_2(t), \quad -\infty < x < \infty, t = 0.$$

Рівняння еліптичного типу

Прикладом рівнянь еліптичного типу є рівняння Лапласа:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0, \quad (x, y) \in G \tag{12.8}$$

чи рівняння Пуассона:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y), \quad (x, y) \in G. \tag{12.9}$$

Ці рівняння описують потік ідеальної рідини в стаціонарних потоках, стаціонарний розподіл температури або напруженості електричних чи магнітних полів. Рівняння Лапласа описує ці процеси у разі відсутності джерел енергії чи стоків, а рівняння Пуассона — ті ж самі процеси за наявності розподілених в області G джерел, що задаються правою частиною рівняння $-f(x, y)$.

Оскільки рівняння Лапласа і Пуассона — стаціонарні, то в постановці задачі задаються тільки граничні умови (рис. 12.2). Залежно від граничних умов маємо:

- першу крайову задачу для рівняння Лапласа (задача Діріхле):

$$u|_{\Gamma} = \varphi(x, y), \quad (x, y) \in \Gamma; \tag{12.10}$$

- другу крайову задачу для рівняння Лапласа (задача Неймана):

$$\left.\frac{\partial u}{\partial \mathbf{n}}\right|_{\Gamma} = \varphi(x, y), \quad (x, y) \in \Gamma; \tag{12.11}$$

- третю крайову задачу для рівняння Лапласа:

$$\alpha u + \beta \left.\frac{\partial u}{\partial \mathbf{n}}\right|_{\Gamma} = \varphi(x, y), \quad (x, y) \in \Gamma. \tag{12.12}$$

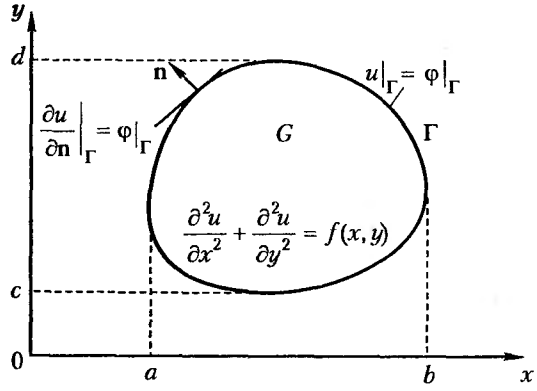


Рис. 12.2. Область визначення розв'язку двовимірного еліптичного рівняння

При цьому в другій і третій крайових задачах похідні слід обчислювати у напрямку зовнішньої нормалі $\mathbf{n}$ відносно області G.

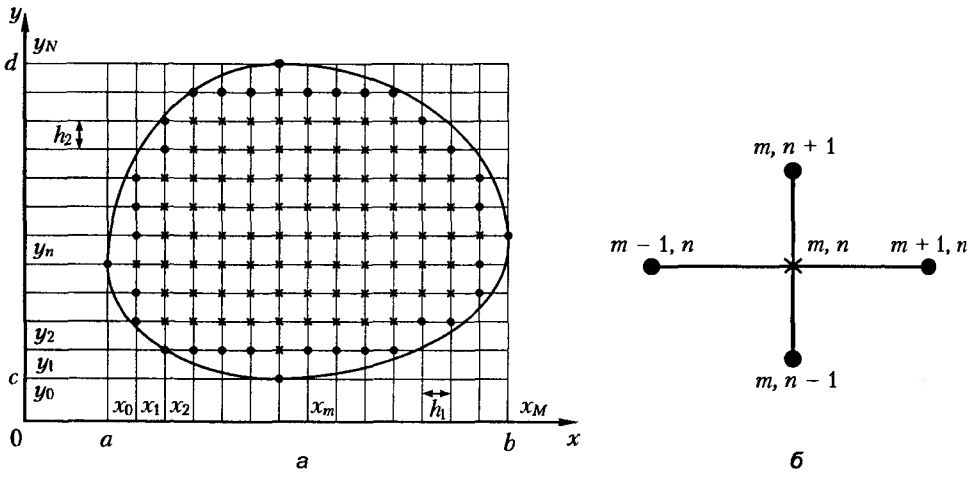
12.2. Основні поняття методу сіток

На практиці розв'язання крайових задач для рівнянь із частинними похідними за допомогою аналітичних методів часто неможливе чи пов'язане зі значними обчислювальними труднощами. У цьому випадку використовують наближені чисельно-аналітичні або чисельні методи. Методи цих типів є універсальними, на відміну від чисто аналітичних, і їх можна застосовувати для розв'язання більш складних задач. Одним із таких наближених методів є метод сіток, або скінченних різниць.

У методі сіток розв'язання рівняння із частинними похідними, як правило, зводиться до розв'язання систем лінійних алгебраїчних рівнянь із розрідженими матрицями [10]. Реалізація методу здійснюється у три етапи.

1. Область неперервного аргументу чи аргументів замінюється дискретною множиною вузлів, яка називається різницевою сіткою. У ній виділяють внутрішні й граничні вузли. Функція дискретного аргументу, визначена на різницевій сітці, називається сітковою функцією.
2. Диференціальне рівняння, а також початкові й граничні умови замінюються (апроксимуються) різницевими аналогами. У результаті диференціальне рівняння зводиться до системи алгебраїчних (різницевих) рівнянь, яка називається *різницевою схемою*. Така система повинна мати єдиний розв'язок. Необхідно, щоб зі збільшенням кількості вузлів сітки розв'язок різницевої схеми наближався (збігався) до розв'язку початкового диференціального рівняння.
3. Розв'язується система алгебраїчних рівнянь (у більшості випадків матриця системи рівнянь має дуже велику розмірність і є розрідженою).

Основні ідеї методу сіток розглянемо на прикладі задачі Діріхле для рівняння Пуассона (12.9) із граничною умовою (12.10), де Г — границя області G (рис. 12.3, *а*), у якій шукається розв'язок $u(x, y)$, що задовольняє рівняння (12.8) і граничну умову (12.10).



а *б*

Рис. 12.3. Область визначення розв'язку еліптичного рівняння:
а — сіткова область; *б* — шаблон різницевої схеми

Спочатку область G неперервної зміни аргументів із границею Г заміняють її сітковою областю G_h із границею Γ_h. Для цього проводять лінії $x_m = const$ і $y_n = const$, так що $x_m = mh_1$, $m = 0, 1, ..., M$ і $y_n = nh_2$, $n = 0, 1, ..., N$. Величини h_1 і h_2, які називаються кроками сітки, у загальному випадку можуть бути різними. Точки перетину ліній $x_m = const$ і $y_n = const$ називають вузлами сітки. Розрізняють два типи вузлів — внутрішні та граничні. Внутрішніми називають такі вузли, для яких чотири сусідніх вузли (по два в кожному напрямку) належать області $G + \Gamma$.

Замінимо диференціальний оператор Лапласа різницевим оператором. Із цією метою виберемо шаблон різницевої схеми — систему вузлів, які використовують для заміни похідних скінченними різницями. Шаблон, що містить p точок, називається p-точковим. Для апроксимації других похідних, що входять до оператора Лапласа, використовуємо п'ятиточковий шаблон, зображений на рис. 12.3, *б*. Нехай $h_1 = h_2 = h$. Розкладаючи точний розв'язок $u(x,y)$ у ряд Тейлора в околі точки (x_m, y_n), маємо:

$$u(x_m \pm h, y_n) = u(x_m, y_n) \pm h\frac{\partial u(x_m, y_n)}{\partial x} + \frac{h^2}{2!}\frac{\partial^2 u(x_m, y_n)}{\partial x^2} \pm \frac{h^3}{3!}\frac{\partial^3 u(x_m, y_n)}{\partial x^3} + O(h^4),$$

$$u(x_m, y_m \pm h) = u(x_m, y_n) \pm h\frac{\partial u(x_m, y_n)}{\partial y} + \frac{h^2}{2!}\frac{\partial^2 u(x_m, y_n)}{\partial y^2} \pm \frac{h^3}{3!}\frac{\partial^3 u(x_m, y_n)}{\partial y^3} + O(h^4).$$

Для скорочення запису введемо індекси для значень функції $u_{m,n}, u_{m-1,n}$ у вузлах шаблону відповідно (див. рис. 12.3). Тоді вирази для похідних запишемо у вигляді:

$$\frac{\partial^2 u_{m,n}}{\partial x^2} = \frac{u_{m-1,n} + 2u_{m,n} + u_{m+1,n}}{h^2} + O(h^2), \qquad (12.13)$$

$$\frac{\partial^2 u_{m,n}}{\partial y^2} = \frac{u_{m,n-1} + 2u_{m,n} + u_{m,n+1}}{h^2} + O(h^2). \qquad (12.14)$$

Використовуючи (12.13) і (12.14), введемо позначення для сіткової функції у вузлах $v_{m,n}, v_{m-1,n}, v_{m,n-1}, v_{m,n+1}$ і запишемо різницеве рівняння, що відповідає рівнянню Пуассона (12.9) на п'ятиточковому шаблоні, у такий спосіб:

$$\frac{v_{m-1,n} - 2v_{m,n} + v_{m+1,n}}{h^2} + \frac{v_{m,n-1} - 2v_{m,n} + v_{m,n+1}}{h^2} = f_{m,n}. \qquad (12.15)$$

Перепишемо систему сіткових рівнянь у вигляді:

$$v_{m-1,n} + v_{m+1,n} - 4v_{m,n} + v_{m,n-1} + v_{m,n+1} = h^2 f_{m,n}. \qquad (12.16)$$

Подібні рівняння можна записати для кожного внутрішнього вузла. Оскільки другі похідні, що входять у рівняння (12.9), апроксимовані з другим порядком за h, різницева схема (12.15), яка апроксимує рівняння (12.9), має такий же порядок апроксимації.

12.2.1. Апроксимація граничних умов

Розглянемо апроксимацію граничних умов першого роду (12.10). Якщо вузол (m, n) вважається граничним, але не належить границі Г, значення сіткової функції $v_{m,n}$ у цьому вузлі покладемо рівним значенню функції $\varphi(x, y)$ у точці границі Г, найближчій до цього вузла. Наприклад, якщо вузол B — граничний вузол, а вузол A — найближчий вузол границі, тоді гранична умова (12.10) буде записана у вигляді $v(B) = \varphi(A)$, як на рис. 12.4, *а*.

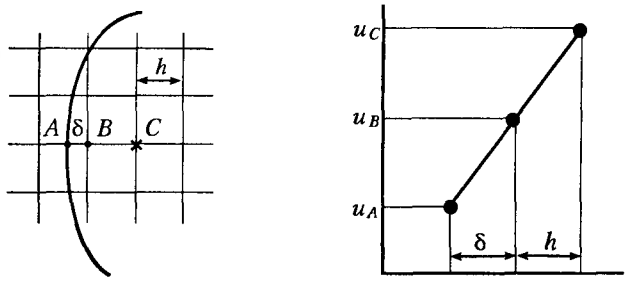


Рис. 12.4. Апроксимація граничних умов:
а — об'єднання з найближчою точкою; *б* — застосування лінійної інтерполяції

У цьому випадку схема апроксимації граничних умов має лише перший порядок за h. Переконаємося в цьому і знайдемо нев'язку

$$r(B) = \varphi(A) - u(B) = \varphi(A) - u(x_A + \delta, y_A) =$$
$$= \varphi(A) - u(x_A, y_A) - \delta\frac{\partial\tilde{u}}{\partial x} = -\delta\frac{\partial\tilde{u}}{\partial x}.$$

Альтернативою простому перенесенню за апроксимації граничних умов є лінійна інтерполяція. Вона полягає в тому, що значення сіткової функції в точці B обчислюється лінійною інтерполяцією її значень у точках A і C (рис. 12.4, *б*), тобто

$$\frac{v(C) - \varphi(A)}{v(B) - \varphi(A)} = \frac{h + \delta}{\delta}.$$

Тоді одержимо рівняння

$$(h + \delta)v(B) - \delta v(C) = h\varphi(A), \qquad (12.17)$$

що апроксимує граничну умову з другим порядком за h.

Розглянемо апроксимацію граничних умов другого роду (12.11). Нехай точка B є одним із граничних вузлів, а точка A є найближчою до неї точкою границі (рис. 12.5). Припустимо, що сітка є квадратною. Якщо $\mathbf{n} = \{\cos\alpha, \cos\beta\}$ — одиничний вектор нормалі до Г, то похідна за напрямком нормалі в точці B дорівнює

$$\frac{\partial u}{\partial \mathbf{n}}\Big|_B = \frac{\partial u}{\partial x}\cos\alpha + \frac{\partial u}{\partial y}\cos\beta.$$

Будемо вважати, що у вузлі B напрямок нормалі той же, що і в точці A:

$$\left(\frac{\partial u}{\partial \mathrm{n}}\right)_A = \left(\frac{\partial u}{\partial \mathrm{n}}\right)_B.$$

Похідні в напрямках x і y замінимо різницевими наближеннями:

$$\left.\frac{\partial u}{\partial \mathrm{n}}\right|_A = \frac{v_{m+1,n} - v_{m,n}}{h}\cos\alpha + \frac{v_{m,n+1} - v_{m,n}}{h}\sin\alpha. \qquad (12.18)$$

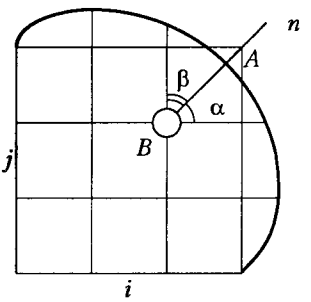


Рис. 12.5. Апроксимація граничної умови другого роду

Похибка виникає через заміну нормальної похідної різницевими співвідношеннями і через перенос нормалі в точку B. Похибка апроксимації в цьому випадку має перший порядок за h.

Такі рівняння (12.18) записуються для кожного граничного вузла. Отже, для знаходження невідомих значень $v_{m,n}$ у вузлах сітки одержимо систему лінійних алгебраїчних рівнянь, у якій кількість рівнянь дорівнює кількості невідомих. Відзначимо, що кількість рівнянь може бути досить великою. Так, для розв'язання задачі з високою точністю потрібно задати кількість вузлів M та N не менше 100, тому кількість рівнянь може сягати кількох тисяч. Наприклад, кожне рівняння (12.16) містить лише п'ять невідомих, але в системі їх близько N^2. Матриця цієї системи є сильно розрідженою. Методи розв'язання систем із розрідженими матрицями розглянуто в розділі 3.

12.2.2. Умови розв'язання різницевих рівнянь

Розглянемо задачу Діріхле для еліптичного рівняння загального вигляду

$$Lu = a\frac{\partial^2 u}{\partial x^2} + b\frac{\partial^2 u}{\partial y^2} + c\frac{\partial u}{\partial x} + d\frac{\partial u}{\partial y} + gu = f, \quad u|_\Gamma = \varphi(x, y), \qquad (12.19)$$

де a, b, c, d, g, f — функції незалежних змінних x, y, що визначені в обмеженій області G із границею Г. Припустимо, що ці функції неперервні в $G + \Gamma$, та $a > 0$, $b > 0$, $g \leqslant 0$.

Апроксимація рівняння (12.19) різницевою схемою (рис. 12.3) виконується досить просто. Дійсно, для других похідних використаємо співвідношення (12.13) і (12.14), а перші похідні апроксимуємо за допомогою того ж шаблону за симетричними формулами:

$$\frac{\partial u_{m,n}}{\partial x} = \frac{u_{m+1,n} - u_{m-1,n}}{2h} + O(h^2),$$
$$\frac{\partial u_{m,n}}{\partial y} = \frac{u_{m,n+1} - u_{m,n-1}}{2h} + O(h^2). \qquad (12.20)$$

Підставивши ці співвідношення у рівняння (12.19), отримаємо:

$$L_h v_{m,n} = a_{m,n}\frac{v_{m-1,n} - 2v_{m,n} + v_{m+1,n}}{h^2} + b_{m,n}\frac{v_{m,n-1} - 2v_{m,n} + v_{m,n+1}}{h^2} +$$
$$+ c_{m,n}\frac{v_{m+1,n} - v_{m-1,n}}{2h} + d_{m,n}\frac{v_{m,n+1} - v_{m,n-1}}{2h} + g_{m,n}v_{m,n} = f_{m,n}, \qquad (12.21)$$

що апроксимує рівняння (12.19) із другим порядком відносно h. Кількість рівнянь (12.21) дорівнює кількості невідомих, тобто кількості внутрішніх вузлів. Перепишемо цю систему в іншому вигляді, згрупувавши члени, що містять однакові невідомі:

$$L_h v_{m,n} = A_{m,n}v_{m-1,n} + B_{m,n}v_{m+1,n} + C_{m,n}v_{m,n-1} + D_{m,n}v_{m,n+1} - E_{m,n}v_{m,n} = f_{m,n}, \qquad (12.22)$$

де

$$A_{m,n} = \frac{a_{m,n}}{h^2} - \frac{c_{m,n}}{2h}, \quad B_{m,n} = \frac{a_{m,n}}{h^2} + \frac{c_{m,n}}{2h}, \quad C_{m,n} = \frac{b_{m,n}}{h^2} - \frac{d_{m,n}}{2h},$$
$$D_{m,n} = \frac{b_{m,n}}{h^2} + \frac{d_{m,n}}{2h}, \quad E_{m,n} = \frac{2a_{m,n}}{h^2} + \frac{2b_{m,n}}{h^2} - g_{m,n}.$$

Оскільки ми припускали, що a, b, c, d, g неперервні в області $G + \Gamma$, причому $a > 0$, $b > 0$, $g \leqslant 0$ в області $G + \Gamma$, то, коли h досить мале, коефіцієнти $A_{m,n}$, $B_{m,n}$, $C_{m,n}$, $D_{m,n}$, $E_{m,n}$ будуть додатними в усіх вузлах сіткової області. Якщо ця умова виконана, є справедливою наведена нижче теорема.

Теорема (принцип максимуму). Нехай $v_{m,n}$ — сіткова функція. Якщо для кожного внутрішнього вузла виконується умова $L_h v_{m,n} \geqslant 0$ ($L_h v_{m,n} \leqslant 0$), то у внутрішніх вузлах області G_h функція $v_{m,n}$ не може мати додатного максимуму (відповідно — від'ємного мінімуму). Винятком є випадок, коли $v_{m,n} \equiv const$.

Справді, нехай $v_{m,n} \neq const$, і у всіх внутрішніх вузлах має місце нерівність $L_h v_{m,n} \geqslant 0$. Припустимо, що $v_{m,n}$ досягає додатного максимуму M у деякому внутрішньому вузлі. Тоді існує такий внутрішній вузол (m_0, n_0), у якому $v_{m_0 n_0} = M$, і хоча б в одному сусідньому з ним вузлі значення $v_{m,n}$ менше M. У виразі $L_h v_{m_0 n_0}$ замінимо $v_{m,n}$ на M, тоді матимемо строгу нерівність:

$$M(A_{m_0,n_0} + B_{m_0,n_0} + C_{m_0,n_0} + D_{m_0,n_0} - E_{m_0,n_0}) > 0.$$

але

$$A_{m_0,n_0} + B_{m_0,n_0} + C_{m_0,n_0} + D_{m_0,n_0} - E_{m_0,n_0} = g_{m_0,n_0}.$$

Таким чином, $g_{m_0 n_0} > 0$, що неможливо. Отже, наше припущення було невірним, і перша частина твердження доведена. Другу частину доводять аналогічно.

Тепер можна показати, що система рівнянь (12.22) має єдиний розв'язок. Для цього досить довести, що відповідна однорідна система має тільки тривіальний розв'язок і всі значення в граничних вузлах дорівнюють нулю, а це відразу випливає з принципу максимуму. Так, якби розв'язок однорідної системи був відмінний від нуля хоча б в одному внутрішньому вузлі G_h, він мав би досягати найбільшого додатного значення у внутрішньому вузлі або найменшого від'ємного значення, що неможливо, оскільки всі $v_{m,n} \equiv 0$ в усіх граничних вузлах. Отже, однорідна система має лише тривіальний розв'язок, а неоднорідна система (12.22) має єдиний розв'язок.

Основною особливістю матриць систем виду (12.22) є розрідженість. Методи розв'язання таких систем розділяються на прямі та ітераційні. На даний час ефективних прямих методів розв'язання подібних систем рівнянь з великими значеннями N не існує (розділ 3). Для їх розв'язання, як правило, використовуються ітераційні методи. Застосування класичних методів лінійної алгебри недоцільне. Наприклад, під час виконання перетворень Гаусса чи LU-розкладання практично всі нульові елементи проміжних нульових діагоналей поступово стають ненульовими, тобто матриця втрачає властивість розрідженості. Ітераційні методи є простішими і менше залежать від виду матриці. З цієї причини для розв'язання двовимірних різницевих рівнянь часто використовувалися винятково ітераційні методи. Однак збіжність таких методів, як метод Якобі та метод Зейделя, дуже повільна. В даний час інтенсивно розвиваються і прямі методи.

Сьогодні все частіше застосовуються багатосіткові та неявні ітераційні методи, у яких розв'язок на кожній ітерації знаходять прямим методом. Хоча такі методи алгоритмічно складніші, ніж явні, їхньою безсумнівною перевагою є те, що вони швидше збігаються.

12.3. Ітераційні методи розв'язання

Методи розв'язання двовимірних різницевих крайових задач розглянемо на прикладі задачі Діріхле для рівняння Пуассона (12.9) із однорідними граничними умовами (12.10) в одиничному квадраті $G(0 < x, y < 1)$ із границею Г:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y), \quad (x, y) \in G,$$

$$u|_{\Gamma} = 0, \qquad (x, y) \in \Gamma.$$

Введемо в області G квадратну сітку з кроком h. Замінимо початкове диференціальне рівняння (12.9) системою різницевих рівнянь (12.16):

$$v_{m-1,n} + v_{m+1,n} - 4v_{m,n} + v_{m,n-1} + v_{m,n+1} = h^2 f_{m,n}, \quad m, n = 1, ..., M-1. \qquad (12.23)$$

У граничних вузлах значення шуканої функції відомі точно, тому значення сіткової функції в цих вузлах будуть $v_{m,0} = v_{m,M} = 0$, $v_{0,n} = v_{M,n} = 0$, $m, n = 0, ..., M$.

Матриці систем, які отримують за апроксимації багатовимірних задач математичної фізики, звичайно мають велику розмірність, характеризуються значною розрідженістю і великим розкидом власних чисел. Наприклад, на сітці, що має крок $h = 0,01$, порядок системи дорівнює приблизно 10000. Сильна розрідженість спричинюється тим, що кожне рівняння системи (12.16) містить не більше п'яти відмінних від нуля елементів. Отже, відношення кількості нульових елементів до загальної кількості її елементів не перевищує $5/(M-1) = O(h)$.

Власні числа матриці системи (12.16) такі, що відношення найменшого $\lambda_{\min}$ власного числа до найбільшого $\lambda_{\max}$ має вигляд

$$\xi = \frac{\lambda_{\min}}{\lambda_{\max}} = \operatorname{tg}^2 \frac{\pi h}{2}.$$

Відношення ξ є величиною другого порядку малості, якщо $h \to 0$, а саме

$$\xi = \frac{1}{\operatorname{cond} A} = \frac{\pi^2 h^2}{4} + O(h^4).$$

Наслідком малості величини ξ є погана обумовленість системи (12.16), тому явні ітераційні методи для цієї системи збігаються повільно.

Покажемо, як можна розв'язати систему (12.16) простими ітераційними методами, а саме методами Якобі та Зейделя, і переконаємося, що швидкість їх збіжності невелика.

Розглянемо задачу:

$$\begin{gathered} \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial x^2} = -2\pi^2 \sin \pi x \sin \pi y, \\ 0 < x < 1, \, 0 < y < 1, \\ u(0, y) = u(1, y), \quad u(x, 0) = u(x, 1) = 0. \end{gathered} \tag{12.24}$$

У граничних вузлах значення шуканої функції відомі точно (12.24). Результатом різницевої апроксимації розглянутої задачі на рівномірній сітці з однаковим кроком $h = 1/(M-1)$ по осях є така система алгебраїчних рівнянь:

$$\begin{cases} v_{m,0} = 0, \quad v_{m,M-1} = 0, & m = 0, ..., M-1, \\ v_{m-1,n} + v_{m+1,n} - 4v_{m,n} + v_{m,n-1} + v_{m,n+1} = & m, n = 1, ..., M-2, \\ \quad = -2h^2 \sin(\pi m h) \sin(\pi n h), & \\ v_{0,n} = 0, \quad v_{M-1,n} = 0, & n = 0, ..., M-1. \end{cases} \tag{12.25}$$

Розв'яжемо систему (12.25) спочатку методом Якобі, а потім методами Зейделя і верхньої релаксації.

12.3.1. Метод Якобі

Для знаходження чисельного розв'язку необхідно задати початкове наближення — значення сіткової функції на нульовій ітерації $v_{m,n}^{(0)}$. Приймемо ці значення рівними нулю. Запишемо формули методу Якобі:

$$
\begin{aligned}
&v_{m,n}^{(0)} = 0, \quad m, n = 0, ..., M-1,\\
&v_{m,0}^{(k+1)} = v_{m,M-1}^{(k+1)} = 0, \quad m = 0, ..., M-1,\\
&v_{0,n}^{(k+1)} = v_{M-1,n}^{(k+1)} = 0, \quad n = 0, ..., M-1,\\
&v_{m,n}^{(k+1)} = 0,25(v_{m-1,n}^{(k)} + v_{m+1,n}^{(k)} + v_{m,n-1}^{(k)} v_{m,n+1}^{(k)} + 2h^2 \sin(\pi m h)\sin(\pi n h)),\\
&m, n = 1, ..., M-2, \quad k = 0, 1, 2, ...
\end{aligned}
$$

Програма, що реалізує метод Якобі, і результати наведені нижче.

Приклад 12.1

Використаємо пакет Mathematica для розв'язання задачі методом Якобі:

```
In[]:= m = 17; k = 0;
       h = 1/(m - 1);
       eps = 0.001; epsl = 10;
       Array[u,{m,m},0]; Array[v,{m,m},0]; Array[f,{m,m},0];
       Do[ f[i,j] = N[2 Pi^2 h^2 Sin[Pi h i]Sin[Pi h j]];
           u[i,j] = 0, {i,0,m-1},{j,0,m-1} ];
          While[ epsl > eps, k = k + 1; epsl = 0;
            Do[ v[i,j] = 0.25 (u[i-1,j] + u[i+1,j] + u[i,j-1] + u[i,j+1] + f[i,j]);
                ep = Abs[v[i,j] - u[i,j]]; epsl = Max[ep,epsl], {i,1,m-2}, {j,1,m-2}];
             Do[ u[i,j] = v[i,j], {i,1,m-2}, {j,1,m-2} ] ];
             U = Array[u,{m,m},0]; em = 1 - u[8,8];
       Print["k = ", k, "tol = ", em]; ListPlot3D[U];
Out[]= k = 154 tol = 0.0473382
```

Графік розв'язку задачі методом Якобі зображено на рис. 12.6.

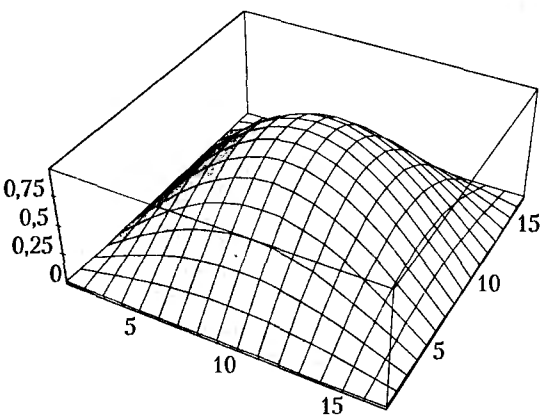


Рис. 12.6. Розв'язок задачі (12.1) методом Якобі, кількість ітерацій 154, максимальна похибка дорівнює 0,04734

12.3.2. Метод Зейделя

Метод Якобі має повільну збіжність, тому його рідко використовують для розв'язання подібних задач. У практиці частіше застосовують метод Лібмана, що являє собою окремий випадок методу Зейделя, розрахункові формули якого наведені нижче.

$$v_{m,n}^{(0)} = 0,$$

$$v_{m,0}^{(k+1)} = v_{m,M-1}^{(k+1)} = 0,$$

$$v_{0,n}^{(k+1)} = v_{M-1,n}^{(k+1)} = 0,$$

$$v_{m,n}^{(k+1)} = 0,25(v_{m-1,n}^{(k+1)} + v_{m+1,n}^{(k)} + v_{m,n-1}^{(k+1)} + v_{m,n+1}^{(k)} + 2h^2 \sin(\pi mh)\sin(\pi nh)),$$

$$m, n = 0, ..., M-1.$$

У цих формулах використовують значення, які обчислюються на $(k-1)$-му та $(k+1)$-му кроках. Швидкість збіжності методу Зейделя приблизно в два рази більша, ніж швидкість збіжності методу Якобі, що підтверджують практичні результати. Похибка розв'язку методом Зейделя, отримана за 96 ітерацій становить 0,021, а похибка, отримана методом Якобі за 154 ітерації — 0,047.

12.3.3. Метод верхньої релаксації

Метод верхньої релаксації дозволяє значно прискорити збіжність ітераційного процесу. Чергове наближення визначається за формулою

$$v_{m,n}^{(k+1)} = v_{m,n}^{(k)} + \omega(\tilde{v}_{m,n}^{(k+1)} - v_{m,n}^{(k)}),$$

де $\tilde{v}_{m,n}^{(k+1)}$ — чергове наближення, знайдене за формулою методу Зейделя, $1 < \omega < 2$ — ітераційний параметр. Розв'язання тієї ж самої задачі показало, що швидкість збіжності даного методу на порядок вище, ніж швидкість збіжності методу Зейделя. Похибка розв'язку після 30 ітерацій становить лише 0,0016.

Незважаючи на ефективність описаної процедури, не слід зловживати зменшенням кроку сітки, оскільки з ростом розмірності N обумовленість задачі знижується, через що неминучі похибки обчислення елементів наближення $v_{m,n}^{(k+1)}$ істотно зменшують точність результатів. Наприклад, для задачі з $N = 10^4$ число обумовленості має порядок $\approx 10^8$.

Тому рекомендуємо змінити крок сітки (наприклад, збільшивши вдвічі кількість її вузлів) і повторити розв'язання, результати виконання яких відповідно до екстраполяції Річардсона (див. підрозділ 1.4) дозволяють значно підвищити точність розв'язку, коли кількість вузлів сітки невелика. При цьому для симетричних формул апроксимації других похідних порядок похибки розв'язку зменшується з $O(h^2)$ до $O(h^4)$. Якщо ж використати три сітки з різною кількістю вузлів і застосувати повторну апроксимацію Річардсона, то порядок похибки можна знизити до $O(h^6)$.

12.3.4. Багатосітковий релаксаційний метод Федоренка

Альтернативою екстраполяції Річардсона є багатосітковий релаксаційний метод Федоренка, ефективність якого оцінимо для рівняння Пуассона

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y).$$

Відповідне різницеве рівняння має вигляд

$$\frac{(v_{m-1,n} + v_{m+1,n} - 4v_{m,n} + v_{m,n-1} + v_{m,n+1})}{h^2} = f_{m,n}, \qquad (12.26)$$
$$m, n = 0, 1, ..., N.$$

У цьому методі для зменшення норми початкової похибки в N разів необхідно виконати $O(N)$ ітерацій. Відомо, що найбільш швидкозбіжний метод Дугласа–Рекфорда (див. підрозділ 13.5.2) вимагає $O(N \ln N)$ ітерацій. Багатосітковий метод можна застосовувати для тих же рівнянь, що і метод простої ітерації.

Для розв'язання різницевих рівнянь (12.26) використовуємо обчислювальну схему

$$v_{m,n}^{(k+1)} = v_{m,n}^{(k)} + \tau\left[\frac{v_{m-1,n}^{(k)} - 2v_{m,n}^{(k)} + v_{m+1,n}^{(k)}}{h^2} + \frac{v_{m,n-1}^{(k)} - 2v_{m,n}^{(k)} + v_{m,n+1}^{(k)}}{h^2} - f_{m,n}\right]. \qquad (12.27)$$

Чисельний розв'язок задачі збігається, однак на дрібній сітці збіжність повільна. Після виконання k ітерацій за формулою (12.27) знайдемо поправку $\varepsilon_{m,n}$ розв'язку $v_{m,n}^{k}$, що задовольняє рівняння на сітці з удвічі більшим кроком:

$$\frac{\varepsilon_{m-1,n} + \varepsilon_{m+1,n} - 4\varepsilon_{m,n} + \varepsilon_{m,n-1} + \varepsilon_{m,n+1}}{(2h)^2} = r_{m,n}^{(k)}, \qquad (12.28)$$

де

$$r_{m,n}^{(k)} = \frac{v_{m-1,n}^{(k)} + v_{m+1,n}^{(k)} - 4v_{m,n}^{(k)} + v_{m,n-1}^{(k)} + v_{m,n+1}^{(k)}}{(2h)^2} - f_{m,n}$$

і являє собою нев'язку, що виникає в разі підстановки v_k у рівняння (12.26). Задача (12.28) розв'язується ітераційно за тими ж формулами, що і (12.27):

$$\varepsilon_{m,n}^{(k+1)} = \varepsilon_{m,n}^{(k)} + \theta\left[\frac{\varepsilon_{m-1,n}^{(k)} + \varepsilon_{m+1,n}^{(k)} - 4\varepsilon_{m,n}^{(k)} + \varepsilon_{m,n-1}^{(k)} + \varepsilon_{m,n+1}^{(k)}}{(2h)^2} - r_{m,n}^{(k)}\right],$$
$$\varepsilon_{m,n}^{0} = 0, \quad \theta = 4\tau.$$

Похибка швидко зменшується завдяки збільшенню кроку сітки. Однак збіжність, яка досягнута за рахунок цього і завдяки використанню формули (12.27), може виявитися недостатньою. Тому крок сітки доцільно ще збільшити вдвічі. Під час розв'язання задачі для кроку, збільшеного в чотири рази, знову слід використовувати процес подвоєння і т. д. Потім починається повернення до сітки з меншим кроком. Спочатку із самої великої сітки інтерполюємо отриману там останню поправку на вдвічі дрібнішу сітку, вносимо її в отриману до цього поправку на передостанній сітці і робимо кілька ітерацій, щоб вибрати внесену під час інтерполяції похибку. Результат цих ітерацій інтерполюємо на ще вдвічі дрібнішу сітку і т. д. На останньому кроці після інтерполяції та внесення поправки з попередньої сітки, виконавши кілька ітерацій, отримаємо наближене рішення.

12.4. Прямі методи

Якщо системи різницевих рівнянь, апроксимуючих еліптичну крайову задачу, мають невелику розмірність, їх можна розв'язати будь-яким прямим методом. Однак для розв'язання більшості практичних задач будують сітки, що мають десятки й сотні тисяч вузлів. У результаті отримуємо системи рівнянь, які мають таку ж кількість невідомих. У цих випадках необхідно використовувати більш ефективні та швидкодіючі методи, які враховують специфіку систем рівнянь (12.22).

Один із підходів до розв'язання системи різницевих рівнянь великої розмірності виду (12.16) прямими методами є підхід, що передбачає використання блочних матриць, за допомогою яких представляють матрицю системи. Такий підхід дозволяє скоротити обсяг обчислень, оскільки в цьому випадку розв'язання однієї великої системи рівнянь зводиться до розв'язання кількох систем меншої розмірності, значна частина яких має нульові матриці та праві частини. Щоб продемонструвати переваги цього підходу розглянемо різницеву схему (12.16), переписавши її в іншому вигляді:

$$\begin{aligned} &v_{m-1,n} - 2v_{m,n} + v_{m,n} + v_{m,n-1} - 2v_{m,n} + v_{m,n+1} = h^2 f_{m,n}, \\ &v_{0,n} = \varphi(0, y_n), \quad v_{m,n} = \varphi(a, y_n), \quad n = 1, ..., N-1, \\ &v_{m,0} = \varphi(x_m, 0), \quad v_{v,N} = \varphi(x_m, b), \quad m = 1, ..., M-1. \end{aligned} \tag{12.29}$$

Припустимо, що $M \gg N$ і введемо позначення

$$\mathbf{V}_m = [v_{m,1}, v_{m,2}, ..., v_{m,N-1}]^{\mathrm{T}}, \quad m = 0, ..., M. \tag{12.30}$$

Покладемо у формулах (12.29) $n = 1, 2, ..., M-1$ і, використовуючи умови (12.30), запишемо систему (12.29) у векторній формі:

$$\begin{aligned} \mathbf{V}_0 &= \Phi_0, \\ \mathbf{V}_{m-1} + \mathbf{A}\mathbf{V}_m + \mathbf{V}_{m+1} &= \mathbf{F}_m, \quad m = 1, ..., M-1, \\ \mathbf{V}_M &= \Phi_a, \end{aligned} \tag{12.31}$$

де $\mathbf{A}$ — тридіагональна матриця порядку $M-1$, яка має вигляд

$$\mathbf{A}=\begin{bmatrix} -4 & 1 & 0 & \dots & 0 & 0 & 0 \\ 1 & -4 & 1 & \dots & 0 & 0 & 0 \\ & & & \dots & & & \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ & & & \dots & & & \\ 0 & 0 & 0 & \dots & 1 & -4 & 1 \\ 0 & 0 & 0 & \dots & 0 & 1 & -4 \end{bmatrix}, \quad \mathbf{F}_m=\begin{bmatrix} h^2 f_{m,1}-\varphi(x_m,0) \\ h^2 f_{m,2} \\ \dots \\ h^2 f_{m,M-2} \\ h^2 f_{m,M-1}-\varphi(x_m,b) \end{bmatrix},$$

$$\mathbf{\Phi}_0=\begin{bmatrix} \varphi(0,y_1) \\ \varphi(0,y_2) \\ \dots \\ \varphi(0,y_{M-1}) \\ \varphi(0,y_M) \end{bmatrix}, \quad \mathbf{\Phi}_a=\begin{bmatrix} \varphi(a,y_1) \\ \varphi(a,y_2) \\ \dots \\ \varphi(a,y_{M-1}) \\ \varphi(a,y_M) \end{bmatrix}.$$

Матриця рівняння (12.31) є блоковою тридіагональною матрицею, що має такий вигляд:

$$\begin{bmatrix} \mathbf{A} & -\mathbf{E} & & \dots & & & \\ -\mathbf{E} & \mathbf{A} & & \dots & & & \\ & & & \dots & & & \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ & & & \dots & & & \\ & & & \dots & & \mathbf{A} & -\mathbf{E} \\ & & & \dots & 0 & -\mathbf{E} & \mathbf{A} \end{bmatrix} \mathbf{V}_m=\mathbf{F}_m,$$

де $\mathbf{E}$ — одинична матриця порядку $M-1$.

12.4.1. Метод матричного прогону

Розв'язок системи (12.31) будемо шукати у вигляді

$$\mathbf{V}_m=\mathbf{P}_{m+1}\mathbf{V}_{m+1}+\mathbf{Q}_{m+1}, \qquad (12.32)$$

де $\mathbf{P}_{m+1}$ — квадратна матриця, а $\mathbf{Q}_{m+1}$ — вектор (розмірність матриці та вектора $M-1$).

Підставивши $\mathbf{V}_{m-1} = \mathbf{P}_m\mathbf{V}_m + \mathbf{Q}_m$ у рівняння (12.31), одержимо рекурентні формули:

$$\mathbf{P}_{m+1} = -(\mathbf{A} + \mathbf{P}_m)^{-1}, \quad \mathbf{Q}_{m+1} = (\mathbf{A} + \mathbf{P}_m)^{-1}(\mathbf{F}_m - \mathbf{Q}_m), \quad m = 1, ..., M-1. \tag{12.33}$$

Початкові значення матриці $\mathbf{P}_1 = 0$ та вектора $\mathbf{Q}_1 = \Phi_0$ задаються відповідно до рівняння $\mathbf{V}_0 = \Phi_0$.

Після обчислення всіх коефіцієнтів $\mathbf{P}_m$, $\mathbf{Q}_m$ вектори $\mathbf{V}_m$ для $m = M-1, ..., 1$ визначаються послідовно з рівняння (12.32), починаючи з $\mathbf{V}_{M-1}$, тому що вектор $\mathbf{V}_M = \Phi_a$.

Розглянутий вище метод називається методом матричного прогону і являє собою узагальнення звичайного методу прогону для системи векторних рівнянь (12.31). На відміну від інших прямих методів цей метод є універсальним, оскільки дозволяє розв'язати рівняння зі змінними коефіцієнтами і не накладає сильних обмежень на граничні умови. Проте даний метод потребує великої кількості обчислень і ресурсів пам'яті, тому що під час його реалізації необхідно зберігати у пам'яті всі матриці $\mathbf{P}_m$, $\mathbf{Q}_m$, $m = 1, ..., M-1$.

Крім того, для обчислення розв'язку в кожному вузлі необхідно знайти обернену матрицю і двічі перемножити матриці порядку $N-1$, для чого треба виконати $O(N^3)$ арифметичних операцій. Отже, для обчислення всіх коефіцієнтів $\mathbf{P}_m$, $\mathbf{Q}_m$, $m = 1, ..., M-1$ потрібно загалом $O(N^3M)$ операцій. Для модельної задачі, коли $M = N = h^{-1}$, кількість операцій зростає до $O(h^4)$. Із цих причин метод порівняно рідко застосовують для розв'язання задач математичної фізики. Однак його можна рекомендувати для розв'язання систем з матрицями невеликої розмірності.

Щоб знайти розв'язок еліптичних крайових задач, застосовують також метод установлення (див. підрозділ 13.6), різницева реалізація якого дозволяє використовувати сучасні чисельні методи змінних напрямків.

12.5. Метод скінченних елементів

Метод скінченних елементів є альтернативою методу скінченних різниць. Основна ідея методу полягає в тому, що розв'язок диференціальних рівнянь із частинними похідними апроксимується дискретною моделлю, яку будують на множині кусково-неперервних функцій, визначених на скінченній кількості підобластей (елементів), на які розбивається область визначення.

Існує певна аналогія між методом скінченних елементів і методом Гальоркіна у разі наближення розв'язку крайової задачі за допомогою системи базисних функцій. Базисними функціями тепер є кусково-неперервні функції, що наближено апроксимують розв'язок задачі для кожного з виділених елементів. Загальний розв'язок формується об'єднанням часткових розв'язків із забезпеченням його неперервності в усій області визначення функції розв'язку.

Такий підхід, по-перше, більш зручний для розв'язання диференціальних рівнянь із частинними похідними зі складними граничними умовами, оскільки форму виділених елементів можна змінювати залежно від типу граничних умов (у методі скінченних різниць елементами сітки є прямокутники). Крім того, можна змінювати і розміри елементів, якщо в цьому є необхідність, із метою покращення апроксимації розв'язку за великих значень градієнтів у різних точках області визначення функції розв'язку. По-друге, метод дозволяє отримати розв'язок у кожній точці області визначення, а не тільки в ізольованих вузлах сітки.

Особливості реалізації методу скінченних елементів залежать від постановки задачі, але послідовність етапів методу не змінюється.

12.5.1. Дискретизація області визначення розв'язку

Дискретизація області полягає в поділі області визначення розв'язку на скінченну кількість підобластей (елементів), що мають певні розміри і форму [32]. Під час розв'язання задач математичної фізики методом скінченних елементів використовують елементи різних типів. Найпростішим елементом є одновимірний відрізок (рис. 12.7, *а*), який може мати поперечний переріз. Такий елемент має два вузли, по одному на кожному кінці.

Для побудови дискретної моделі двовимірної області використовуються два основних типи елементів: трикутники і чотирикутники (рис. 12.7, *б*), що можуть мати й криволінійні сторони. Для моделювання криволінійних границь у середину сторін цих елементів включаються додаткові вузли. Трикутні й прямокутні елементи можуть застосовувати одночасно. Із тривимірних елементів найчастіше використовують тетраедри і паралелепіпеди (рис. 12.7, *в*), грані яких можуть бути криволінійними поверхнями.

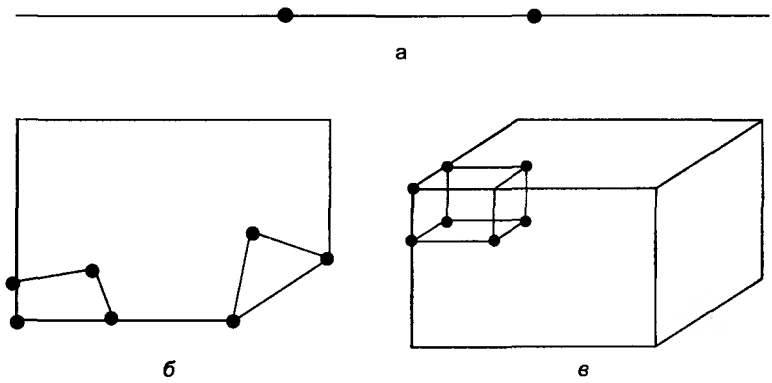


Рис. 12.7. Можливі конфігурації скінченних елементів:
а — одновимірні відрізки; *б* — трикутники і чотирикутники; *в* — паралелепіпед

Як правило, область розв'язку розбивається нерівномірно, в результаті чого всі елементи мають різну форму і розміри, які вибирають залежно від фізичних особливостей задачі, наприклад концентрації механічних напруг, температурних градієнтів і т. д. Можливість варіювати розміри і форму елементів — важлива властивість цього методу (рис. 12.8).

Слід зазначити, що під час дискретизації області розв'язку необхідно не лише виділяти елементи і вузли, але й нумерувати їх. Спосіб нумерації вузлів істотно впливає на ефективність обчислень.

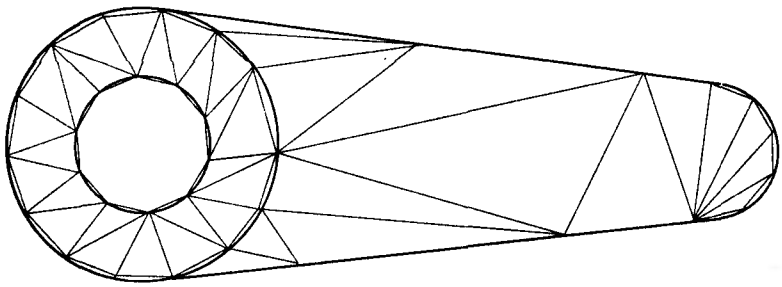


Рис. 12.8. Двовимірна дискретна модель механічної деталі

Метод скінченних елементів також зводить розв'язання крайових задач до розв'язання систем лінійних рівнянь великої розмірності з розрідженими стрічковими матрицями. Ширина стрічки (смуги) матриці визначається співвідношенням $B = (1 + K)M$, де M — кількість змінних у кожному вузлі, K — найбільша різниця між номерами вузлів у окремому напрямку. Мінімізація значення K, зокрема, досягається послідовною нумерацією вузлів у напрямку до області розв'язку найменшого розміру.

12.5.2. Кусково-неперервні функції елементів

Наступний етап полягає у пошуку наближеного розв'язку для кожного елемента. Для цього спочатку задається вид апроксимуючої кусково-неперервної функції з невідомими коефіцієнтами, що використовується для побудови розв'язку, а потім обчислюються значення невідомих коефіцієнтів за умови найкращого наближення цієї функції до розв'язку.

Апроксимуючими функціями найчастіше є поліноми, степінь яких визначається кількістю вузлів, зв'язаних з елементом. Для найпростішого одновимірного елемента такою функцією є

$$u(x) = a_0 + a_1 x, \tag{12.34}$$

яка повинна задовольняти граничні умови:

$$u(x_1) = u_1 = a_0 + a_1 x_1,$$
$$u(x_2) = u_2 = a_0 + a_1 x_2,$$

звідки

$$a_0 = \frac{u_1 x_2 - u_2 x_1}{x_2 - x_1}, \quad a_1 = \frac{x - x_1}{x_2 - x_1}. \tag{12.35}$$

Після підстановки виразів (12.35) у (12.34) отримаємо:

$$u(x) = N_1 u_1 + N_2 u_2, \tag{12.36}$$

де

$$N_1 = \frac{x_2 - x}{x_2 - x_1}, \quad N_2 = \frac{x - x_1}{x_2 - x_1}.$$

Формула (12.36) являє собою, власне кажучи, інтерполяційний поліном Лагранжа першого порядку, що задає значення функції на проміжку $[x_1, x_2]$.

Для двовимірного елемента (трикутника з вершинами 1, 2 і 3) лінійна функція може мати вигляд

$$u(x, y) = a_0 + a_1 x + a_2 y. \tag{12.37}$$

Використовуючи значення функції у вершинах трикутника, запишемо систему рівнянь для знаходження невідомих коефіцієнтів:

$$\begin{aligned} u(x_1, y_1) &= u_1 = a_0 + a_1 x_1 + a_2 y_1, \\ u(x_2, y_2) &= u_2 = a_0 + a_1 x_2 + a_2 y_2, \\ u(x_3, y_3) &= u_3 = a_0 + a_1 x_3 + a_2 y_3, \end{aligned}$$

котра в матричній формі має вигляд:

$$\begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \begin{vmatrix} a_0 \\ a_1 \\ a_2 \end{vmatrix} = \begin{vmatrix} u_1 \\ u_2 \\ u_3 \end{vmatrix}. \tag{12.38}$$

Розв'язання наведеної системи дозволяє знайти невідомі коефіцієнти:

$$\begin{aligned} a_0 &= \frac{1}{2A}[u_1(x_2 y_3 - x_3 y_2) + u_2(x_3 y_1 - x_1 y_3) + u_3(x_1 y_2 - x_2 y_1)], \\ a_1 &= \frac{1}{2A}[u_1(y_2 - y_3) + u_2(y_3 - y_1) + u_3(y_1 - y_2)], \\ a_2 &= \frac{1}{2A}[u_1(x_3 - x_2) + u_2(x_1 - x_3) + u_3(x_2 - x_1)], \\ A &= \frac{1}{2}[(x_2 y_3 - x_3 y_2) + (x_3 y_1 - x_1 y_3) + (x_1 y_2 - x_2 y_1)]. \end{aligned}$$

Після підстановки цих коефіцієнтів у вираз (12.37) отримаємо

$$u(x, y) = N_1 u_1 + N_2 u_2 + N_3 u_3, \tag{12.39}$$

де

$$N_1 = \frac{1}{2A}[(x_2y_3 - x_3y_2) + (y_2 - y_3)x + (x_3 - x_2)y],$$

$$N_2 = \frac{1}{2A}[(x_3y_1 - x_1y_3) + (y_3 - y_1)x + (x_1 - x_3)y],$$

$$N_3 = \frac{1}{2}[(x_1y_2 - x_2y_1) + (y_1 - y_2)x + (x_2 - x_1)y].$$

Функції елементів (12.36) і (12.39) можна диференціювати та інтегрувати. Наприклад, диференціюючи вираз (12.39), знаходимо значення градієнта за напрямком x:

$$\frac{\partial u(x, y)}{\partial x} = \frac{\partial N_1}{\partial x}u_1 + \frac{\partial N_2}{\partial x}u_2 + \frac{\partial N_3}{\partial x}u_3. \qquad (12.40)$$

Але оскільки

$$\frac{\partial N_1}{\partial x} = \frac{y_2 - y_3}{2A}, \quad \frac{\partial N_2}{\partial x} = \frac{y_3 - y_1}{2A}, \quad \frac{\partial N_3}{\partial x} = \frac{y_1 - y_2}{2A},$$

то градієнт (12.40) не залежить від заданих координат x_i, y_i і значень функції $u(x_i, y_i)$. Сталість градієнта всередині кожного елемента вказує на необхідність використання для апроксимації швидко змінюваних функцій елементів дуже малих розмірів.

Продемонструємо можливість інтегрування введених функцій на прикладі найпростішого лінійного скінченного елемента:

$$\int_{x_1}^{x_2} u(x)\,dx = \int_{x_1}^{x_2} (N_1u_1 + N_2u_2)\,dx.$$

З урахуванням (12.36)

$$\int_{x_1}^{x_2} N_1u_1\,dx = \int_{x_1}^{x_2} \frac{x_2 - x}{x_2 - x_1}\,dx = \frac{1}{2}(x_2 - x_1)u_1,$$

$$\int_{x_1}^{x_2} N_2u_2\,dx = \int_{x_1}^{x_2} \frac{x - x_1}{x_2 - x_1}\,dx = \frac{1}{2}(x_2 - x_1)u_2$$

отримаємо

$$\int_{x_1}^{x_2} u(x)\,dx = \frac{u_1 + u_2}{2}(x_2 - x_1),$$

що дорівнює площі трапеції з висотою $(x_2 - x_1)$ і основами u_2 і u_1.

Відмітимо, що змінний за значенням градієнт всередині елемента можна отримати вибором для двовимірного трикутника замість формули (12.37) нелінійної функції, наприклад полінома другого порядку:

$$u(x, y) = a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 xy + a_5 y^2.$$

Але в такому елементі для обчислення шести невідомих коефіцієнтів повинно бути шість вузлів для формування системи із шести рівнянь. Це істотно збільшує обсяг необхідних обчислень, виконання яких без комп'ютера (на відміну від методу скінченних різниць) вже неможливе.

Для тривимірного елемента (тетраедра з чотирма вершинами 1, 2, 3 і 4) вибирають найпростіший поліном типу

$$u(x, y, z) = a_0 + a_1 x + a_2 y + a_3 z, \tag{12.41}$$

невідомі коефіцієнти якого визначають з урахуванням чотирьох умов у вузлах тетраедра:

$$u(x_1, y_1, z_1) = u_1 = a_0 + a_1 x_1 + a_2 y_1 + a_3 z_1,$$
$$u(x_2, y_2, z_2) = u_2 = a_0 + a_1 x_2 + a_2 y_2 + a_3 z_2,$$
$$u(x_3, y_3, z_3) = u_3 = a_0 + a_1 x_3 + a_2 y_3 + a_3 z_3,$$
$$u(x_4, y_4, z_4) = u_4 = a_0 + a_1 x_4 + a_2 y_4 + a_3 z_4.$$

12.5.3. Знаходження наближених розв'язків методом скінченних елементів

Наближений розв'язок для кожного елемента можна обчислити з використанням початкового диференціального рівняння. Зокрема, з цією метою успішно застосовується метод Гальоркіна (див. розділ 11).

Нагадаємо основне положення цього методу: різниця між наближеним і точним розв'язками рівняння (нев'язка) повинна бути ортогональною до апроксимуючих функцій. Якщо виходити з диференціального рівняння $Lu - f = 0$, де L — диференціальний оператор, і наближений розв'язок шукати у вигляді $\tilde{u} = \sum_i N_i u_i$, то завдання полягає в мінімізації нев'язки $\varepsilon = L\tilde{u} - f$, тобто виконання таких умов:

$$\int_R \varepsilon N_i \, dR = 0, \quad i = 1, 2, ..., n, \tag{12.42}$$

де R — область визначення функції, що задається конфігурацією скінченного елемента, n — кількість вузлів скінченного елемента.

Найвищий порядок похідної, обумовлений диференціальним оператором L, не обмежений, але він перевищує порядок неперервності використовуваних

інтерполяційних поліномів на одиницю. У разі вибору для скінчених елементів поліномів першого порядку, розглянутих вище, функція u неперервна, але не її перша похідна, тому в цьому випадку в рівняння (12.42) можуть бути включені похідні лише першого порядку. Для подолання цього обмеження для обраного кусково-лінійного опису скінченного елемента слід зменшувати порядок рівняння, заданого оператором L, використанням процедури інтегрування частинами.

Подальший виклад прив'яжемо до найпростішого прикладу розв'язання одновимірного рівняння Пуассона:

$$\frac{d^2T}{dx^2} = -f(x), \tag{12.43}$$

яке описує розподіл температури в стрижні довжиною l із граничними умовами

$$T(0, t) = T_1, \quad T(l, t) = T_2. \tag{12.44}$$

Нев'язка наближення розв'язку цього рівняння на скінченному лінійному одновимірному елементі обчислюється за формулою

$$\varepsilon = \frac{d^2\tilde{T}}{dx^2} + f(x),$$

а рівняння (12.42) набуває такого вигляду:

$$\int_{x_1}^{x_2} \left[\frac{d^2\tilde{T}}{dx^2} + f(x)\right] N_i \, dx = 0, \quad i = 1, 2. \tag{12.45}$$

Розв'яжемо ці рівняння, застосовуючи метод інтегрування частинами для першого добутку під інтегралом:

$$\int_{x_1}^{x_2} \frac{d^2\tilde{T}}{dx^2} N_i \, dx = N_i \frac{d\tilde{T}}{dx}\bigg|_{x_1}^{x_2} - \int_{x_1}^{x_2} \frac{d\tilde{T}}{dx}\frac{dN_i}{dx} dx, \quad i = 1, 2. \tag{12.46}$$

Обчислимо спочатку окремі члени правої частини виразу (12.46) для $i = 1$:

$$N_1 \frac{d\tilde{T}}{dx}\bigg|_{x_1}^{x_2} = N_1(x_2)\frac{d\tilde{T}(x_2)}{dx} - N_1(x_1)\frac{d\tilde{T}(x_1)}{dx}.$$

Згідно з формулами (12.36) маємо $N_1(x_2) = 0$ і $N_1(x_1) = 1$, тому

$$N_1 \frac{d\tilde{T}}{dx}\bigg|_{x_1}^{x_2} = -\frac{d\tilde{T}(x_1)}{dx}. \tag{12.47}$$

Аналогічно для $i = 2$ отримаємо:

$$N_2 \frac{d\tilde{T}}{dx}\Big|_{x_1}^{x_2} = \frac{d\tilde{T}(x_2)}{dx}. \tag{12.48}$$

Другий член правої частини виразу (12.46) знаходимо підстановкою (12.46), (12.47) і (12.48) у вираз (12.45). Для $i = 1$ отримаємо:

$$\int_{x_1}^{x_2} \frac{d\tilde{T}}{dx}\frac{dN_1}{dx}dx = -\frac{dT(x_1)}{dx} + \int_{x_1}^{x_2} f(x)N_1(x)dx, \tag{12.49}$$

а для $i = 2$ маємо:

$$\int_{x_1}^{x_2} \frac{d\tilde{T}}{dx}\frac{dN_2}{dx}dx = \frac{dT(x_2)}{dx} + \int_{x_1}^{x_2} f(x)N_2(x)dx. \tag{12.50}$$

Відзначимо, що інтегрування частинами дозволяє ввести граничні умови безпосередньо в рівняння скінченного елемента.

Обчислимо праві частини виразів (12.49) і (12.50), підставляючи в них похідні від N_1, N_2 і $\tilde{T}$ (12.36):

$$\frac{dN_1}{dx} = -\frac{1}{x_2 - x_1}, \quad \frac{dN_2}{dx} = \frac{1}{x_2 - x_1},$$
$$\frac{du}{dx} = \frac{dN_1}{dx}u_1 + \frac{dN_2}{dx}u_2 = \frac{1}{x_2 - x_1}(-u_1 + u_2).$$

Одержимо для скінченного елемента, якщо $i = 1$ і $i = 2$, такі рівняння:

$$\int_{x_1}^{x_2} \frac{T_1 - T_2}{(x_2 - x_1)^2}dx = \frac{1}{x_2 - x_1}(T_1 - T_2), \tag{12.51}$$

$$\int_{x_1}^{x_2} \frac{-T_1 + T_2}{(x_2 - x_1)^2}dx = \frac{1}{x_2 - x_1}(-T_1 + T_2). \tag{12.52}$$

Рівняння (12.51) і (12.52) можна записати в матричній формі як рівняння обчислення наближеного розв'язку одновимірного рівняння Пуассона у вузлах скінченного лінійного елемента:

$$\frac{1}{x_2 - x_1}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \times \begin{vmatrix} T_2 \\ T_1 \end{vmatrix} = \begin{vmatrix} dT(x_1)/dx \\ dT(x_2)/dx \end{vmatrix} + \begin{vmatrix} \int_{x_1}^{x_2} f(x)N_1\,dx \\ \int_{x_1}^{x_2} f(x)N_2\,dx \end{vmatrix}. \tag{12.53}$$

Якщо позначити матрицю у лівій частині рівняння (12.53) через **k**, вектор невідомих значень температур у вузлах сітки через **u**, а вектор, що враховує зовнішні збурення елемента і граничні умови для нього через **F**, отримаємо загальний вигляд рівняння для скінченного елемента, що являтиме собою модель типу

$$\mathbf{Tu} = \mathbf{F}, \tag{12.54}$$

Ця модель може служити основою під час побудови спеціалізованої бібліотеки підпрограм для розв'язання диференціальних рівнянь із частинними похідними визначеного типу (параболічного, гіперболічного чи еліптичного) методом скінченних елементів.

12.5.4. Композиція загального розв'язку задачі

Дискретна модель для неперервної функції (розв'язку задачі) будується на *множині кусково-неперервних функцій, кожна з яких є чисельним розв'язком* рівняння для окремого елемента. Умова неперервності функції забезпечується рівністю значень функції у вузлах, що є спільними для сусідніх скінченних елементів. Якщо скінченні елементи задані поліномами другого порядку і вище, то умови неперервності розв'язку можуть включати також вимоги рівності похідних у вузлах сусідніх елементів.

Вимогу рівності значень функції у вузлах сусідніх елементів можна задати в матричній формі за умови узгодження глобальної нумерації вузлів всієї області розв'язку (координатного базису задачі) і локальної нумерації вузлів моделі рівнянь окремого елемента (координатного базису елемента), як це прийнято в теорії електронних схем у разі побудови матриці всієї схеми з моделей окремих компонентів. У результаті формується система лінійних рівнянь, що визначає вузлові значення в усій області розв'язку задачі:

$$\mathbf{Ku} = \mathbf{F}, \tag{12.55}$$

де **K** — стрічкова матриця жорсткості дискретної моделі задачі, **u** — вектор вузлових значень, **F** — узагальнений вектор зовнішніх збурень.

Оскільки деякі складові вектора **u** відомі з граничних умов, систему рівнянь (12.55) можна модифікувати, перенісши відомі величини з лівої частини рівняння у праву і виключивши їх із інших рівнянь системи. Надалі таку розріджену систему лінійних рівнянь розв'язують відомими методами. Наближений чисельний розв'язок диференціального рівняння з частинними похідними одержують у вигляді значень функції у вузлах.

Слід відмітити, що розв'язок рівняння, отриманий методом скінченних елементів, є єдиним і визначеним у всій області розв'язку, а не тільки у вузлах. Послідовність значень у вузлах є просто компактним зображенням функції розв'язку (а для двовимірної функції — кусково-планарною апроксимацією поверхні розв'язку). Метод скінченних елементів дозволяє досить точно описати складні криволінійні границі області визначення розв'язку і граничних умов.

Приклад 12.2

Обчислимо розподіл температури у стрижні довжиною $l = 8$ см, якщо температура на кінцях стрижня $T(0) = 20$, $T(8) = 100$ та існує рівномірне джерело тепла $f(x) = 10$. Теплові процеси у стрижні описуються рівнянням Пуассона (12.43):

$$\frac{d^2T}{dx^2} = -10.$$

Виділимо на стрижні чотири лінійних скінченних елементи довжиною $l_e = 2$ і задамо номери п'яти вузлів: 1, 2, 3, 4, 5.

Щоб скористатися рівняннями (12.53), необхідно обчислити вирази

$$\int_{x_1}^{x_2} f(x) N_1(x)\,dx = \int_0^2 10\frac{2-x}{2-0}\,dx = 10,$$

$$\int_{x_1}^{x_2} f(x) N_2(x)\,dx = \int_0^2 10\frac{x-0}{2-0}\,dx = 10.$$

Тоді рівняння (12.53) набувають такого вигляду:

$$\frac{1}{2}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \times \begin{vmatrix} T_2 \\ T_1 \end{vmatrix} = \begin{vmatrix} dT(x_1)/dx + 10 \\ dT(x_2)/dx + 10 \end{vmatrix}. \qquad (12.56)$$

Щоб записати рівняння для всіх чотирьох елементів, встановимо відповідність між локальними й глобальними номерами вузлів стрижня згідно з табл. 12.1.

Таблиця 12.1. Відповідність між локальними й глобальними номерами вузлів

Номер елемента	Локальна нумерація вузлів	Глобальна нумерація вузлів
1	1,2	1,2
2	1,2	2,3
3	1,2	3,4
4	1,2	4,5

Система рівнянь (12.55) для обчислення розв'язку в усіх вузлах створюється підсумовуванням рівнянь для всіх елементів і має такий вигляд:

$$\begin{array}{ccccc} T_1 & T_2 & T_3 & T_4 & T_5 \\ 0{,}5 & -0{,}5 & & & \\ -0{,}5 & 1 & -0{,}5 & & \\ & -0{,}5 & 1 & -0{,}5 & \\ & & -0{,}5 & 1 & -0{,}5 \\ & & & -0{,}5 & 0{,}5 \end{array} \times \begin{array}{c} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \end{array} = \begin{array}{r} -dt(x_1)/dx + 10 \\ 20 \\ 20 \\ 20 \\ -dT(x_5)/dx + 10 \end{array}$$

За послідовного внесення рівнянь скінченного елемента (12.56) у загальну систему граничні умови в суміжних вузлах 2, 3 і 4 взаємно компенсують одна одну. Це не виконується у першому і останньому вузлах (вузли 1 і 5).

Оскільки значення температури T_1 і T_5 задані (відомі з граничних умов), то отриману систему рівнянь можна перетворити у такий спосіб.

1. Члени лівої частини рівнянь, що містять T_1 і T_5, перенести в праву.
2. Невідомі з правої частини $(d(x_1)/dx, d(x_5)/dx)$, перенести в ліву.

У результаті отримаємо систему рівнянь з тридіагональною матрицею:

$$\begin{matrix} 1 & -0,5 & & & & dT(x_1)/dx & & 0 \\ & 1 & -0,5 & & & T_2 & & 30 \\ & -0,5 & 1 & -0,5 & \times & T_3 & = & 20 \\ & & -0,5 & 1 & & T_4 & & 70 \\ & & & -0,5 & -1 & dT(x_5)/dx & & -40 \end{matrix}.$$

Розв'язуючи цю систему рівнянь, отримаємо:

$$d(x_1)/dx = 50,$$
$$T_2 = 100, \quad T_3 = 140, \quad T_4 = 140,$$
$$d(x_5)/dx = -30.$$

Висновки

1. Розв'язання диференціальних рівнянь із частинними похідними є однією з найбільш трудомістких обчислювальних задач, що базуються на чисельних методах, розглянутих у попередніх розділах.
2. Існують два основних підходи до розв'язання диференціальних рівнянь із частинними похідними: метод скінченних різниць і метод скінченних елементів. Перший з них більше підходить для регулярних сіток. У разі використання симетричних формул апроксимації похідних він забезпечує другий порядок точності. Другий метод застосовується для розв'язання задач, в яких границя області визначення є досить складною (наприклад, під час проектування виробів зі складними поверхнями й у механіці рідких середовищ і газів). Цей метод за відповідного вибору базисних функцій забезпечує більш високу точність.
3. Метод скінченних різниць після підстановки різницевих апроксимацій для похідних зводить задачу розв'язання диференціальних рівнянь із частинними похідними до розв'язання еквівалентної системи розріджених лінійних рівнянь зі стрічковими матрицями, обумовленість яких погіршується зі збільшенням кількості вузлів сітки. Метод розв'язання еквівалентної системи рівнянь залежить від виду диференціального рівняння (параболічне, еліптичне або гіперболічне).
4. Еквівалентні системи лінійних алгебраїчних рівнянь для одновимірних і двовимірних еліптичних диференціальних рівнянь із частинними похідними зви-

чайно розв'язують прямими методами. Для забезпечення стійкості розв'язку необхідно підтримувати лінійну пропорційність між кроками змінювання змінних ($\Delta t \approx \Delta x$).

5. Еквівалентні системи лінійних алгебраїчних рівнянь для тривимірних еліптичних диференціальних рівнянь з частинними похідними звичайно розв'язують через їх велику розмірність ітераційними методами, наприклад методом Ліб-мана з урахуванням вказаних вище обмежень на вибір кроків для змінних.
6. Метод скінченних елементів, що базується на ідеях методів Гальоркіна (розділ 11), є більш складним, але має явні переваги порівняно з методом скінченних різниць, особливо коли область визначення функції розв'язку має складну форму. При цьому базисними функціями для апроксимації розв'язку можуть виступати кускові поліноми. Оскільки вони є фінітними, та їх вплив поширюється тільки в межах скінченного елемента довільної форми, матриці систем лінійних алгебраїчних рівнянь для обчислення вагових коефіцієнтів біля базисних функцій також мають стрічкову структуру.
7. Для прискорення розв'язання диференціальних рівнянь із частинними похідними, наприклад релаксаційним багатосітковим методом Федоренка, можна успішно застосовувати багатопроцесорні обчислювальні системи.

Контрольні запитання та завдання

1. Доведіть, що функція $u(x, y) = a\sin(x)\,\mathrm{sh}(y) + b\,\mathrm{sh}(x)\sin(y)$ є розв'язком рівняння Лапласа.
2. Доведіть, що функція $u(x, y) = a\sin(nx)\,\mathrm{sh}(ny) + b\,\mathrm{sh}(nx)\sin(ny)$ є розв'язком рівняння Лапласа для кожного цілого додатного n.
3. Нехай $u(x, y)$ має вигляд $u(x, y) = ax^2 + bxy + cy^2 + dx + fy + g$:
   а) знайдіть співвідношення між коефіцієнтами, які гарантують виконання рівності $u_{xx} + u_{yy} = 0$;
   б) знайдіть співвідношення між коефіцієнтами функції $u(x, y) = ax^2 + bxy + cy^2 + dx + fy + g$, які гарантують рівність $u_{xx} + u_{yy} = -1$;
   в) знайдіть коефіцієнти функції $u(x, y)$, заданої в п. *а*, які задовольняють рівняння Лапласа і граничні умови $u(x, 0) = 0$ та $u(x, l) = 0$;
   г) знайдіть коефіцієнти функції $u(x, y)$, заданої в п. *б*, які задовольняють рівняння з п. *б*, і граничні умови $u(x, 0) = 0$ та $u(x, l) = 0$.
4. Розв'яжіть рівняння

$$u_{xx} + u_{yy} = -4u$$

в області $G = \{(x, y) : 0 \leqslant x \leqslant 1, 0 \leqslant y \leqslant 1\}$ з граничними умовами $u(x, y) = \sin 2y + \cos x$.

5. Складіть систему різницевих рівнянь, що апроксимують на квадратній сітці з кроком h рівняння $u_{xx} + u_{yy} = xy$, задане в півколі з границею:

$$\begin{cases} x^2 + y^2 < 1, \\ y > 0 \end{cases}$$

для граничних умов:

а) $u(x,y)\big|_{(x,y)\in\Gamma} = |x + y| - 1;$

б) $\left[u - \dfrac{\partial u}{\partial n}\right]_{(x,y)\in\Gamma} = \begin{cases} y - \sin \pi x, & y > 0, \\ 0 & y = 0. \end{cases}$

Який порядок апроксимації побудованих різницевих схем? Знайдіть розв'язок отриманих різницевих рівнянь.

Розділ 13

Різницеві методи розв'язання мішаної задачі для параболічних рівнянь

- Стійкість і збіжність різницевих схем
- Спектральна ознака стійкості
- Різницеві схеми підвищеної точності
- Методи встановлення і прямих

Рівняння математичної фізики описують реальні фізичні процеси, тому їх розв'язок повинен бути єдиним і неперервно залежати від вихідних даних. Остання вимога обумовлена тим, що вихідні дані, як правило, визначаються під час експериментів, тому необхідно гарантувати незалежність розв'язку від похибок вимірювань.

Введення понять апроксимації, стійкості та збіжності дозволило створити необхідну базу для побудови ефективних різницевих схем розв'язування задач математичної фізики [8, 28, 30, 34].

У даному розділі розглядаються явні та неявні різницеві схеми розв'язування параболічних рівнянь та аналізуються умови їх стійкості.

13.1. Апроксимація, стійкість, збіжність різницевих схем. Основні поняття

У розділах 11 і 12 були введені поняття похибки апроксимації та стійкості різницевих схем. Тепер дамо їх формальне визначення. Спочатку розглянемо деяку задачу, визначену диференціальним рівнянням і граничними умовами, і запишемо її в операторній формі:

$$LU = F, \tag{13.1}$$

де L — деякий диференціальний оператор, що впливає на шукану функцію U, F — права частина. Вважатимемо, що оператор L включає як диференціальне рівняння, так і граничні умови.

Пояснимо введений оператор (13.1) на прикладі мішаної задачі для одновимірного рівняння теплопровідності.

Постановка цієї задачі була розглянута в підрозділі 12.1.3, а тут наведемо відповідні формули для неї:

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} + f(x,t), \quad 0 < x < l, 0 < t. \tag{13.2}$$

Оскільки рівняння містить похідну за часом, необхідно задавати початкові умови для $t = 0$:

$$u(x, 0) = \varphi(x), \quad 0 \leqslant x \leqslant l, t = 0 \tag{13.3}$$

і граничні умови для $x = 0, \ x = l, \ t > 0$.

Спочатку розглянемо граничні умови першого роду

$$\begin{aligned} u(0, t) &= \psi(t), \quad x = 0, t > 0, \\ u(l, t) &= \xi(t), \quad x = l, t > 0. \end{aligned} \tag{13.4}$$

У цьому випадку маємо першу мішану задачу, інакше задачу Коші з граничними умовами (рис. 13.1).

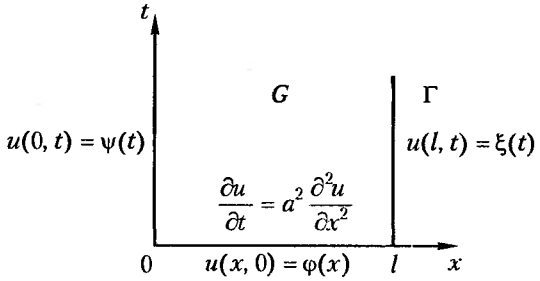


Рис. 13.1. Область визначення розв'язку одновимірного рівняння теплопровідності

Встановлюючи зв'язок мішаної задачі для одновимірного параболічного рівняння (13.2), (13.3) і (13.4) з операторним рівнянням (13.1), запишемо

$$LU = \begin{cases} \dfrac{\partial u}{\partial t} - a^2 \dfrac{\partial^2 u}{\partial x^2}, & \text{для} \quad t > 0, 0 < x < l, \\ u(x, 0), & \text{для} \quad t = 0, 0 \leqslant x \leqslant l, \\ u(0, t), & \text{для} \quad t > 0, x = 0, \\ u(l, t), & \text{для} \quad t > 0, x = l. \end{cases}$$

Відповідно

$$F = \begin{cases} f(x,t), & \text{для} \quad t > 0, 0 < x < l \\ \varphi(x), & \text{для} \quad t = 0, 0 \leqslant x \leqslant l, \\ \psi(t), & \text{для} \quad t > 0, x = 0, \\ \xi(t), & \text{для} \quad t > 0, x = l. \end{cases}$$

Для побудови сітки (рис. 13.2, *а*) виберемо крок $h = l/M$ по осі Ox і $\tau = T/K$ і проведемо лінії $x_m = const$ і $t_k = const$, так що $x_m = mh$, $m = 0, 1, ..., M$ і $t_k = k\tau$, $k = 0, 1, ..., K$. Тепер замінимо диференціальне рівняння (13.1) різницевим. Із цією метою виберемо шаблон різницевої схеми, зображений на рис. 13.2, *в*.

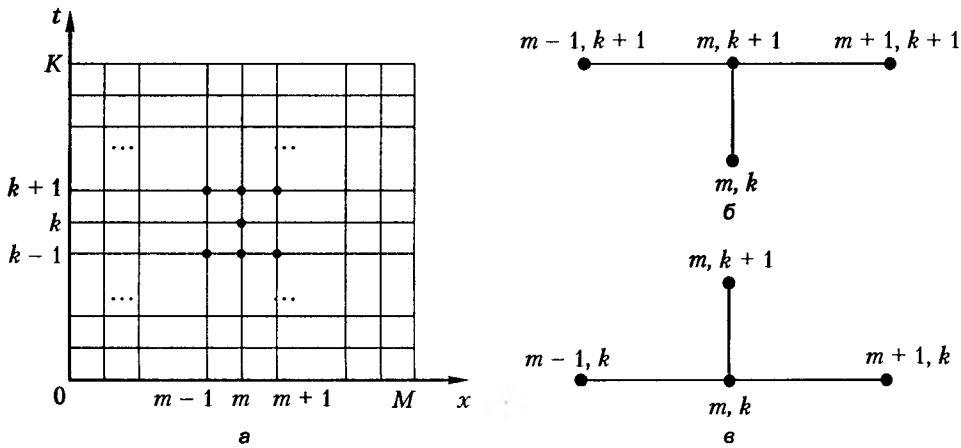


Рис. 13.2. Сіткова область і шаблони різницевої схеми для параболічного рівняння (*а*): *б* — шаблон неявної схеми; *в* — шаблон явної схеми

Введемо позначення для сіткової функції у вузлах v_m^k і на основі вибраного шаблону запишемо різницеве рівняння для задачі (13.2) у такий спосіб:

$$\frac{v_m^{k+1} - v_m^k}{\tau} = a^2 \frac{v_{m-1}^k - 2v_m^k + v_{m+1}^k}{h^2} + f_m^k, \quad m = 1, ..., M-1, k = 1, 2, ..., \qquad (13.5)$$

де $f_m^k = f(x_m, t_k)$.

Подібні рівняння запишемо для кожного внутрішнього вузла сітки. Початкові та граничні умови першого роду апроксимуються без похибки завдяки таким співвідношенням:

$$v_m^0 = \varphi(x_m) = \varphi_m, \quad m = 0, 1, ..., M, \qquad (13.6)$$

$$v_0^k = \psi(t_k) = \psi^k, \quad v_M^k = \xi(t_k) = \xi^k, \quad k = 1, 2, ... \qquad (13.7)$$

Тепер різницеву схему (13.5), (13.6) і (13.7) зобразимо в операторній формі:

$$L_h V^h = F_h, \qquad (13.8)$$

де $V^h = (v_0^k, v_1^k, ..., v_M^k)$, $k = 0, 1, ..., M$ — сукупність значень сіткової функції на k-му часовому шарі:

$$L_h V^h = \begin{cases} \dfrac{v_m^{k+1} - v_m^k}{\tau} - a^2 \dfrac{v_{m-1}^k - 2v_m^k + v_{m+1}^k}{h^2}, & m = 1, ..., M-1, k = 1, 2, ..., \\ v_m^0 = \varphi_m, \quad v_0^k = \psi^k, \quad v_M^k = \xi^k, & m = 0, ..., M, k = 1, ..., K-1. \end{cases} \qquad (13.9)$$

Порівнюючи вирази (13.9) і (13.8), побачимо, що L_hV^h визначається лівими частинами рівнянь (13.5), (13.6) і (13.7). Відповідно

$$F_h = \begin{cases} f_m^k - \text{у всіх внутрішніх вузлах сітки,} \\ \varphi_m - \text{у вузлах нижнього шару,} \\ \psi^k - \text{у вузлах лівой границі,} \\ \xi^k - \text{у вузлах правой границі.} \end{cases} \tag{13.10}$$

Похибкою наближеного (різницевого) розв'язку називається функція, яка визначена у вузлах сітки в такий спосіб:

$$\varepsilon^h = V^h - U^h, \quad \text{тобто} \quad \varepsilon_m^k = v_m^k - u(x_m, t_k), \\ m = 0, ..., M, \; k = 0, ..., K. \tag{13.11}$$

Оскільки функції, визначені на сітковій області, можуть інтерпретуватися як вектори, то вибираючи норму у відповідному векторному просторі, приймемо за похибку значення $\|\varepsilon^h\|$.

Визначення. Наближений розв'язок V^h збігається до точного U, якщо $\|\varepsilon^h\| \to 0$ і $h, \tau \to 0$.

У попередніх розділах ми використовували поняття «похибка апроксимації». Дамо його формальне (загальне) визначення. Після підстановки точного розв'язку U рівняння (13.1) у різницевий оператор L^h маємо

$$L_hU = F_h + R_h, \tag{13.12}$$

де R_h називається нев'язкою. Значення похибки апроксимації позначається як $\|R_h\|$.

Із виразу (13.12) видно, що в загальному випадку R_h є сукупністю нев'язок, які отримують у разі підстановки точного розв'язку в різницеві рівняння, у початкові та граничні умови.

В окремому випадку для розглянутої різницевої схеми (13.5), (13.6) і (13.7) нев'язка різницевого рівняння має такий вигляд:

$$r_m^k = a^2 \frac{u_{m-1}^k - 2u_m^k + u_{m+1}^k}{h^2} + f_m^k - \frac{u_m^{k+1} - u_m^k}{\tau}, \\ m = 1, ..., M-1, \; k = 1, 2, ..., \tag{13.13}$$

нев'язки початкової та граничної умов дорівнюють нулю:

$$r_m^0 = \varphi - u(x_m, 0), \quad m = 0, ..., M, \\ r_0^k = \psi^k - u(0, t_k) = 0, \quad r_M^k = \xi^k - u(l, t_k) = 0, \quad k = 1, ..., K.$$

Визначення. Метод (різницева схема) (13.8) апроксимує задачу (13.1) на її розв'язку, коли $\|R_h\| \to 0$ і $h, \tau \to 0$. Якщо при цьому норма нев'язки задовольняє умову

$$\|R_h\| \leqslant Ch^p \tag{13.14}$$

і константа C не залежить від h, має місце апроксимація з порядком p. Наприклад, різницевий оператор

$$L_h v_{m,n} = \frac{v_{m-1,n} + v_{m+1,n} - 4v_{m,n} + v_{m,n-1} + v_{m,n+1}}{h^2},$$

що апроксимує оператор Лапласа, має другий порядок точності (підрозділ 12.2).

Порядок апроксимації диференціального рівняння не завжди збігається з порядком точності різницевої схеми, тому що останній залежить як від схеми апроксимації диференціального рівняння, так і від схеми апроксимації граничних і початкових умов.

Проте не всі апроксимуючі схеми є стійкими. У зв'язку з цим важливим є поняття стійкості різницевої схеми. Помилки обчислень початкових і граничних умов, а також правих частин рівнянь через округлення величин, що використовуються під час обчислень, й інші причини можна розглядати як збурення початкових і граничних умов і правих частин рівнянь. Очевидно, що різницева крайова задача (чи мішана задача) буде коректною і стійкою, якщо малі зміни початкових і граничних умов та правих частин рівняння, які пов'язані з випадковими похибками, не зумовлюють суттєвих змін розв'язку різницевої задачі. Якщо це не так, різницева задача є нестійкою. Важливо відзначити, що в нестійких різницевих схемах сітка не стає стійкою внаслідок зменшення її кроку, оскільки будь-які малі збурення розв'язку з часом необмежено ростуть. У зв'язку з цим для лінійних систем можна сформулювати визначення стійкості.

Різницева схема (13.8) є стійкою, якщо для будь-якої функції F_h має місце єдиний розв'язок V^h, такий, що

$$\|V^h\| \leqslant C\|F_h\| \tag{13.15}$$

з константою C, яка не залежить від параметрів сітки.

Очевидно, що для будь-якої з векторних норм, застосовуваних для оцінювання величини $\|F_h\|$, справедлива нерівність

$$\|F_h\| \leqslant \|f_h\| + \|\varphi_h\| + \|\psi_h\| + \|\xi_h\|.$$

Отже, перевірка стійкості зводиться до з'ясування, за яких умов виконується нерівність, що випливає з визначення (13.14):

$$\|V^h\| \leqslant C(\|f_h\| + \|\varphi_h\| + \|\psi_h\| + \|\xi_h\|). \tag{13.16}$$

Запис умови стійкості у вигляді (13.16) деталізує уявлення про фактори, від яких залежить стійкість різницевої схеми: наявність у правій частині величини f_h означає стійкість за правими частинами різницевих рівнянь, φ_h —

за початковими умовами, ψ_h, ξ_h — за граничними. Зрозуміло, що стійкість за кожним із згаданих факторів, є в загальному випадку, необхідною умовою стійкості в цілому (в розумінні виконання нерівності (13.16) з усіма доданками в правій частині). Проте якщо вихідна задача являє собою задачу Коші для еволюційних рівнянь, то визначальними факторами є стійкість за правими частинами і початковими даними (крайових умов у задачі Коші немає). Для стаціонарної задачі (не залежної від часу) немає початкових даних, і стійкість визначається за правими частинами рівнянь і граничними умовами. Нижче на конкретних прикладах переконаємося, що перевірка нерівності (13.16) є надзвичайно актуальною під час аналізу різницевих схем із метою вибору конкретної схеми для розв'язання задачі. Іноді вдається встановити справедливість нерівності (13.16) без всяких припущень щодо параметрів сітки. У цьому випадку схема називається *абсолютно стійкою*. Якщо нерівність (13.16) задовольняється в припущенні про деяке співвідношення між кроками сітки, то це припущення є достатньою умовою стійкості.

Коли вдається встановити, що за деякого обмеження на сіткові параметри стійкість має місце лише за окремим фактором, то дане обмеження на крок сітки трактується як необхідна умова стійкості.

Тепер можна сформулювати умови, за яких чисельний розв'язок (13.8) збігається до точного.

Теорема. Нехай різницева задача $L_h V^h = F_h$ апроксимує початкову диференціальну задачу $LU = F$ на її розв'язку з порядком p і різницева задача є стійкою. Тоді наближений розв'язок V^h збігається до точного, і має місце оцінка

$$\|V^h - U\| \leqslant Ch^p, \tag{13.17}$$

де C — константа, що не залежить від p і вхідних даних F.

Доведення. V^h є розв'язком різницевої задачі (13.8), тобто $L_h V^h = F_h$, нев'язка якої має вигляд (13.12), тобто $L_h U = F_h + R_h$. Підставимо в останнє співвідношення значення $F_h = L_h V^h$, одержимо $L_h U - L_h V^h = R_h$ і перепишемо його у вигляді $L_h(U - V^h) = R_h$.

Величина $U - V^h$ згідно з (13.11) є похибкою ε^h розв'язку різницевої схеми. Звідси отримаємо формулу для похибки:

$$L_h \varepsilon^h = -R_h.$$

За умовою теореми ця різницева схема є стійкою, тому згідно з (13.15) можемо записати

$$\|\varepsilon^h\| \leqslant C_1 \|R_h\|.$$

Використовуємо той факт, що різницева схема апроксимує початкову задачу з порядком p, тобто $\|R_h\| \leqslant C_2 h^p$, одержимо тоді $\|\varepsilon^h\| C_1 C_2 h^p$, що і було потрібно довести, — збіжність наближеного розв'язку різницевої задачі до точного розв'язку початкової задачі з порядком p.

Доведена теорема має загальне значення, тому що вона визначає умови, достатні для збіжності не тільки задач із рівняннями, які містять частинні похідні, але і для задач зі звичайними диференціальними рівняннями.

13.2. Різницеві методи розв'язання мішаної задачі для одновимірного параболічного рівняння

13.2.1. Апроксимація одновимірного параболічного рівняння

Розглянемо методи чисельного розв'язання мішаної одновимірної задачі (13.2), (13.3), (13.4). Відомо, що для цієї задачі існує єдиний розв'язок $u(x, t)$. Нев'язка різницевого рівняння (13.5) на точному розв'язку диференціального рівняння (13.2) визначається формулою (13.13). Встановимо її порядок. Для цього підставимо у формулу (13.13) вирази для різниць (див. формулу 12.13):

$$\frac{u_{m-1}^k - 2u_m^k + u_{m+1}^k}{h^2} = \frac{\partial^2 u_m^k}{\partial x^2} + O(h^2),$$

$$\frac{u_m^{k+l} - u_m^k}{\tau} = \frac{\partial u_m^k}{\partial t} + O(\tau)$$

і одержимо вираз для нев'язки:

$$r_m^k = f_m^k + a^2 \frac{\partial^2 u_m^k}{\partial x^2} + O(h^2) - \frac{\partial u_m^k}{\partial t} + O(\tau).$$

Враховуючи, що $u(x, t)$ є розв'язком рівняння (13.2), отримаємо остаточно:

$$\max_{0<m<M,\ 1\leqslant k\leqslant K} |r_m^k| = \|r_h\| = O(\tau + h^2).$$

Оскільки перша похідна за часом апроксимується з першим порядком точності за τ, а друга похідна за змінною x — з другим порядком відносно h, то різницеве рівняння (13.5) апроксимує рівняння (13.1) на його розв'язку з порядком $O(\tau + h^2)$.

Співвідношення

$$v_m^0 = \varphi_m, \quad m = 0, ..., M, \tag{13.18}$$

$$v_0^k = \psi^k, \quad v_M^k = \xi^k, \quad k = 1, 2, ... \tag{13.19}$$

без похибки замінюють початкові та граничні умови першого роду.

Розглянемо різницеву схему апроксимації на основі шаблона (рис. 13.2, *б*). Для цього використаємо вирази для похідних, що входять у рівняння (13.1) (див. формулу 12.13):

$$\frac{\partial^2 u_m^{k+1}}{\partial x^2} = \frac{u_{m-1}^{k+1} - 2u_m^{k+1} + u_{m+1}^{k+1}}{h^2} + O(h^2),$$

$$\frac{\partial u_m^{k+1}}{\partial t} = \frac{u_m^{k+1} - u_m^k}{\tau} + O(\tau).$$

Замінивши похідні в рівнянні (13.1) їх апроксимаціями, одержимо різницеві рівняння:

$$\frac{v_m^{k+1} - v_m^k}{\tau} = a^2 \frac{v_{m-1}^{k+1} - 2v_m^{k+1} + v_{m+1}^{k+1}}{h^2} + f_m^{k+1}, \qquad (13.20)$$
$$m = 1, ..., M-1, \quad k = 1, 2, ...$$

Початкові та граничні умови записують у вигляді (13.18) і (13.19). Отже, обидві різницеві схеми апроксимації задачі (13.1), (13.2) і (13.3) мають однаковий порядок точності відносно h і τ.

13.2.2. Обчислювальні алгоритми

Розв'язавши рівняння (13.5) відносно v_m^{k+1}, одержимо:

$$v_m^{k+1} = \sigma^2 v_{m-1}^k + (1 - 2\sigma^2)v_m^k + \sigma^2 v_{m+1}^k + \tau f_m^k, \qquad (13.21)$$

де

$$\sigma^2 = \frac{a^2\tau}{h^2}.$$

Оскільки значення шуканої функції на нульовому часовому шарі задані початковими умовами (13.6), то значення на першому шарі v_m^1, $m = 1, ..., M-1$ можна знайти за формулою (13.21) для $k = 0$. Значення $v_0^1 = \psi^1$, $v_M^1 = \xi^1$ знаходять з граничних умов (13.7). Використовуючи знайдені значення v_m^1, $m = 0, ..., M$ на першому шарі, можна за тією ж формулою (13.21) знайти значення функції v_m^2, $m = 1, ..., M-1$ на другому шарі. Таким чином, розв'язок системи (13.5), (13.6) і (13.7) знаходять за формулою (13.21) явно, шар за шаром. Така різницева схема (13.5), (13.6) і (13.7) називається *явною*.

Розглянемо різницеву задачу (13.18), (13.19) і (13.20). Різницеве рівняння (13.20) запишемо в такому вигляді:

$$\sigma^2 v_{m-1}^{k+1} - (1 + 2\sigma^2)v_m^{k+1} + \sigma^2 v_{m+1}^{k+1} = -v_m^k - \tau f_m^{k+1}, \qquad (13.22)$$
$$m = 1, ..., M-1, \quad k = 1, 2, ...$$

На першому шарі для $k = 0$ рівняння (13.22) має вигляд:

$$\sigma^2 v_{m-1}^1 - (1 + 2\sigma^2)v_m^1 + \sigma^2 v_{m+1}^1 = -v_m^0 - \tau f_m^0. \qquad (13.23)$$

Значення $v_m^0 = \varphi_m$ задають, виходячи з початкових умов. Згідно (13.18) і (13.19) маємо

$$v_0^1 = \psi^1, \quad v_M^1 = \xi^1. \qquad (13.24)$$

Таким чином, щоб знайти v_m^1, $m = 1, ..., M-1$, треба розв'язати триточкове різницеве рівняння з граничними умовами першого роду. Така різницева схема (13.18),

(13.19) і (13.20) називається *неявною*. Розв'язок різницевої схеми (13.18), (13.19) і (13.20) знаходять шар за шаром методом прогону [37], оскільки на кожному шарі задовольняються умови стійкості методу прогону для системи рівнянь (13.22). При цьому кількість необхідних арифметичних операцій для знаходження розв'язку на одному шарі прямо пропорційна значенню M, тобто не більше, ніж у разі застосування явної формули (13.21).

13.2.3. Стійкість і збіжність явних схем

Розглянемо спочатку різницеву схему (13.5), (13.6) і (13.7). Припустимо, що задовольняється умова $1-2\sigma^2 \geqslant 0$. Знайдемо величину v_m^{k+1} із рівняння (13.10)

$$|v_m^{k+1}| \leqslant \sigma^2 \left(|v_{m-1}^k| + |v_{m+1}^k|\right) + (1-2\sigma^2)|v_m^k| + \tau|f_m^k| \leqslant$$
$$\leqslant 2 \max_{0\leqslant m\leqslant M} |v_m^k| + (1-2\sigma^2) \max_{0\leqslant m\leqslant M} |v_m^k| + \tau \max_{0<m<M} |f_m^k| \leqslant \max_{0\leqslant m\leqslant M} |v_m^k| + \tau \max_{0<m<M} |f_m^k|.$$

Остання нерівність справедлива для будь-якого $0 < m < M$, у тому числі й для такого m^*, за якого $|v_{m^*}^{k+1}|$ досягає максимуму, тобто $|v_{m^*}^{k+1}| = \max\limits_{0<m<M} |v_m^{k+1}|$. Тому для цього вузла останню нерівність можна записати у вигляді

$$\max_{0<m<M} |v_m^{k+1}| \leqslant \max_{0\leqslant m\leqslant M} |v_m^k| + \tau \max_{0<m<M} |f_m^k|$$

чи, підсилюючи нерівність за рахунок другого доданка в правій частині формули:

$$\max_{0<m<M} |v_m^{k+1}| \leqslant \max_{0\leqslant m\leqslant M} |v_m^k| + \tau \max_{0<m<M,\ 0<k\leqslant K} |f_m^k| \leqslant \max_{0\leqslant m\leqslant M} |v_m^k| + \tau\|f\|.$$

Тепер врахуємо граничні умови на $(k+1)$-му шарі:

$$\max_{0\leqslant m\leqslant M} |v_m^{k+1}| \leqslant \max\left\{ \max_{0\leqslant m\leqslant M} |v_m^k| + \tau\|f\|, |\psi^{k+1}|, |\xi^{k+1}| \right\}. \qquad (13.25)$$

Нерівність (13.25) називається принципом максимуму для різницевої схеми (13.5), (13.6) і (13.7). Враховуючи рекурентність нерівності (13.25) за m, можна переписати її у вигляді:

$$\max_m |v_m^{k+1}| \leqslant \max\left\{ \max_{0\leqslant m\leqslant M} |v_m^{k-1}| + 2\tau\|f_h\|, \max\left\{|\psi^k|, |\psi^{k+1}|, |\xi^k|, |\xi^{k+1}|\right\}\right\} \leqslant \ldots$$
$$\ldots \leqslant \max\left\{ \max_{0\leqslant m\leqslant M} |v_m^0| + (K+1)\tau\|f_h\|, \max\left\{ \max_{0\leqslant k\leqslant K} |\psi^{k+1}|, \max_{0\leqslant k\leqslant K} |\xi^{k+1}| \right\}\right\}.$$

З огляду на довільність значення k із останньої нерівності випливає, що

$$\max_m |v_m^{k+1}| \leqslant \max\left\{ \max_{0\leqslant m\leqslant M} |v_m^0| + (k+1)\tau\|f_h\|, \max\left\{\|\psi^{(h)}\|, \|\xi^{(h)}\|\right\}\right\}.$$

Враховуючи початкові умови

$$\max_{0\leqslant m\leqslant M}\left|v_m^0\right| = \max_{0\leqslant m\leqslant M}\left|\varphi^m\right| = \|\varphi_h\|,$$

отримаємо

$$\|V^h\| \leqslant \max\{\|\varphi_h\| + T\|f_h\|, \|\psi_h\|, \|\xi_h\|\} \leqslant$$
$$\leqslant T\|f_h\| + \|\varphi_h\| + \|\psi_h\| + \|\xi_h\|$$

і остаточно

$$\|V^h\| \leqslant C(\|f_h\| + \|\varphi_h\| + \|\psi_h\| + \|\xi_h\|),$$

де $C = \max(1, T)$.

Отже, отримано нерівність (13.16) для розглянутої схеми, тобто доведено достатність умови $1-2\sigma^2 \geqslant 0$ для стійкості явної схеми (13.5), (13.6) і (13.7).

Таким чином, явна різницева схема (13.5), (13.6) і (13.7) апроксимує початкову задачу (13.2), (13.3) і (13.4) на її розв'язку з порядком $O(\tau + h^2)$ і є стійкою за умови $1-2\sigma^2 \geqslant 0$.

Із теореми (див. с. 400) одержимо, що за умови $1-2\sigma^2 \geqslant 0$ наближений розв'язок крайової задачі v_m^k за явною різницевою схемою (13.5), (13.6) і (13.7) збігається для $\tau \to 0$ і $h \to 0$ до точного $u(x_m, t_k)$ з першим порядком за τ і другим порядком за h.

Явну схему (13.5), (13.6) і (13.7) можна застосовувати лише за умови (13.18), тобто, коли $\tau \leqslant h^2/(2a^2)$. Це означає, що крок τ треба брати досить малим. Різницева схема, стійкість якої зв'язана з деякими обмеженнями на крок, називається умовно стійкою.

Явна схема є більш простою і потребує меншої кількості операцій для підрахунку значень наближеного розв'язку на одному часовому шарі. Її застосування обмежується тим, що явна різницева схема є стійкою лише для

$$\tau \leqslant \frac{h^2}{2a^2}.$$

Умова стійкості накладає занадто сильне обмеження на крок за часом. Дійсно, нехай, наприклад, $h = 10^{-2}$. Тоді крок τ не повинен перевищувати $0{,}5\cdot 10^{-4}/a^2$, якщо $a^2 = 10^2$, $\tau \leqslant 0{,}5\cdot 10^{-10}$. Для того щоб обчислити розв'язок v_m^1 за $t = 1$, треба взяти кількість кроків $n = \tau^{-1} = 0{,}5\cdot 10^6$, а для всіх m провести $0{,}5\cdot 10^8$ обчислень за формулою (10.5). Тому явні схеми для рівнянь параболічного типу використовують рідко. Далі буде показано, що багато неявних схем позбавлені цього недоліку.

13.2.4. Стійкість і збіжність неявних схем

Покажемо, за яких умов неявна різницева схема є збіжною. Рівняння похибки неявної різницевої схеми має такий вигляд:

$$\varepsilon_m^k = v_m^k - u(x_m, t_k).$$

Підставивши $v_m^k = \varepsilon_m^k + u(x_m, t_k)$ у рівняння (13.9), отримаємо:

$$\frac{\varepsilon_m^{k+1} - \varepsilon_m^k}{\tau} - a^2 \frac{\varepsilon_{m-1}^{k+1} - 2\varepsilon_m^{k+1} + \varepsilon_{m+1}^{k+1}}{h^2} =$$
$$= f_m^{k+1} - \frac{u_m^{k+1} - u_m^k}{\tau} + a^2 \frac{u_{m-1}^{k+1} - 2u_m^{k+1} + u_{m+1}^{k+1}}{h^2}, \qquad (13.26)$$
$$m = 1, ..., M-1.$$

Вираз

$$r_m^{k+1} = f_m^k - \frac{u_m^{k+1} - u_m^k}{\tau} + a^2 \frac{u_{m-1}^{k+1} - 2u_m^{k+1} + u_{m+1}^{k+1}}{h^2}$$

визначає нев'язку рівняння (13.9) для розв'язку диференціального рівняння (13.2). Розв'язавши рівняння (13.26) відносно ε_m^{k+1}, отримаємо

$$(1 + 2\sigma^2)\varepsilon_m^{k+1} = \sigma^2 \varepsilon_{m-1}^{k+1} + \sigma^2 \varepsilon_{m+1}^{k+1} + \varepsilon_m^k + \tau r_m^{k+1}, \qquad (13.27)$$
$$m = 1, ..., M-1.$$

Для початкових і граничних умов маємо

$$\varepsilon_0^{k+1} = 0, \quad \varepsilon_M^{k+1} = 0, \quad \varepsilon_m^0 = 0, \qquad (13.28)$$
$$m = 0, ..., M.$$

Нехай у вузлі m_0 похибка приймає найбільше за абсолютною величиною значення серед всіх абсолютних значень похибок на $(k+1)$-му часовому шарі, тобто

$$\max_m |\varepsilon_m^{k+1}| = |\varepsilon_{m_0}^{k+1}|.$$

Якщо такого внутрішнього вузла немає, то похибки в усіх вузлах $(k+1)$-го шару дорівнюють нулю і, отже, внаслідок довільності k наближений розв'язок збігається з точним. Якщо такий вузол є, то запишемо рівняння (13.27) для вузла $(m_0, k+1)$:

$$(1 + 2\sigma^2)\varepsilon_{m_0}^{k+1} = \sigma^2 \varepsilon_{m_0-1}^{k+1} + \sigma^2 \varepsilon_{m_0+1}^{k+1} + \varepsilon_{m_0}^k + \tau r_{m_0}^{k+1}.$$

Візьмемо модуль лівої та правої частин останнього рівняння:

$$(1 + 2\sigma^2)|\varepsilon_{m_0}^{k+1}| \leqslant \sigma^2 |\varepsilon_{m_0-1}^{k+1}| + \sigma^2 |\varepsilon_{m_0+1}^{k+1}| + |\varepsilon_{m_0}^k| + \tau |r_{m_0}^{k+1}|.$$

Ми лише підсилимо останню нерівність, якщо в правій частині замінимо

$$|\varepsilon_{m_0-1}^{k+1}| \quad \text{і} \quad |\varepsilon_{m_0+1}^{k+1}| \quad \text{на} \quad |\varepsilon_{m_0}^{k+1}|.$$

Одержимо тоді

$$|\varepsilon_{m_0}^{k+1}| \leqslant |\varepsilon_{m_0}^k| + \tau |r_{m_0}^{k+1}|. \qquad (13.29)$$

Позначимо через ε^k множину значень сіткової функції на шарі k, тобто

$$\varepsilon^k = (\varepsilon_0^k, \varepsilon_1^k, ..., \varepsilon_M^k), \quad k = 1, 2, ..., K,$$

і визначимо норму

$$\|\varepsilon^k\| = \max_m |\varepsilon_m^k| \quad \text{і} \quad \|r^k\| = \max_m |r_m^k|.$$

Підставивши $\|\varepsilon^k\|$ й $\|r^k\|$ у вираз (13.29), знову підсилимо нерівність і одержимо замість (13.29) нерівність

$$\|\varepsilon^{k+1}\| \leqslant \|\varepsilon^k\| + \tau\|r^{k+1}\|. \tag{13.30}$$

Записавши (13.30) для $k = 0, 1, ..., K-1$, отримаємо:

$$\begin{gathered}
\|\varepsilon^{k+1}\| \leqslant \|\varepsilon^k\| + \tau\|r^{k+1}\|, \\
\|\varepsilon^k\| \leqslant \|\varepsilon^{k-1}\| + \tau\|r^k\|, \\
\ldots \quad \ldots \quad \ldots \\
\|\varepsilon^1\| \leqslant \|\varepsilon^0\| + \tau\|r^1\|
\end{gathered}$$

і, підсумувавши ці вирази, одержимо

$$\|\varepsilon^{k+1}\| \leqslant \|\varepsilon^0\| + K\tau \max_{1 \leqslant j \leqslant k+1} \|r^j\|.$$

Якщо $\|\varepsilon^0\| = 0$, враховуючи значення порядку апроксимації неявної різницевої схеми, отримаємо остаточно для будь-якого $k = 1, ..., K-1$ оцінку

$$\|\varepsilon^{k+1}\| \leqslant T, \quad O(\tau + h^2), \tag{13.31}$$

з якої випливає збіжність розв'язку неявного методу (13.20) розв'язання одновимірної параболічної задачі, якщо $\tau \to 0$ і $h \to 0$, до точного $u(x_m, t_k)$ із першим порядком за τ і другим порядком за h. Неявна схема має абсолютну стійкість.

Такі ж викладки (пов'язані з використанням принципу максимуму) іноді забезпечують успіх під час дослідження стійкості та збіжності деяких різницевих схем. У наступному розділі ми одержимо більш простий універсальний метод, що дозволяє сформулювати необхідну умову стійкості різницевих схем для еволюційних задач.

13.3. Спектральна ознака стійкості

Для багатьох задач необхідні умови стійкості можна отримати досить легко. Такі умови дозволяють відразу виключити ті схеми, для яких вони не виконуються, і не аналізувати стійкість цих схем (оскільки у випадку порушення необхідної умови стійкості бути не може).

Виявляється, за деяких додаткових припущень розв'язок відповідних різницевих рівнянь може бути отриманий в явному вигляді. Аналізуючи поведінку цих розв'язків, отримують умови, які трактуються як необхідні у разі поширення їх на схеми загального вигляду. Сформулюємо згадані припущення. Під час дослідження стійкості різницевих схем вважатимемо праві частини рівнянь і граничні умови рівними нулю, тобто будемо розглядати схеми, що апроксимують задачу Коші для однорідних диференціальних рівнянь. Крім того, будемо вважати коефіцієнти цих рівнянь сталими.

За цих умов різницеві рівняння мають частинні гармонічні розв'язки:

$$v_m^k = \lambda^k e^{im\alpha}, \tag{13.32}$$

де α — довільне дійсне число, а значення λ необхідно знайти для кожної конкретної схеми.

Надалі в усіх прикладах будемо перевіряти існування розв'язків різницевих схем (13.32). Внаслідок спрощень вхідних даних для задачі побудови схеми є лише початкові умови. Умова стійкості за початковими даними для розв'язків (13.32) зводиться до вимоги обмеженості амплітуди цих гармонік, тобто

$$|v_m^k| = |\lambda^k| \leqslant const.$$

Остання нерівність буде виконуватися для $k \to \infty$, якщо

$$|\lambda| \leqslant 1. \tag{13.33}$$

Нерівність (13.33) називається умовою Неймана стійкості різницевих схем для еволюційних задач.

Далі ми побачимо, що для кожної схеми існує певна залежність амплітуди від параметрів сітки і параметра α: $\lambda = \lambda(\tau, h, \alpha)$. Вимагаючи виконання нерівності (13.33) для довільного α (тобто для довільної гармоніки), знаходимо необхідну умову стійкості конкретної схеми у вигляді деякого обмеження на кроки сітки τ та h. Її аналіз показує, що здебільшого вона є головною для визначення стійкості різницевих схем, які апроксимують еволюційні задачі.

Тепер перейдемо до дослідження конкретних різницевих схем.

Приклад 13.1

Перевіримо стійкість явної схеми (13.5), яка після спрощень, прийнятих у методі гармонік, має такий вигляд:

$$\frac{v_m^{k+1} - v_m^k}{\tau} = a^2 \frac{v_{m-1}^k - 2v_m^k + v_{m+1}^k}{h^2}, \quad m = 1, \ldots, M-1, \; k = 1, 2, \ldots,$$
$$v_m^0 = \varphi_m, \quad m = 0, \ldots, M. \tag{13.34}$$

Будемо шукати розв'язок рівняння (13.34) у вигляді $v_m^k = \lambda^k e^{im\alpha}$. Підставивши його в рівняння (13.34), одержимо:

$$\frac{\lambda^{k+1} e^{im\alpha} - \lambda^k e^{im\alpha}}{\tau} = a^2 \frac{\lambda^k e^{i(m-1)\alpha} - 2\lambda^k e^{im\alpha} + \lambda^k e^{i(m+1)\alpha}}{h^2}.$$

Скоротимо обидві частини рівняння на $\lambda^k e^{im\alpha}$:

$$\frac{\lambda - 1}{\tau} = a^2 \frac{e^{-i\alpha} - 2 + e^{i\alpha}}{h^2}$$

і знайдемо λ:

$$\lambda = 1 + \sigma^2(e^{-i\alpha} - 2\lambda + e^{i\alpha}), \quad \text{де} \quad \sigma^2 = a^2\tau/h^2.$$

Скориставшись тригонометричними формулами, отримаємо:

$$\lambda = 1 - 2\sigma^2(1 - \cos\alpha) = 1 - 4\sigma^2 \sin^2\alpha.$$

Використаємо умови Неймана:

$$-1 \leqslant 1 - 4\sigma^2 \sin^2\alpha \leqslant 1 \quad \text{або} \quad -1 \leqslant 1 - 4\sigma^2 \leqslant 1$$

і остаточно матимемо

$$\sigma^2 \leqslant \frac{1}{2}, \text{ тобто } \tau \leqslant \frac{h^2}{2a^2}. \tag{13.35}$$

Отже, нами одержано той же результат, що і в підрозділі 13.2.3.

Приклад 13.2

Розглянемо стійкість неявної різницевої схеми, що апроксимує параболічне рівняння зі змінним коефіцієнтом

$$\frac{\partial u}{\partial t} = \vartheta(x,t) \frac{\partial^2 u}{\partial x^2} + f(x,t).$$

Неявна різницева схема для цього рівняння має вигляд:

$$\frac{v_m^{k+1} - v_m^k}{\tau} = \vartheta_m^{k+1} \frac{v_{m-1}^{k+1} - 2v_m^{k+1} + v_{m+1}^{k+1}}{h^2} + f_m^{k+1}, \quad m = 1, ..., M-1, \quad k = 1, 2, ... \tag{13.36}$$

Для рівнянь зі змінними коефіцієнтами метод гармонік застосовується в припущенні, що коефіцієнти «заморожені» (постійні). Потім в остаточній умові коефіцієнти «розморожуються». Стосовно розглянутої вище явної схеми останнє означає перехід від умови (13.35) до

$$\sigma^2 = \max_{t,x}[\vartheta(x,t)]\tau/h^2 \leqslant \frac{1}{2}.$$

Приймаючи, що $f_m^{k+1} = 0$ в (13.36), і підставляючи в це рівняння $v_m^k = \lambda^k e^{im\alpha}$, одержимо після скорочень:

$$\frac{\lambda - 1}{\tau} = \vartheta \frac{(e^{-i\alpha} - 2 + e^{i\alpha})\lambda}{h^2}.$$

Виконавши перетворення, аналогічні перетворенням, проведеним у прикладі 13.1, одержимо:

$$\lambda = \frac{1}{1 + \frac{4\vartheta\tau}{h^2}\sin^2 \alpha/2}.$$

Очевидно, що завжди $|\lambda| = \lambda \leqslant 1$, незалежно від величини кроків сітки і значень $\vartheta(x, t)$, тому схема (13.36) є абсолютно стійкою.

13.4. Різницеві схеми підвищеної точності

Розглянемо кілька різницевих схем для одновимірного параболічного рівняння (13.2), що часто використовується на практиці. Ефективність цих схем було багато разів підтверджено.

13.4.1. Схема Кранка–Ніколсона

Схема будується за шеститочковим шаблоном, показаним на рис. 13.3, *а*, і має вигляд:

$$\frac{v_m^{k+1} - v_m^k}{\tau} = a^2 \left(\frac{v_{m-1}^k - 2v_m^k + v_{m+1}^k}{2h^2} + \frac{v_{m-1}^{k+1} - 2v_m^{k+1} + v_{m+1}^{k+1}}{2h^2} \right) + f_m^{k+\frac{1}{2}}. \qquad (13.37)$$

Встановимо її порядок апроксимації. Запишемо нев'язку

$$r_m^{k+\frac{1}{2}} = a^2 \frac{u_{m-1}^k - 2u_m^k + u_{m+1}^k + u_{m-1}^{k+1} - 2u_m^{k+1} + u_{m+1}^{k+1}}{2h^2} - \frac{u_m^{k+1} - u_m^k}{\tau} + f_m^{k+\frac{1}{2}}.$$

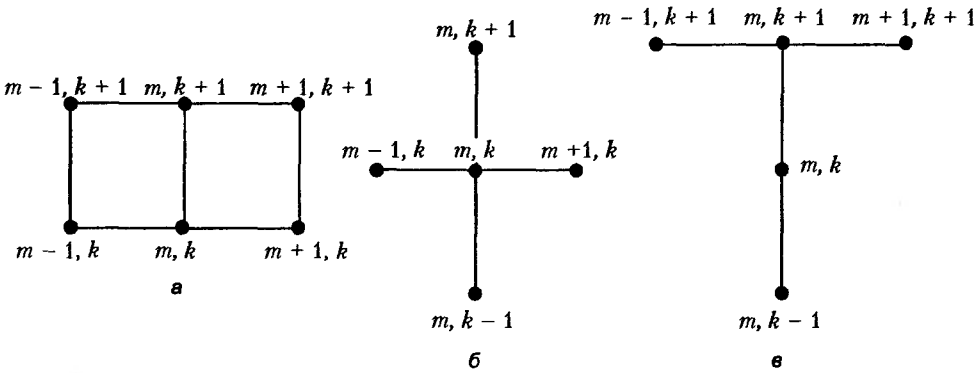


Рис.13.3. Різницеві схеми підвищеної точності: а — шеститочковий шаблон, *б* — чотириточковий тришаровий шаблон «хрест»; *в* — п'ятиточковий тришаровий шаблон

Скориставшись виразами (12.13)

$$\frac{u_{m-1}^k - 2u_m^k + u_{m+1}^k}{h^2} = \frac{\partial^2 u_m^k}{\partial x^2} + O(h^2),$$
$$\frac{u_{m-1}^{k+1} - 2u_m^{k+1} + u_{m+1}^{k+1}}{h^2} = \frac{\partial^2 u_m^{k+1}}{\partial x^2} + O(h^2),$$

замінимо різниці у формулі нев'язки

$$r_m^{k+\frac{1}{2}} = \frac{a^2}{2}\left(\frac{\partial^2 u_m^k}{\partial x^2} + \frac{\partial^2 u_m^{k+1}}{\partial x^2}\right) - \frac{u_m^{k+1} - u_m^k}{\tau} + f_m^{k+\frac{1}{2}} + O(h^2). \qquad (13.38)$$

Розкладемо функції u_m^k, u_m^{k+1}, що входять в останню формулу, в ряд Тейлора в околі точки $(x_m, t_k + \tau/2)$. Для скорочення запису позначимо через $t_* = t_k + \tau/2$. Тоді матимемо:

$$u_m^k = u(x_m, t_* - \tau/2) = u(x_m, t_*) - \frac{\tau}{2}\frac{\partial u(x_m, t_*)}{\partial t} + \frac{\tau^2}{8}\frac{\partial^2 u(x_m, t_*)}{\partial t^2} + O(\tau^3),$$
$$u_m^{k+1} = u(x_m, t_* + \tau/2) = u(x_m, t_*) + \frac{\tau}{2}\frac{\partial u(x_m, t_*)}{\partial t} + \frac{\tau^2}{8}\frac{\partial^2 u(x_m, t_*)}{\partial t^2} + O(\tau^3),$$

звідки одержимо

$$\frac{u_m^{k+1} - u_m^k}{\tau} = \frac{\partial u(x_m, t_*)}{\partial t} + O(\tau^2). \qquad (13.39)$$

Аналогічно розкладемо й похідні:

$$\frac{\partial^2 u_m^k}{\partial x^2} = \frac{\partial^2 u(x_m, t_* - \tau/2)}{\partial x^2} = \frac{\partial^2 u(x_m, t_*)}{\partial x^2} - \frac{\tau}{2}\frac{\partial^3 u(x_m, t_*)}{\partial x^3} + O(\tau^2),$$
$$\frac{\partial^2 u_m^{k+1}}{\partial x^2} = \frac{\partial^2 u(x_m, t_* + \tau/2)}{\partial x^2} + \frac{\tau}{2}\frac{\partial^3 u(x_m, t_*)}{\partial x^3} + O(\tau^2),$$

звідки матимемо, що

$$\frac{1}{2}\left(\frac{\partial^2 u_m^k}{\partial x^2} + \frac{\partial^2 u_m^{k+1}}{\partial x^2}\right) = \frac{\partial^2 u(x_m, t_*)}{\partial x^2} + O(\tau^2).$$

Підставляючи останній вираз і вираз (13.39) у формулу для нев'язки (13.38), одержимо:

$$r_m^{k+\frac{1}{2}} = a^2\frac{\partial^2 u_m^{k+\frac{1}{2}}}{\partial x^2} - \frac{\partial u_m^{k+\frac{1}{2}}}{\partial t} + f_m^{k+\frac{1}{2}} + O(\tau^2 + h^2).$$

Враховуючи, що $u(x, t)$ є точним розв'язком рівняння (13.2), маємо:

$$r_m^{k+\frac{1}{2}} = O(\tau^2 + h^2),$$

тобто неявна різницева схема Кранка–Ніколсона апроксимує початкове рівняння (13.2) з точністю $O(\tau^2 + h^2)$.

Перевіримо стійкість схеми Кранка–Ніколсона методом гармонік. Покладемо в рівнянні (13.37) $f_m^{k+1/2} = 0$, підставимо в нього $v_m^k = \lambda^k e^{im\alpha}$, і після скорочень на $\lambda^k e^{im\alpha}$ одержимо:

$$\frac{\lambda - 1}{\tau} = a^2 \frac{e^{-i\alpha} - 2 + e^{i\alpha} + \lambda(e^{-i\alpha} - 2 + e^{i\alpha})}{2h^2}.$$

Спростивши останнє рівняння і використавши відомі тригонометричні формули, як це було зроблено в прикладі 13.1, отримаємо:

$$\lambda = 1 - 4\sigma^2 \sin^2 \alpha/2 - \lambda\, 4\sigma^2 \sin^2 \alpha/2,$$

де $\sigma^2 = a^2 \tau/h^2$.
Звідси

$$\lambda = \frac{1 - \sigma^2 \sin^2 \alpha/2}{1 + \sigma^2 \sin^2 \alpha/2}, \quad |\lambda| \leqslant 1. \tag{13.40}$$

Оскільки в умові (13.40) завжди $|\lambda| \leqslant 1$ незалежно від величини кроків сітки, схема Кранка–Ніколсона абсолютно стійка. Це дозволяє використовувати її з довільними кроками h і τ.

Поряд із розглянутими вище застосовуються й інші схеми апроксимації рівняння теплопровідності (13.2) з менш прозорою логікою побудови. Нижче наводяться дві з них.

13.4.2. Схема Дюфорта–Франкела

Схема будується за чотириточковим тришаровим шаблоном (рис. 13.3, *б*) і має вигляд:

$$\frac{v_m^{k+1} - v_m^{k-1}}{2\tau} = a^2 \frac{v_{m+1}^k - v_m^{k+1} - v_m^{k-1} + v_{m-1}^k}{h^2} + f_m^k. \tag{13.41}$$

Ця явна схема є абсолютно стійкою, і її похибка дорівнює:

$$r_m^k = O(\tau^2 + h^2) + O\left(\frac{\tau^2}{h^2}\right). \tag{13.42}$$

Розв'язок рівняння збігається за умови, що $\tau/h \to 0$, для $\tau, h \to 0$.

13.4.3. Неявна п'ятиточкова тришарова схема

Схема будується за шаблоном, зображеним на рис. 13.3, *в*, і визначається такою формулою:

$$\frac{3}{2}\frac{v_m^{k+1}-v_m^k}{\tau}-\frac{1}{2}\frac{v_m^k-v_m^{k-1}}{\tau}=a^2\frac{v_{m+1}^{k+1}-2v_m^{k+1}+v_{m-1}^{k+1}}{h^2}+f_m^{k+1}. \qquad (13.43)$$

Вона також є абсолютно стійкою і має похибку апроксимації $r_m^{k+1}=O(\tau^2+h^2)$. Тришарові схеми для рівнянь теплопровідності застосовуються значно рідше двошарових, їх іноді використовують для підвищення порядку апроксимації чи для покращення стійкості. Щоб почати розрахунки за формулами тришарових схем, необхідно одержати розв'язок на першому часовому шарі будь-яким двошаровим методом.

13.5. Різницеві методи розв'язання мішаної задачі для параболічного рівняння з двома просторовими змінними

Розглянемо рівняння теплопровідності (рис. 13.4):

$$\frac{\partial u}{\partial t}=a^2\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)+f(x,y,t),\quad 0<x<l_x,\ 0<y<l_y,\ 0<t, \qquad (13.44)$$

$$u(x,y,0)=\varphi(x,y), \qquad (13.45)$$

$$\begin{aligned} &u(0,y,t)=\psi_1(y,t),\quad u(l_x,y,t)=\psi_2(y,t),\\ &u(x,0,t)=\xi_1(x,t),\quad u(x,l_y,t)=\xi_2(x,t). \end{aligned} \qquad (13.46)$$

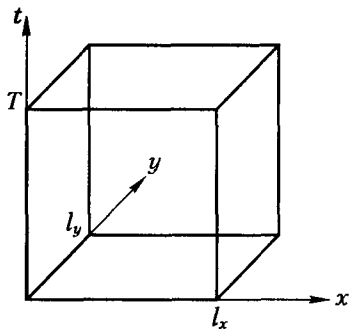


Рис. 13.4. Область розв'язку задачі (13.44) – (13.46)

Як видно на рис. 13.4, розв'язок необхідно шукати всередині області незалежних змінних (x, y, t) із прямокутником $l_x \times l_y$ в основі, що являє собою паралелепіпед. Значення розв'язку в основі паралелепіпеда задаються відомими початковими даними (13.45), а на бокових гранях — заданими граничними умовами (13.46).

Насамперед уведемо в розрахункову область сітку:

$$\omega_h = \{(x_m, y_n, t_k)\}:$$
$$x_m = mh, \quad m = 0, ..., M,$$
$$y_n = nh, \quad n = 0, ..., N,$$
$$t_k = k\tau, \quad k = 0, ..., K,$$
$$M = l_x/h, \quad N = l_y/h, \quad K = T/\tau.$$

Обмежимося розглядом областей найпростішої структури — прямокутних із постійними кроками сітки. В області трьох незалежних змінних сітка триіндексна. Вузол $\binom{k}{m,n}$ лежить на перетині шарів із індексами k, m та n.

13.5.1. Явні і неявні різницеві схеми

Обмежимо, як звичайно, задачу обчислення наближених значень розв'язку (13.44), (13.45) і (13.46) у вузлах сітки. Шукане значення у вузлі $\binom{k}{m,n}$ будемо позначати через $v^k_{m,n}$, а всю множину шуканих величин будемо розглядати як сіткову функцію V^h. Побудуємо для V^h різницеву схему, що відповідає шаблону явної шеститочкової схеми (рис. 13.5, *а*):

$$\frac{v^{k+1}_{m,n} - v^k_{m,n}}{\tau} = a\frac{v^k_{m-1,n} - 2v^k_{m,n} + v^k_{m+1,n} + v^k_{m,n-1} - 2v^k_{m,n} + v^k_{m,n+1}}{h^2} + f^k_{m,n}, \qquad (13.47)$$
$$m = 1, ..., M-1, \quad n = 1, ..., N-1, \quad k = 1, ..., K-1,$$
$$v^0_{m,n} = \varphi_{m,n}, \quad m = 0, ..., M, \quad n = 0, ..., N,$$
$$v^k_{0,n} = \psi_1(y_n, t_k), \quad v^k_{M,n} = \psi_2(y_n, t_k), \quad n = 1, ..., N, \; k = 1, ..., K, \qquad (13.48)$$
$$v^k_{m,0} = \xi_1(x_m, t_k), \quad v^k_{m,N} = \xi_2(x_m, t_k), \quad m = 0, ..., N, \; k = 1, ..., K. \qquad (13.49)$$

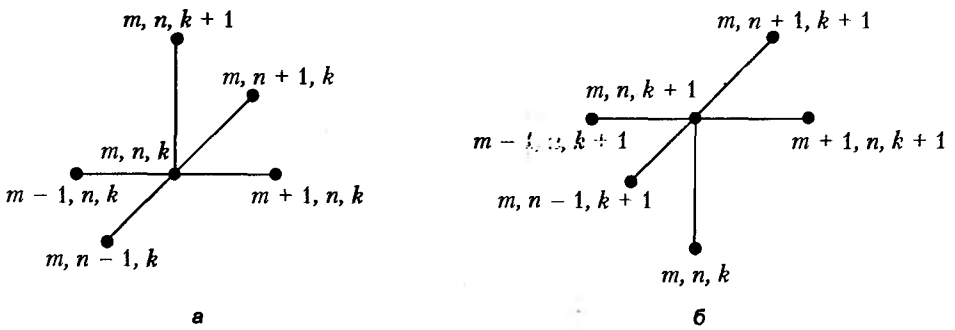


Рис. 13.5. Шаблони різницевих схем: *а* — явної; *б* — неявної

Відзначимо, що методи аналізу порядку апроксимації та стійкості різницевих схем, розглянуті вище для одновимірного параболічного рівняння, досить просто переносяться на багатовимірні рівняння. Перейдемо до визначення порядку

апроксимації різницевої схеми (13.47), (13.48), (13.49). Нев'язка різницевого рівняння (13.47) на точному розв'язку диференціального рівняння (13.44) визначається за формулою

$$r_{m,n}^k = a\frac{u_{m-1,n}^k - 2u_{m,n}^k + u_{m+1,n}^k + u_{m,n-1}^k - 2u_{m,n}^k + u_{m,n+1}^k}{h^2} - \frac{u_{m,n}^{k+1} - u_{m,n}^k}{\tau} + f_{m,n}^k.$$

Підставимо в неї формули для скінченних різниць (12.13):

$$r_{m,n}^k = a\left(\frac{\partial^2 u_{m,n}^k}{\partial x^2} + \frac{\partial^2 u_{m,n}^k}{\partial y^2}\right) - \frac{\partial u_{m,n}^k}{\partial t} + f_{m,n}^k + O(\tau) + O(h^2) = O(\tau + h^2).$$

Оскільки $u(x, y, t)$ є точним розв'язком диференціального рівняння (13.44), одержимо остаточно:

$$r_{m,n}^k = O(\tau + h^2).$$

У граничних вузлах похибка апроксимації нульова. Таким чином, явна різницева схема (13.47), (13.48), (13.49) є схемою першого порядку апроксимації за часом і другим порядком — за просторовими змінними.

Стійкість явної різницевої схеми перевіримо за допомогою методу гармонік. Будемо шукати розв'язок однорідного різницевого рівняння, що відповідає (13.47), у вигляді:

$$v_{m,n}^k = \lambda^k e^{i(m\alpha + n\beta)}. \tag{13.50}$$

Підставляючи (13.50) у рівняння (13.47) з $f_{m,n}^k = 0$, після скорочень на $\lambda^k e^{i(m\alpha+n\beta)}$ одержимо:

$$\frac{\lambda - 1}{\tau} = a\left(\frac{e^{-i\alpha} - 2 + e^{i\alpha}}{h^2} + \frac{e^{-i\alpha} - 2 + e^{i\alpha}}{h^2}\right).$$

Виконавши тригонометричні перетворення, знайдемо значення λ:

$$\lambda = 1 - 2\sigma^2(1 - \cos\alpha) - 2\sigma^2(1 - \cos\beta) = 1 - 4\sigma^2(\sin^2\alpha/2 + \sin^2\beta/2),$$

де $\sigma^2 = a\tau/h^2$.

Вимагаючи, щоб за будь-яких α і β виконувалася нерівність $|\lambda| \leqslant 1$, приходимо до необхідної умови стійкості явної шеститочкової схеми для двовимірного рівняння теплопровідності:

$$\tau \leqslant \frac{h^2}{4a}, \tag{13.51}$$

що є узагальненням умови стійкості, отриманої раніше для одновимірного рівняння.

Як уже відзначалося в підрозділі 13.2, більша частка обчислень у разі використання різницевих схем для еволюційних задач виконується під час переходу від k-го шару до $(k+1)$-го шару, тобто в рівнянні (13.47), де невідомими є $v_{m,n}^{k+1}$.

Кожне рівняння містить одну невідому, і значення шуканої функції на нульовому шарі відомі з початкової умови $v^0_{m,n} = \varphi_{m,n}$. Тому розв'язання задачі за явною схемою зводиться до розрахунків за формулою (13.52), починаючи з першого часового шару

$$v^0_{m,n} = \varphi_{m,n}, \quad m = 0, ..., M, \; n = 0, ..., N,$$
$$v^{k+1}_{m,n} = \sigma^2(v^k_{m-1,n} + v^k_{m+1,n}) + (1 - 4\sigma^2)v^k_{m,n} + \sigma^2(v^k_{m,n-1} + v^k_{m,n+1}) + \tau f^k_{m,n}, \quad (13.52)$$
$$m = 1, ..., M-1, \; n = 1, ..., N-1, \; k = 1, ..., K-1.$$

Оцінимо трудомісткість обчислень за схемою (13.52). Очевидно, кількість арифметичних операцій пропорційна кількості вузлів сітки. Для простоти оцінок трудомісткості приймемо, що $M = N = 1/h$. Тоді за умови стійкості $\tau = 1/K \approx 1/M^2$. Отже, кількість вузлів і відповідно кількість операцій буде $W \approx M^2 K \approx M^4$. Для $M = 10^2$ кількість операцій буде мати порядок $W \approx 10^8$, а це вже забагато для серійних розрахунків, і це змушує шукати більш ефективні алгоритми.

Розглянемо тепер неявну шеститочкову різницеву схему для розв'язання задачі (13.43), (13.44), (13.45). Шаблон схеми зображений на рис. 13.5, *б*. Різницеві рівняння для внутрішніх вузлів мають такий вигляд:

$$\frac{v^{k+1}_{m,n} - v^k_{m,n}}{\tau} = a\frac{v^{k+1}_{m-1,n} - 2v^{k+1}_{m,n} + v^{k+1}_{m+1,n} + v^{k+1}_{m,n-1} - 2v^{k+1}_{m,n} + v^{k+1}_{m,n+1}}{h^2} + f^{k+1}_{m,n}, \quad (13.53)$$
$$m = 1, ..., M-1, \; n = 1, ..., N-1, \; k = 0, ..., K-1,$$
$$v^0_{m,n} = \varphi_{m,n}, \quad m = 0, ..., M, \; n = 0, ..., N.$$

Інші рівняння схеми збігаються з (13.48) і (13.49). Перепишемо рівняння (13.53), переносячи невідомі в його ліву частину:

$$\sigma^2(v^{k+1}_{m-1,n} + v^{k+1}_{m+1,n}) - (1 + 4\sigma^2)v^{k+1}_{m,n} + \sigma^2(v^{k+1}_{m,n-1} + v^{k+1}_{m,n+1}) = -v^k_{m,n} - \tau f^k_{m,n}, \quad (13.54)$$
$$m = 1, ..., M-1, \; n = 1, ..., N-1, \; k = 0, ..., K-1.$$

Метод гармонік для цієї схеми застосовується так само, як і для явної схеми, і приводить до рівняння

$$2\lambda\sigma^2 \cos\alpha - \lambda(1 + 4\sigma^2) + 2\lambda\sigma^2 \cos\alpha = -1.$$

Звідси одержимо такий вираз для λ:

$$\lambda = \frac{1}{1 + 4\sin^2 \alpha/2 + 4\sin^2 \beta/2},$$

що свідчить про абсолютну стійкість цього неявного методу.

Розглянемо методи розв'язання рівнянь (13.54), кожне з яких містить п'ять невідомих. Їх розв'язки обчислюють для окремих часових шарів, починаючи з першого. Якщо перенумерувати невідомі $v^{k+1}_{m,n}$ для перетворення їх в елементи одноіндексного масиву (вектора), щоб одержати традиційний запис лінійної

системи рівнянь (13.54), то обчислення розв'язку на $(k+1)$-му шарі зводиться до розв'язання системи з п'ятидіагональною матрицею. На жаль, заповнені ненульовими елементами діагоналі не утворять «суцільної стрічки». Тому, незважаючи на сильну розрідженість матриці, побудувати простий економічний метод розв'язання такої системи (подібний до методу прогону для систем із тридіагональною матрицею) не так просто. Якщо використовувати метод Гаусса, то одержимо вкрай несприятливі оцінки ефективності: кількість необхідних операцій на одному кроці за часом пропорційна кубу числа невідомих, тобто $(M^2)^3 = M^6$.

Враховуючи стрічкову структуру матриці, можна на два порядки знизити трудомісткість обчислень на одному часовому кроці (розділ 3). Кількість елементарних операцій, необхідних для розв'язання методом виключення системи зі стрічковою матрицею $\approx s^2P$, де s — половина ширини стрічки, а P — порядок системи. У силу абсолютної стійкості схеми тут немає обмежень на вибір кроку за часом, і якщо з міркувань точності можна вибрати $\tau \approx h$ (тобто $K = M$), то, враховуючи, що в нашому випадку $s \approx M$, $P \approx M^2$, загальний обсяг необхідних для розв'язання задачі операцій буде $W \approx M^5$. Це менше, ніж для явної схеми.

Оскільки явна і неявна шеститочкові схеми не є економічними, розглянемо методи побудови різницевих схем для двовимірного рівняння теплопровідності, що приводять до більш ефективних чисельних алгоритмів.

13.5.2. Різницеві схеми розщеплення

Різницеві схеми розщеплення використовуються, якщо кількість просторових змінних — дві і більше. Ці схеми крім стійкості забезпечують мінімальний обсяг обчислень і є одним із важливих засобів розв'язання багатовимірних нестаціонарних задач. Розглянемо метод побудови схеми розщеплення для двовимірного рівняння теплопровідності (13.44), (13.45):

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}, \quad u(x, y, 0) = \varphi(x, y). \tag{13.55}$$

Перепишемо рівняння (13.55) в операторній формі, прийнявши (для скорочення запису) $a = 1$:

$$\frac{\partial u}{\partial t} = Lu, \quad \text{де} \quad Lu = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}. \tag{13.56}$$

Припустимо, що розв'язок (13.55) у момент часу t_k відомий, тобто відомі значення функції $u(x, y, t_k)$. Значення функції $u(x, y, t_{k+1})$ розкладемо з урахуванням відомого значення $u(x, y, t_k)$ у ряд Тейлора:

$$u(x, y, t_{k+1}) = u(x, y, t_k) + \tau\frac{\partial u}{\partial t} + O(\tau^2) = u(x, y, t_k) + \tau\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + O(\tau^2) =$$

$$= u(x, y, t_k) + \tau Lu(x, y, t_k) + O(\tau^2) = (E + \tau L)u(x, y, t_k) + O(\tau^2),$$

де E — одиничний оператор $(Eu = u)$.

Покладемо $L_1 = \partial^2/\partial x^2$, $L_2 = \partial^2/\partial y^2$, тоді $L = L_1 + L_2$. Поряд із задачею (13.55) розглянемо дві допоміжні задачі:

$$\frac{\partial z}{\partial t} = \frac{\partial^2 z}{\partial x^2}, \quad z(x, y, t_k) = u(x, y, t_k), \quad t_k < t \leqslant t_{k+1}, \tag{13.57}$$

$$\frac{\partial w}{\partial t} = \frac{\partial^2 w}{\partial y^2}, \quad w(x, y, t_k) = z(x, y, t_{k+1}), \quad t_k < t \leqslant t_{k+1}. \tag{13.58}$$

Відзначимо, що задачі (13.57) і (13.58), на відміну від задачі (13.56), одновимірні. Їх можна розв'язувати незалежно і послідовно, спочатку задачу (13.57) — це дозволить обчислити функцію $z(x, y, t_{k+1})$, потім задачу (13.58) — це дозволить обчислити функцію $w(x, y, t_{k+1})$. Встановимо зв'язок між значеннями $w(x, y, t_{k+1})$ і $w(x, y, t_{k+1})$. Матимемо:

$$\begin{aligned}
z(x, y, t_{k+1}) &= (E + \tau L_1)\, z(x, y, t_k) + O(\tau^2) = (E + \tau L_1)\, u(x, y, t_k) + O(\tau^2), \\
w(x, y, t_{k+1}) &= (E + \tau L_2)\, z(x, y, t_k) + O(\tau^2) = (E + \tau L_2)\, z(x, y, t_{k+1}) + O(\tau^2) = \\
&= (E + \tau L_1)[(E + \tau L_2)\, z(x, y, t_k) + O(\tau^2)] = \\
&= [E + \tau(L_1 + L_2) + \tau^2 L_1 L_2]\, u(x, y, t_k) + O(\tau^2) = \\
&= (E + \tau L)\, u(x, y, t_k) + O(\tau^2) = \\
&= u(x, y, t_{k+1}) + O(\tau^2).
\end{aligned}$$

Таким чином, якщо послідовно розв'язати задачі (13.57) і (13.58), то для $t = t_{k+1}$, одержимо значення функції $w(x, y, t_{k+1})$, які відрізняються від дійсного розв'язку $u(x, y, t_{k+1})$ задачі (13.56) лише на величину $O(\tau^2)$. При цьому вдалося замінити розв'язання двовимірної задачі (13.56) послідовним розв'язанням двох одновимірних задач (13.57) і (13.58). Трудомісткість обчислень за такого підходу менша, ніж трудомісткість інших, і це робить метод розщеплення вигідним.

Побудуємо схему розщеплення, замінивши рівняння (13.57) і (13.58) неявними різницевими схемами відповідно до шаблона, наведеного на рис. 13.6, *а*. Цей метод називається локально-одновимірним методом. Передбачається двоетапний спосіб переходу від k-го шару до $(k + 1)$-го шару:

$$\frac{\vartheta_{m,n}^{k+1} - v_{m,n}^{k}}{\tau} = a^2 \frac{\vartheta_{m-1,n}^{k+1} - 2\vartheta_{m,n}^{k+1} + \vartheta_{m+1,n}^{k+1}}{h^2} + \frac{1}{2} f_{m,n}^{k}, \tag{13.59}$$

$$m = 1, \ldots, M-1, \quad n = 1, \ldots, N-1$$

із граничними умовами для $m = 0, M$;

$$\frac{v_{m,n}^{k+1} - \vartheta_{m,n}^{k+1}}{\tau} = a^2 \frac{v_{m,n-1}^{k+1} - 2v_{m,n}^{k+1} + v_{m,n+1}^{k+1}}{h^2} + \frac{1}{2} f_{m,n}^{k+1}, \tag{13.60}$$

$$m = 1, \ldots, M-1, \quad n = 1, \ldots, N-1$$

із граничними умовами для $n = 0, N$.

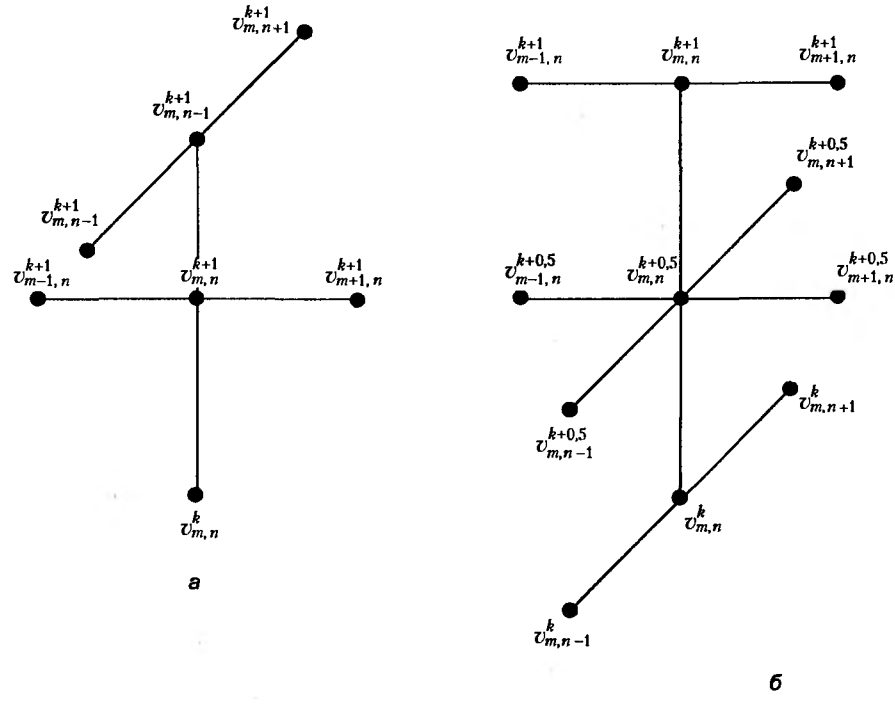


Рис. 13.6. Шаблони різницевих схем: *а* — локально-одновимірна схема; *б* — метод змінних напрямків

Система рівнянь (13.59) розшаровується на m тридіагональних систем рівнянь для кожного фіксованого n. Останні можна розв'язати методом прогону за напрямом x. Кількість необхідних операцій при цьому $\approx MN \approx M^2$. Аналогічно система рівнянь (13.60) розшаровується на n тридіагональних систем рівнянь для кожного фіксованого m, які можна розв'язати методом прогону за напрямом y. Сумарна кількість операцій на першому і другому етапах $\approx M^2$. Таким чином, розрахунок на кожному етапі є економічним.

Відзначимо, що на етапі розв'язання (13.59) для кожного n відповідна тридіагональна система рівнянь еквівалентна неявній чотириточковій схемі для одновимірного рівняння теплопровідності (див. підрозділ 13.2.1), яка абсолютно стійка. Те ж саме стосується й етапу розв'язання (13.60). І, як результат, локально-одновимірна схема також є абсолютно стійкою. В цьому об'єднанні вибір кроку τ зв'язаний лише жорсткою умовою точності обчислень. І якщо це припустимо, то можна взяти $\tau \approx h$ і $K \approx M$. У цьому випадку загальна кількість операцій обчислення розв'язку в усіх вузлах сітки $MK \approx M^3$, що цілком припустимо для сучасних обчислювальних засобів.

Відмітимо, що на кожному окремому етапі різницеві рівняння не апроксимують диференціальне рівняння. Однак має місце так звана сумарна *апроксимація*: різницева схема, що є наслідком (13.59), (13.60), після виключення проміжних значень $\vartheta^{k+1}_{m,n}$ апроксимує початкову задачу з порядком $O(\tau^2 + h^2)$.

13.5.3. Метод Пісмена–Речфорда

Розглянемо ще один підхід до побудови ефективних схем для багатовимірних нестаціонарних задач, що одержав назву методу Пісмена–Речфорда.

Задачу (13.55) можна записати у вигляді:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + \frac{1}{2}\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right), \quad u(x, y, 0) = \varphi(x, y) \tag{13.61}$$

і поставити їй у відповідність на відрізку $t_k \leqslant t \leqslant t_{k+1}$ дві системи:

$$\frac{\partial z}{\partial t} = \frac{1}{2}\left(\frac{\partial^2 z}{\partial x^2} + \frac{\partial^2 z}{\partial y^2}\right), \quad z(x, y, t_k) = u(x, y, t_k), \quad t_k < t \leqslant t_{k+1} \tag{13.62}$$

і

$$\frac{\partial w}{\partial t} = \frac{1}{2}\left(\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2}\right), \quad w(x, y, t_k) = z(x, y, t_{k+1}), \quad t_k < t \leqslant t_{k+1}. \tag{13.63}$$

Різницеву схему побудуємо за шаблоном, наведеним на рис. 13.6, *б*. Спочатку з рівнянь

$$\frac{v_{m,n}^{k+\frac{1}{2}} - v_{m,n}^{k}}{\tau} =$$

$$= \frac{a^2}{2}\left(\frac{v_{m,n-1}^{k} - 2v_{m,n}^{k} + v_{m,n+1}^{k}}{h^2} + \frac{v_{m-1,n}^{k+\frac{1}{2}} - 2v_{m,n}^{k+\frac{1}{2}} + v_{m+1,n}^{k+\frac{1}{2}}}{h^2}\right) + \frac{1}{2} f_{m,n}^{k+\frac{1}{2}}, \tag{13.64}$$

$$m = 1, \ldots, M-1, n = 1, \ldots, N-1,$$

доповнених граничними умовами, коли $m = 0, M$, знаходять допоміжні невідомі $v_{m,n}^{k+\frac{1}{2}}$ на проміжному шарі. Потім з рівнянь

$$\frac{v_{m,n}^{k+1} - v_{m,n}^{k+\frac{1}{2}}}{\tau} =$$

$$= \frac{a^2}{2}\left(\frac{v_{m-1,n}^{k+\frac{1}{2}} - 2v_{m,n}^{k+\frac{1}{2}} + v_{m+1,n}^{k+\frac{1}{2}}}{h^2} + \frac{v_{m,n-1}^{k+1} - 2v_{m,n}^{k+1} + v_{m,n+1}^{k+1}}{h^2}\right) + \frac{1}{2} f_{m,n}^{k+\frac{1}{2}}, \tag{13.65}$$

$$m = 1, \ldots, M-1, n = 1, \ldots, N-1$$

з граничними умовами, коли $n = 0, N$, знаходять розв'язок на $(k+1)$-му шарі. Тут так само, як і для локально-одновимірної схеми, система (13.64) розшаровується на три діагональні системи лінійних рівнянь для кожного фіксованого n, а вираз (13.65) зводиться до аналогічних систем для кожного фіксованого m.

Приклад 13.3

Розв'яжемо двовимірну крайову задачу в системі Mathematica:

$$\frac{\partial u}{\partial t} = a\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right), \quad 0 < x, y < 1, \ 0 < t < T,$$
$$u(x, y, 0) = \sin \pi x \sin \pi y, \tag{13.66}$$
$$u(0, y, t) = u(1, y, t) = 0,$$
$$u(x, 0, t) = u(x, 1, t) = 0.$$

Matematica не має стандартного оператора для розв'язання двовимірних крайових задач, тому необхідно складати програми реалізації відповідного алгоритму. Запишемо локально-одновимірну різницеву схему у вигляді, зручному для розв'язання методом прогону. Спочатку здійснюється прогін за умови зміни індексу m для кожного фіксованого n:

$$\begin{cases} v_{0n} = 0, \\ s v_{m-1,n} - (1+2s) v_{mn} + s v_{m+1,n} = -u_{mn}^k, \\ v_{M-1,n} = 0. \end{cases}$$
$$m = 1, ..., M-2, \quad n = 1, ..., M-1, \quad s = a^2 \tau / h^2,$$

де $v_{m,n}$ — проміжні значення схеми розщеплення.
Потім проводиться прогін за умови зміни індексу n для кожного фіксованого m:

$$\begin{cases} u_{m0}^{k+1} = 0, \\ s u_{mn-1}^{k+1} - (1+2s) u_{mn}^{k+1} + s u_{mn+1}^{k+1} = -v_{mn}, \\ u_{m,N-1} = 0, \end{cases}$$
$$n = 1, ..., M-2.$$

Скористуємось програмою розв'язання задачі (13.66) за допомогою локально-одновимірної різницевої схеми. Введемо початкові дані й обчислимо початкові умови у вузлах сітки:

```
In[]:= m = 20; k = 10;
       Array[u, {m,m}, 0]; Array[v, {m,m}, 0];
       h = 1/(m - 1); a = 1; t = 0.25 h^2/a; s = a t/h^2; s1 = 1 + 2 s;
       Do [u[i,j] = N[Sin[Pi i h] Sin[Pi j h]], {i,0,m-1}, {j,0,m-1} ];
       U = Array[u, {m,m}, 0];
       ListPlot3D[U]
```

На рис.13.7 зображена поверхня, що відповідає початковим умовам задачі.
Крайова задача (13.66) має аналітичний розв'язок, який записується такою формулою:

$$u(x, y, t) = e^{-2a^2\pi^2 t} \sin(\pi x) \sin(\pi y). \tag{13.67}$$

Тому наближений чисельний розв'язок можна буде порівняти з точним.
Фрагмент програми розрахунку значень сіткової функції u_{mn}^k послідовно на заданих часових шарах за формулами локально-одновимірної схеми наводиться нижче:

```
In[]:= Do [v[0,j] = 0; v[m-1,j] = 0, {j,0,m-1} ];
       Do [v[i,0] = 0; v[i,m-1] = 0, {i,0,m-1} ];
```

```
Array[l,m,0]; Array[n,m,0];
l[0] = 0; n[0] = 0;
Do [ Do [Do [d = s1 - s l[j-1]; l[j] = s/d; n[j] = (u[i,j] + s n[j-1])/d,{j,1,m-2}];
Do [ v[i,m-j] = l[m-j] v[i,m-j + 1] + n[m-j],{j,2,m-1}],{i,1,m-2}];
Do [ Do [d = s1-s l[i-1]; l[i] = s/d; n[i] = (v[i,j] + s n[i-1])/d,{i,1,m-2}];
Do [u[m-i,j] = l[m-i] u[m-i + 1,j] + n[m-i], {i,2,m-1}], {j,1,m-2}];
U = Array[u,{m,m},0]; ListPlot3D[U];
anmax = N[u[m/2,m/2]]; gmax = N[Exp[-2 Pi^2 k1 t a]];
Print["k = ",k1," ",anmax," ",gmax," eps = ",gmax-anmax],{k1,1,k}]
```

Ця програма виводить на екран графік поверхні розв'язку на всіх часових шарах, одна з яких наведена на рис. 13.8. У табл. 13.1 подані значення наближеного і точного розв'язків у точці, де функція досягає максимуму, і похибки для цієї точки.

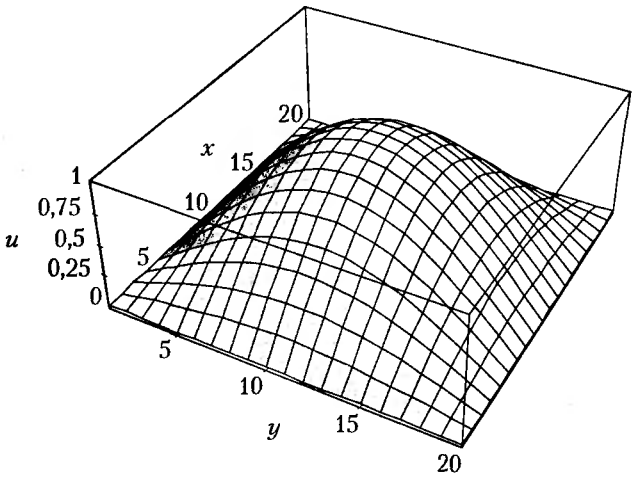


Рис. 13.7. Розв'язок крайової задачі (13.66), що відповідає початковим умовам

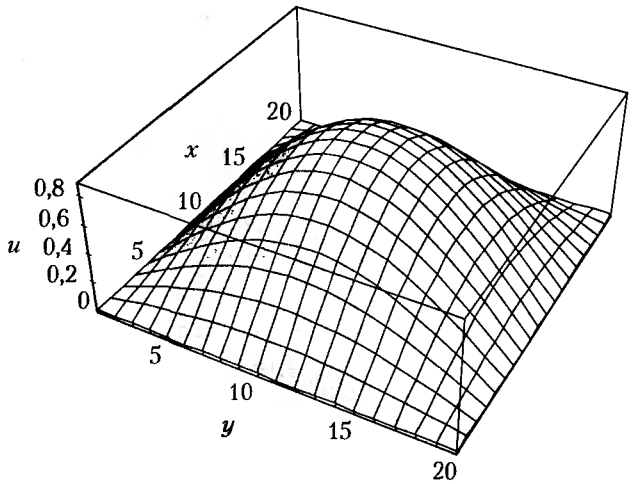


Рис. 13.8. Розв'язок крайової задачі для $t = 10\tau$

Таблиця 13.1. Значення чисельного розв'язку крайової задачі

k	u_{mn}^k	$u(x, y, t)$	ε	k	u_{mn}^k	$u(x, y, t)$	ε
1	0,979772	0,986423	0,006650	6	0,915397	0,921255	0,005857
2	0,966545	0,973031	0,006485	7	0,903039	0,908747	0,005707
3	0,953496	0,95982	0,006323	8	0,903039	0,908747	0,005708
4	0,940624	0,946789	0,006165	9	0,878821	0,884239	0,005418
5	0,927925	0,933934	0,00601	10	0,866956	0,872233	0,005277

13.6. Метод установлення

Метод установлення звичайно застосовують для розв'язання еліптичних чи мішаних еліптично-гіперболічних задач. Ідею методу встановлення розглянемо на прикладі задачі Діріхле для рівняння Пуассона:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y,), \quad u(x, y)|_\Gamma = \psi(x, y)|_\Gamma . \tag{13.68}$$

Задачу (13.68) будемо називати стаціонарною, тому що розв'язок задачі не залежить від часу. Поряд із задачею (13.68) розглянемо еволюційну (нестаціонарну) задачу для параболічного рівняння з тими ж граничними умовами і довільно обраними початковими умовами:

$$\begin{gathered} \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + f(x, y), \\ u(x, y, t)|_\Gamma = \psi(x, y)|_\Gamma , \quad u(x, y, 0) = \varphi(x, y). \end{gathered} \tag{13.69}$$

У загальному випадку задачу (13.69) називають нестаціонарною задачею, якщо граничні умови залежать від часу. В представленому формулюванні задача (13.69) називається виродженою нестаціонарною задачею, тому що умова на границі не залежить від часу. З фізичних міркувань очевидно, що розв'язок еволюційної задачі (13.69) для $t \to \infty$ буде прямувати до розв'язку стаціонарної задачі (13.68). Дійсно, розв'язок стаціонарної задачі відображає розподіл значень шуканої функції (наприклад, температури) в області G за заданого її розподілу на межі області Γ (рис. 13.9). Розв'язок еволюційної задачі (13.69) дає розподіл значень функції в часі для тих же, що й у стаціонарній задачі, граничних умов. Очевидно, що через досить великий проміжок часу в області G встановиться розподіл значень функції, що залежить не від початкових умов, а від значень функції на межі. Вищенаведені міркування дозволяють припустити, що розв'язок еволюційної задачі для $t \to \infty$ збігається до розв'язку стаціонарної задачі й розв'язок еволюційної задачі за досить великого $t = T$ може бути прийнятий за наближений розв'язок стаціонарної задачі.

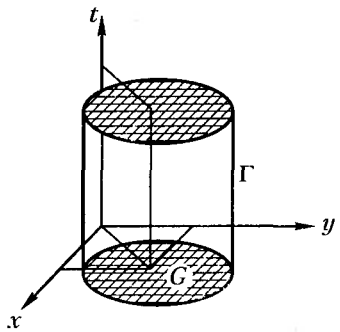


Рис. 13.9. Область розв'язку задачі методом установлення

Таким чином, у методі встановлення вводиться нова незалежна змінна t, і задача формально ускладнюється, тому що збільшується розмірність задачі. Однак при цьому збільшується набір можливих різницевих схем серед тих, які були розглянуті у даному розділі, що допускає вибір із них стійких і економічних.

Приклад 13.4

Розв'яжемо стаціонарну задачу про розподіл температури $T(x, y, t)$ в алюмінієвій пластинці розміром $l \times l$ см:

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0, \quad 0 < x, y < l.$$

Граничні умови по боках пластинки:

$$T(x, 0) = 0, T(x, l) = 150 - 100\, x/l, T(0, y) = 150\, y/l, T(l, y) = 50\, y/l.$$

Застосуємо метод встановлення для розв'язання задачі:

$$\frac{\partial T}{\partial t} = a\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right), \quad 0 < x, y < l, 0 < t < T$$

з граничними умовами виду

$$T(x, 0) = 0, T(x, l) = 150 - 100\, x/l, T(0, y) = 150\, y/l, T(l, y) = 50\, y/l$$

та початковими умовами, що задаються для $t = 0$.

Щоб спростити процедуру розв'язування, використаємо явну різницеву схему і задамо однаковий крок як на осі $0x$, так і на осі $0y$. Приймемо, що $M = N$, тоді $h = l/M$.

$$\frac{v_{m,n}^{k+1} - v_{m,n}^{k}}{\tau} = a\frac{v_{m-1,n}^{k} - 2v_{m,n}^{k} + v_{m+1,n}^{k} + v_{m,n-1}^{k} - 2v_{m,n}^{k} + v_{m,n+1}^{k}}{h^2},$$

$$m = 1, ..., M-1, \quad n = 1, ..., N-1, \quad k = 1, ..., K-1,$$

$$v_{m,n}^{0} = \varphi_{m,n}, \quad m = 0, ..., M, \quad n = 0, ..., N, \tag{13.70}$$

$$v_{m,0}^{k} = 0, \quad v_{m,N}^{k} = 150 - 100\, m/M, \quad m = 0, ..., M, \quad k = 1, ..., K,$$

$$v_{0,n}^{k} = 150\, n/N, \quad v_{M,n}^{k} = 50\, n/N, \quad n = 0, ..., M, \quad k = 1, ..., K.$$

Для розв'язання системи (13.70) використаємо засоби пакету. Спочатку введемо величину кроку за часом таким чином, щоб виконувались умови збіжності. Потім задамо початкові та граничні умови для шуканої функції:

```
In[]:= M = 11; k = 1; eps = 0.0001; eps1 = 10; Array[u, {M,M}, 0];
       h = 1/(M - 1); a = 0.005; t = 0.25 h^2/a; s = a t/h^2;
       Do [u[m,n] = 100*n/(M-1), {m, 0 ,M-1}, {n, 0, M-1} ];
       Do [u[m,0] = 0; u[m, M-1] = 150 - 100*m/(M-1), {m, 0, M-1} ];
       Do [u[0,n] = 150*n/(M-1); u[M-1,n] = 50*n/(M-1), {n, 0, M-1} ];
       U = Array[u, {M,M}, 0];    ListPlot3D[U, AxesLabel → {"y","x","u"}]
```

Початкові значення показані на рис. 13.10.

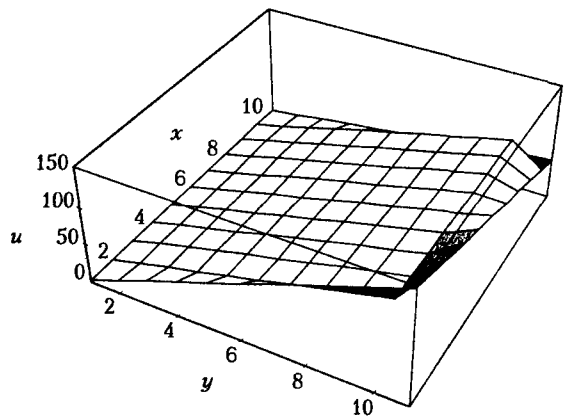


Рис. 13.10. Початкові значення різницевої задачі

Ітерації будемо виконувати до тих пір, поки норма похибки розв'язку на двох сусідніх часових шарах не стане меншою, ніж задане значення ε:

```
In[]:= While[eps1 > eps, k = k + 1; eps1 = 0;
       Do [ Do [v[i,j] = s (u[i-1,j] + u[i+1,j]) + (1-4 s) u[i,j] + s (u[i,j-1] + u[i,j+1]);
       ep = Abs[u[i,j] - v[i,j]]; eps1 = Max[ep,eps1], {i,1,M-2}, {j,1,M-2}]];
       Do [Do [u[i,j] = v[i,j], {i,1,M-2}, {j,1,M-2}]]];
```

Після завершення ітерацій побудуємо графік отриманого розв'язку (рис. 13.11).

```
In[]:= U = Array[u, {M,M}, 0];
       ListPlot3D[U, ViewPoint → {1.2, 1.2, 1.2}, AxesLabel → {"x","y","u"}];
```

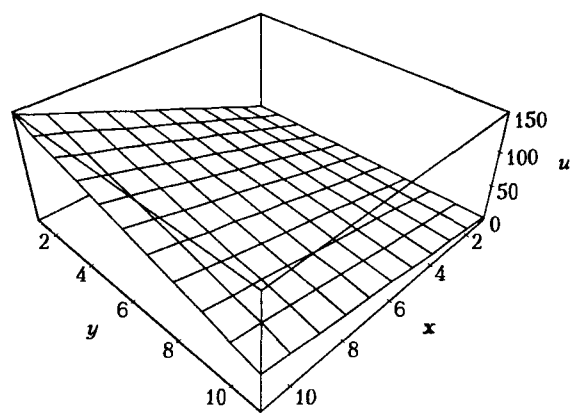


Рис. 13.11. Графік розв'язку задачі (13.69)

13.7. Метод прямих

Основна ідея методу прямих полягає в зведенні рівняння з частинними похідними до системи звичайних диференціальних рівнянь. У цьому й полягає відмінність від методу сіток, який безпосередньо зводить розв'язання рівнянь з частинними похідними до розв'язання систем алгебраїчних рівнянь. Метод прямих можна застосувати для розв'язання рівнянь будь-якого типу, але використовують його звичайно для параболічних і еліптичних рівнянь.

Ідею цього методу покажемо на прикладі лінійного диференціального рівняння параболічного типу зі змінним коефіцієнтом:

$$\frac{\partial u}{\partial t} = a(x,t)\frac{\partial^2 u}{\partial x^2} + f(x,t), \qquad a(x,t) > 0, \quad 0 < x < l, \quad 0 < t \tag{13.71}$$

із початковими

$$u(x,0) = \varphi(x), \quad 0 \leqslant x \leqslant l, \quad t = 0$$

і граничними умовами першого роду

$$u(0,t) = \psi(t), \quad x = 0, \quad t > 0,$$
$$u(l,t) = \xi(t), \quad x = l, \quad t > 0.$$

На відрізку $[0, l]$ виберемо M рівновіддалених точок $x_m = mh$, $m = 0, ..., M$, $h = l/M$ і будемо шукати розв'язок задачі $u(x,t)$ на прямих $x = x_m$, $m = 1, ..., M-1$ в області $t > 0$ (рис. 13.12). Покладемо в рівнянні (13.71) $x = x_m$ і замінимо похідні за x різницевими відношеннями

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{x=x_m} = \frac{u_{m-1}(t) - 2u_m(t) + u_{m+1}(t)}{h^2} + O(h^2), \quad m = 1, ..., M-1.$$

Рис. 13.12. Розв'язання рівнянь методом прямих

Проводячи таку заміну на всіх прямих $x = x_m$ і нехтуючи малими другого порядку, одержимо систему $M-1$ звичайних диференціальних рівнянь першого порядку:

$$\frac{du_m(t)}{dt} = a_m(t)\frac{u_{m-1}(t) - 2u_m(t) + u_{m+1}(t)}{h^2} + f_m(t), \tag{13.72}$$

де $m = 1, ..., M-1$,

$$a_m(t) = a(x_m, t), \quad f_m(t) = f(x_m, t), \quad u_0(t) = \psi(t), \quad u_M(t) = \xi(t) \tag{13.73}$$

із початковими умовами

$$u_m(0) = \varphi(x_m). \tag{13.74}$$

Система звичайних диференціальних рівнянь (13.72), (13.73), (13.74) апроксимує рівняння (13.66) разом із граничними і початковими умовами і називається *системою рівнянь методу прямих*. Цю систему рівнянь розв'язують чисельно або аналітично.

Приклад 13.5

Розв'яжемо одновимірну параболічну задачу стандартними засобами пакета Mathematica, які реалізують метод прямих:

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2} + f(x, t), \quad 0 < x < l, 0 < t,$$
$$u(x, 0) = \varphi(x), u(0, t) = \psi(t), u(1, t) = \xi(t). \tag{13.75}$$

Нехай у задачі (13.75) задані $f(x, t) = te^{-t}$, $a = 1$, початкові умови $\varphi(x) = \sin \pi x/4$ і граничні умови $\varphi(t) = \xi(t) = 0$. Для розв'язання задачі використаємо оператор NDSolve, за допомогою якого одержимо чисельний розв'язок задачі.

```
In[]:= l = 4; T = 9; f[x_, t_]:= t*e^(-t);
       r = NDSolve[{D[u[x,t], t] = D[u[x,t], x, x] + f[x,t], u[x,0] = Sin[π*x/4],
       u[0,t] = 0, u[l,t] = 0}, u, {x, 0, l}, {t, 0, T}]
```

Результатом є таблиця значень сіткової функції, за якою в пакеті будується інтерполяційна функція, що зображує отриманий наближений розв'язок (рис. 13.13).

```
In[]:= U[x_,t_] = u[x,t] /. r[[1,1]];
       Plot3D[U[x,t], {x,0,l}, {t,0,T}, AxesLabel → {x,t,U}, PlotPoints → 30]
```

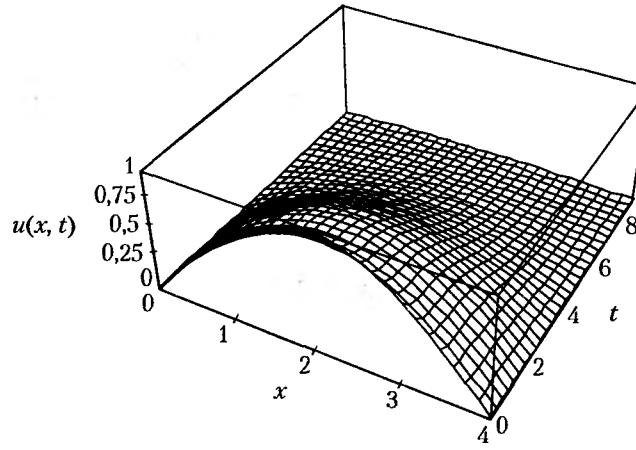


Рис.13.13. Розв'язок одновимірної параболічної задачі

Висновки

1. Оскільки умови стійкості розв'язку одновимірних параболічних рівнянь накладають обмеження на вибір кроків змінних у гірший бік порівняно з розв'язанням еліптичних рівнянь ($\Delta t < k(\Delta x)^2$), то для розв'язання одномірних параболічних рівнянь слід застосовувати неявні схеми обчислень Кранка–Ніколсона і Дюфорта–Франкела, які є абсолютно стійкими і забезпечують підвищену точність.
2. Для розв'язання параболічних рівнянь із двома просторовими змінними умови стійкості розв'язку стають ще більш жорсткими ($\Delta t < [(\Delta x)^2 + (\Delta y)^2]$), тому доцільно застосовувати неявні різницеві схеми розщеплення, чи метод змінних напрямків Пісмена–Ретфорда, які теж є абсолютно стійкими у разі зміни кроків обчислень за часом і просторовими координатами.
3. Різницеві методи зводять задачу розв'язання диференціальних рівнянь із частинними похідними до задачі розв'язання систем лінійних рівнянь. *Метод установлення* і *метод прямих* зводять цю задачу до розв'язання систем звичайних диференціальних рівнянь, що може виявитися для конкретних задач і умов більш ефективною процедурою.
4. Із погляду стійкості розв'язку можна провести певну аналогію між методами розв'язання звичайних диференціальних рівнянь і диференціальних рівнянь із частинними похідними: метод Кранка–Нікольсона можна вважати подібним неявному *A-стійкому* методу трапецій, який використовують для жорстких звичайних диференціальних рівнянь, а метод кінцевих різниць для одновимірних задач — явному методу Ейлера.

Контрольні запитання та завдання

1. Перевірте, що функція $u(x,t) = e^{-(an\pi)^2 t}\sin(n\pi x)$ — це розв'язок рівняння

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} \tag{13.76}$$

для кожного додатного числа $n = 1, 2, \ldots$.

2. Обчисліть за допомогою явної різницевої схеми наближений розв'язок мішаної задачі

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}, \quad 0 < x < 1, 0 < t < 0{,}1$$

з початковою умовою $u(x, 0) = |2x - 1|$ і граничними умовами $u(0, t) = u(1, t) = 0$. Отримайте розв'язок цієї ж задачі за допомогою оператора `NDSolve` системи Mathematica і порівняйте два розв'язки.

3. У формулі (13.21) явної різницевої схеми покладіть $\sigma^2 = a^2\,\tau/h^2 = 1/2$ і спростіть її. Визначте порядок апроксимації рівняння (13.75) на його розв'язку, за допомогою отриманої різницевої формули.

4. У формулі (13.37) неявної різницевої схеми Кранка–Ніколсона покладіть $a^2\,\tau/h^2 = 1$ і спростіть її. Запишіть отриману систему рівнянь, з урахуванням граничних умов $u(0, t) = u(1, t) = 0$, у матричній формі. Чи буде отримана матриця діагональнопереважаючою?

5. Знайдіть нев'язку різницевої схеми, коли $0 \leqslant \theta \leqslant 1$:

$$\frac{v_m^{k+1} - v_m^k}{\tau} = \\ = a^2\left[\theta\frac{v_{m-1}^k - 2v_m^k + v_{m+1}^k}{h^2} + (1-\theta)\frac{v_{m-1}^{k+1} - 2v_m^{k+1} + v_{m+1}^{k+1}}{h^2}\right] + f_m^{k+\frac{1}{2}}. \tag{13.77}$$

6. Дослідіть властивості різницевої схеми Річардсона:

$$\frac{v_m^{k+1} - v_m^{k-1}}{2\tau} = a^2\,\frac{v_{m+1}^k - 2v_m^k + v_{m-1}^k}{h^2} + f_m^k. \tag{13.78}$$

7. Дослідіть властивості різницевої схеми Дюфорта–Франкела:

$$\frac{v_m^{k+1} - v_m^{k-1}}{2\tau} = a^2\,\frac{v_{m+1}^k - v_m^{k+1} - v_m^{k-1} + v_{m-1}^k}{h^2} + f_m^k. \tag{13.79}$$

8. За яких значень параметра θ різницева схема (13.77), що апроксимує задачу Коші для рівняння теплопровідності, задовольняє спектральну ознаку стійкості Неймана за будь-якого $\sigma^2 = a^2\,\tau/h^2$?

9. Визначте, чи стійка різницева схема Річардсона (13.78).

10. Визначте, чи стійка різницева схема Дюфорта–Франкела (13.79).

Розділ 14

Методи розв'язання гіперболічних рівнянь

- Явні та неявні різницеві схеми
- Стійкість і збіжність різницевих схем
- Методи розв'язання мішаних задач
- Методи розв'язання систем гіперболічних рівнянь

У цьому розділі основна увага приділяється побудові та дослідженню властивостей різницевих методів розв'язання рівнянь гіперболічного типу. На сьогодні крім них інтенсивно розвиваються також наближені методи, які зводять задачу розв'язання рівнянь цього виду до задачі Коші для звичайних диференціальних рівнянь [8, 28, 34]. Одним із найбільш ефективних методів такого типу є метод характеристик, що широко застосовується на практиці для розв'язання рівнянь переносу і хвильових рівнянь. Однією з головних переваг цього методу є те, що для його реалізації можна використовувати математичне і програмне забезпечення, що розроблено для розв'язання звичайних диференціальних рівнянь.

14.1. Рівняння переносу

Існує багато задач, що стосуються розповсюдження частин у речовині: перенесення домішок потоком рідини або газу, визначення нейтронних і теплових потоків у реакторі, зумовлених дифузією нейтронів і електронів, та ін. Як приклад розглянемо найпростіше одновимірне лінійне рівняння переносу:

$$\frac{\partial u}{\partial t} + a(x,t)\frac{\partial u}{\partial x} = f(x,t). \tag{14.1}$$

За його допомогою можна пояснити основні ідеї методів розв'язання таких рівнянь. Якщо припустити, що $a = const$ і функція $f(x, y) = 0$, то загальний розв'язок цього рівняння матиме вигляд біжучої хвилі:

$$u(x,t) = \varphi(x - at). \tag{14.2}$$

Звідси видно, що параметр a характеризує швидкість переносу. Якщо $a > 0$, то хвиля рухається вправо.

Розглянемо просте рівняння з частинними похідними

$$\frac{dx}{dt} = a,$$

розв'язком якого є ряд ліній (у даному випадку прямих), що описуються рівнянням $x - \alpha t = C$, де C – довільна константа. Рівняння переносу (14.1) уздовж кожної прямої можна записати як звичайне диференціальне рівняння:

$$\left.\frac{du}{dt}\right|_C = f(x,t)|_C, \tag{14.3}$$

де

$$\left.\frac{du}{dt}\right|_C = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x}\frac{dx}{dt} = \frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x}$$

являє собою так звану похідну за напрямком, що задається диференціальним рівнянням $dx/dt = a$. Лінії, уздовж яких рівняння з частинними похідними (14.1) перетворюється у звичайне, називаються характеристиками. Наявність дійсних характеристик є ознакою рівняння гіперболічного типу. Рівняння (14.1) для $a = const$, має ряд прямолінійних характеристик $x = at + const$, що заповнюють усю площину (рис. 14.1).

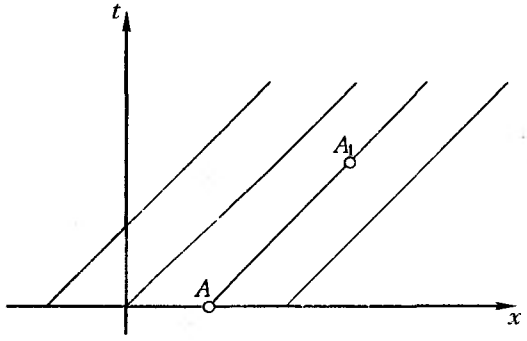


Рис. 14.1. Характеристики рівняння (14.1) для a = *const*

На кожній характеристиці виконується умова сумісності $u = C$, де C – константа, яка змінюється від одної характеристики до іншої. Наприклад, значення функції в точці A_1 дорівнює значенню функції в точці A, тобто $u_{A1} = u_A$, оскільки вони знаходяться на одній характеристиці (рис. 14.1). Відзначені властивості дозволяють легко побудувати розв'язок рівняння (14.1).

Нехай треба знайти розв'язки рівняння (14.1) для $a = const > 0$ у прямокутній області $0 \leqslant t \leqslant T$, $0 \leqslant x \leqslant l$.

Щоб знайти розв'язок рівняння переносу в будь-якій внутрішній точці B (рис. 14.2), можна, провівши через цю точку характеристику, розв'язати задачу Коші для рівняння (14.3).

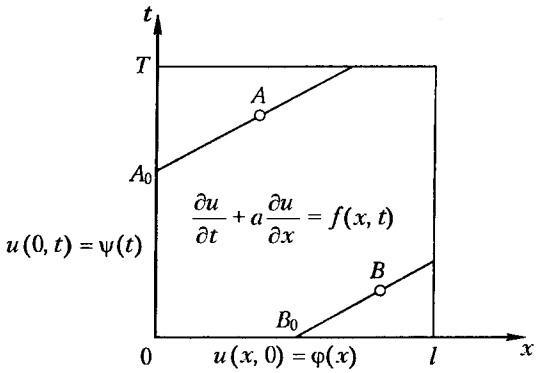


Рис. 14.2. Область розв'язку рівняння переносу

Якщо взяти до уваги, що t може лише зростати, то для знаходження єдиного розв'язку в точці B необхідно задати початкові умови в точці B_0. Так само для пошуку розв'язку в точці A необхідно задати початкові умови в точці A_0. Отже, щоб знайти розв'язок у всіх внутрішніх точках, необхідно задати як граничні умови на відрізку $[0, l]$ осі $0x$, так і початкові умови на відрізку $[0, T]$ осі $0t$. Математична постановка задачі для цього випадку має такий вигляд:

$$\begin{cases} \dfrac{\partial u}{\partial t} + a(x,t)\dfrac{\partial u}{\partial x} = f(x,t), & 0 < x \leqslant l, t > 0, \\ u(x,0) = \varphi(x), & 0 \leqslant x \leqslant l, \\ u(0,t) = \psi(t), & 0 < t. \end{cases} \tag{14.4}$$

Очевидно, якщо $a < 0$, треба було б задавати граничні умови не на лівій межі осі $0t$, а на правій.

14.1.1. Різницеві схеми для рівняння переносу

Розглянемо використання методу сіток для чисельного розв'язання задачі (14.4). Уведемо в області визначення розв'язку множину вузлів сітки $\omega^h = \{x_m, t_k\}$ із кроком $h = l/M$ на осі $0x$ і $\tau = T/K$ по осі $0t$ і проведемо лінії $x_m = const$ і $t_k = const$. Для заміни диференціального рівняння (14.1) різницевим виберемо шаблон, наведений на рис. 14.3, *а*. Різницева схема, що відповідає цьому шаблону, є явною:

$$\frac{v_m^{k+1} - v_m^k}{\tau} + a\frac{v_{m+1}^k - v_m^k}{h} = f_m^k. \tag{14.5}$$

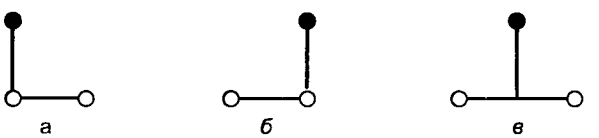


Рис. 14.3. Шаблони явних різницевих схем

Оскільки для апроксимації похідних використовуються односторонні несиметричні різниці, отримаємо різницеву схему першого порядку апроксимації за h і τ. Вираз (14.5) дозволяє одержати розв'язок рівняння на $(k+1)$-му шарі у вигляді:

$$v_m^{k+1} = -\sigma v_{m+1}^k + (1+\sigma)v_m^k + \tau f_m^k, \quad \sigma = a\tau/h. \tag{14.6}$$

Проаналізуємо стійкість цієї схеми (14.6) за спектральною ознакою. Припустимо, що $a > 0$ і $\sigma > 0$. Розв'язок рівняння будемо шукати у вигляді $v_m^k = \lambda^k e^{ima}$. Підставивши його в (14.6) і припустивши, що $f_m^k = 0$, після скорочення на $\lambda^k e^{ima}$ одержимо:

$$\lambda = -\sigma e^{ia} + 1 + \sigma. \tag{14.7}$$

Вираз (14.7) описує на комплексній площині коло з центром у точці $1+\sigma$ і радіусом σ (рис. 14.4).

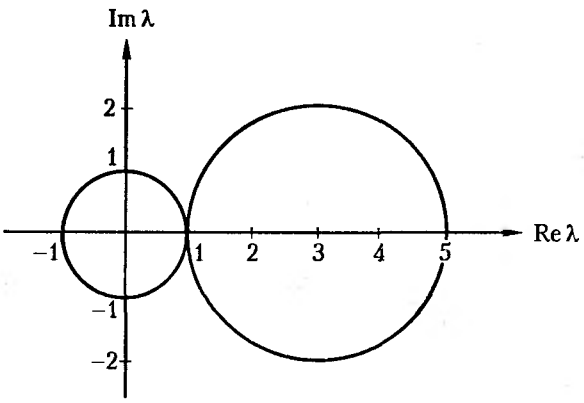


Рис. 14.4. Годограф спектра різницевої схеми (14.6)

Коло з центром у точці (3,0), зображене на рис. 14.4, є спектром λ диференціального рівняння. Коло з центром на початку координат і радіусом, що дорівнює одиниці, відповідає області значень, для яких $|\lambda| \leqslant 1$, тобто області стійкості. На рис. 14.4 видно, що кола не перетинаються, тобто схема (14.5) абсолютно нестійка.

Перейдемо до розгляду різницевої схеми, побудованої за шаблоном рис. 14.3, *б*:

$$\frac{v_m^{k+1} - v_m^k}{\tau} + a\frac{v_m^k - v_{m-1}^k}{h} = 0. \tag{14.8}$$

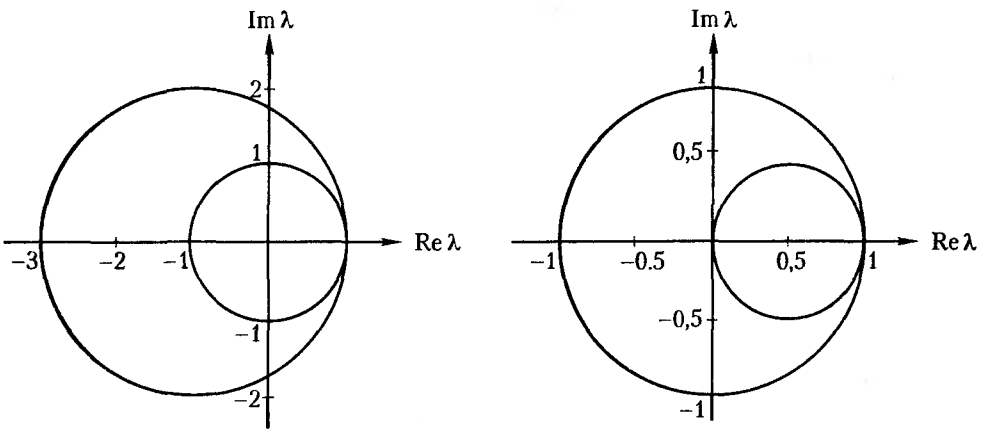
Розв'язок рівняння на $(k+1)$-му шарі буде записано в такому вигляді:

$$v_m^{k+1} = (1-\sigma)v_m^k + \sigma v_{m-1}^k + f_m^k, \quad \sigma = a\tau/h > 0. \tag{14.9}$$

Підставивши в рівняння (14.9) розв'язок у вигляді $v_m^k = \lambda^k e^{ima}$, скоротивши на $\lambda^k e^{ima}$, одержимо:

$$\lambda = \sigma e^{-ia} + 1 - \sigma. \tag{14.10}$$

Вираз (14.10) описує коло з центром у точці $1-\sigma$ і радіусом σ на комплексній площині (рис. 14.5).



а *б*

Рис. 14.5. Годограф спектра для різницевої схеми (14.9): *а* — $\sigma = 2$; *б* — $\sigma = 0{,}5$

Для $\sigma = 2$ одиничне коло міститься всередині годографа спектра λ (рис. 14.5, *а*), тому дана різницева схема (14.8) є нестійкою. Для $\sigma = 0{,}5$ годограф спектра λ буде розташований усередині одиничного кола (рис. 14.5, *б*), тому відповідна різницева схема (14.8) є стійкою. Легко переконатися в тому, що різницева схема (14.8) стійка за умови $\sigma \leqslant 1$, тобто для $\tau \leqslant h/a$.

Послідовність обчислень, які необхідно виконати під час розв'язання рівняння за допомогою умовно стійкої різницевої схеми (14.8), така ж, як і для розв'язання за явною схемою одновимірного параболічного рівняння (див. підрозділ 13.2.1). У цьому разі спочатку знаходять значення сіткової функції $v_m^0 = \varphi_m$, $m = 0, ..., M$ у всіх вузлах початкового шару, а потім, використовуючи граничні умови (14.4), — значення у вузлі, що лежить на лівій межі $v_0^{k+1} = \psi^{k+1}$. І, нарешті, за формулою (14.9) обчислюються значення в інших вузлах $(k+1)$-го часового шару.

Приклад 14.1

Визначимо порядок апроксимації та стійкість різницевої схеми Лакса (рис. 14.6):

$$\frac{v_m^{k+1} - (v_{m+1}^k - v_{m-1}^k)/2}{\tau} + a\frac{v_{m+1}^k - v_{m-1}^k}{2h} = 0 \tag{14.11}$$

для рівняння

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0. \tag{14.12}$$

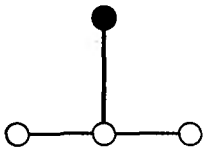


Рис. 14.6. Шаблон різницевої схеми Лакса

Знайдемо нев'язку

$$r_m^k = \frac{u_m^{k+1} - (u_{m+1}^k + u_{m-1}^k)/2}{\tau} + a\frac{u_{m+1}^k - u_{m-1}^k}{2h}. \tag{14.13}$$

Розкладемо в ряди Тейлора різниці, що входять до виразу для нев'язки:

$$(u_{m+1}^k + u_{m-1}^k)/2 = u_m^k + \frac{h^2}{2}\frac{\partial^2 u_m^k}{\partial x^2} + O(h^4),$$

$$\frac{u_m^{k+1} - (u_{m+1}^k + u_{m-1}^k)/2}{\tau} = \frac{\partial u_m^k}{\partial t} + O(\tau) + \frac{h^2}{2\tau}\frac{\partial^2 u_m^k}{\partial x^2} + O(h^3/\tau),$$

$$\frac{u_{m+1}^k - u_{m-1}^k}{2h} = \frac{\partial u_m^k}{\partial x} + O(h^2).$$

Підставляючи отримані співвідношення в (14.13), маємо

$$r_m^k = \frac{\partial u_m^k}{\partial t} + O(\tau) + O\left(\frac{h^2}{2\tau}\right) + a\frac{\partial u_m^k}{\partial x} + O(h^2).$$

Враховуючи, що u_m^k є точним розв'язком рівняння (14.12), отримаємо оцінку для порядку нев'язки:

$$r_m^k = O\left(\tau + h^2 + \frac{h^2}{\tau}\right). \tag{14.14}$$

Приклад 14.2

Розв'яжемо мішану задачу:

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0,$$
$$u(x, 0) = 2x(L - x), \quad 0 \leqslant x \leqslant L, \tag{14.15}$$
$$u(0, t) = 0, \quad t > 0$$

для $a = 0{,}5$, $L = 1$, використовуючи стандартний оператор NDSolve. Так само, як і в прикладі 13.6, отримуємо функцію, що інтерполює наближений розв'язок:

```
In[]:= L = 1; a = 0.5;
       F = NDSolve[{D[u[x,t],t] + a D[u[x,t],x] == 0,
          u[x,0] == 2x (L-x), u[0,t] == 0}, u, {x,0,L}, {t,0,2.5}];
Out[]= {{u → InterpolatingFunction[{{0..1.},{0..2.5}},<>]}}
```

Таким чином, отримано інтерполяційний поліном, який є наближенням до чисельного розв'язку задачі. Графічно його можна зобразити у вигляді поверхні (рис. 14.7).

```
In[] = Gr = Plot3D[Evaluate[u[x,t] /. F[[1]]], {x,0,L}, {t,0,2.5},
          AxesLabel → {"x","t","U(x,t)"}]
```

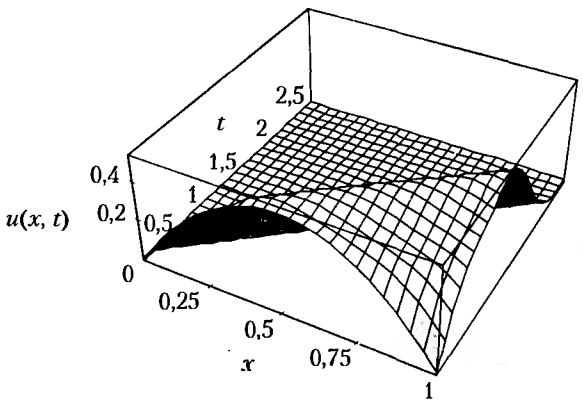


Рис. 14.7. Поверхня, що апроксимує чисельний розв'язок задачі

Графіки розв'язку зображено як функції змінної x для фіксованих моментів часу (рис. 14.8, *а*) та як функції змінної часу, що задані у фіксованих точках простору (рис. 14.8, *б*). На правому графіку видно, як хвиля початкового збурення розповсюджується вправо зі швидкістю, що дорівнює a.

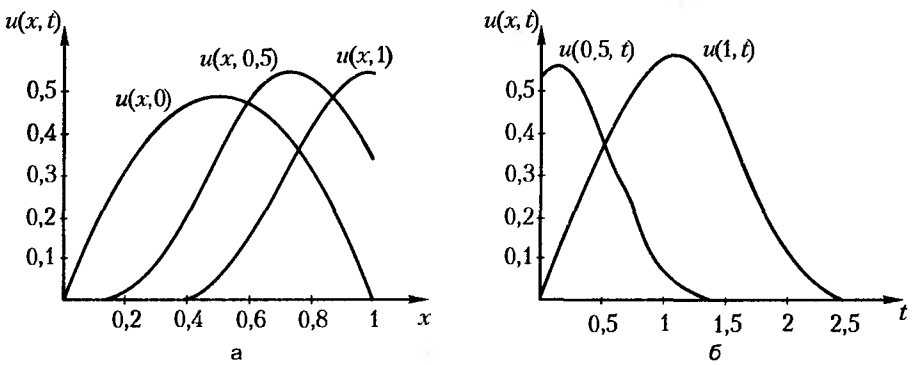


Рис. 14.8. Лінії перетину поверхні розв'язку: *а* — площиною, перпендикулярною до осі 0x; *б* — площиною, перпендикулярною до осі 0t

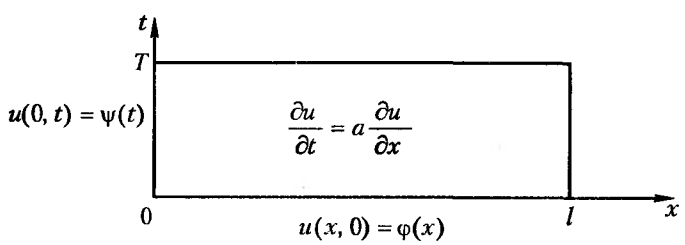


Рис. 14.9. Область розв'язку мішаної задачі для хвильового рівняння

14.2. Різницеві схеми для хвильового рівняння

Розглянемо різницеві методи розв'язання лінійних диференціальних рівнянь другого порядку гіперболічного типу. Такими рівняннями описується багато фізичних явищ і процесів, наприклад коливання струни і подовжні коливання пружного стрижня, рух стиснутого газу, розповсюдження електромагнітних хвиль і багато інших.

Типовим прикладом рівняння з постійними коефіцієнтами гіперболічного типу є хвильове рівняння:

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2} + f(x,t), \quad 0 < x \leqslant l, t > 0 \tag{14.16}$$

з початковими умовами

$$u(x,0) = \varphi(x), \quad \frac{\partial u(x,0)}{\partial t} = \psi(x), \quad 0 \leqslant x \leqslant l. \tag{14.17}$$

Відзначимо, що на відміну від рівнянь параболічного типу (13.2) гіперболічні рівняння вимагають задавання двох початкових умов. Наприклад, для струни необхідно задати початкове зміщення струни $u(x,0)$ і початкові швидкості в кожній її точці $\partial u(x,0)/\partial t$. Граничні умови, як і для параболічних рівнянь, можуть бути одного з трьох родів. Нижче вказані граничні умови першого роду:

$$u(0,t) = \xi(t), \quad u(l,t) = \phi(t), \quad 0 < t. \tag{14.18}$$

У разі коливань струни умови (14.17) визначають закони руху її кінців. Метод сіток для рівняння гіперболічного типу має багато спільного з методом сіток для рівнянь еліптичного і параболічного типів.

14.2.1. Явна різницева схема

Розглянемо методи чисельного розв'язання задачі (14.16), (14.17), (14.18). Уведемо в розрахункову область множину вузлів сітки $\omega^h = \{x_m, t_k\}$, виберемо крок $h = l/M$ уздовж осі $0x$ і $\tau = T/K$ вздовж осі $0t$ та проведемо лінії $x_m = const$ і $t_k = const$. Замінимо диференціальне рівняння (14.16) різницевим за шаблоном «хрест» (рис. 13.3, *б*):

$$\frac{v_m^{k+1} - 2v_m^k + v_m^{k-1}}{\tau^2} = a^2 \frac{v_{m-1}^k - 2v_m^k + v_{m+1}^k}{h^2} = f_m^k, \tag{14.19}$$

яке апроксимує хвильове рівняння з порядками точності $O(\tau^2 + h^2)$.

Позначивши

$$\sigma^2 = \frac{a^2\tau^2}{h^2}, \tag{14.20}$$

перетворимо (14.19) на просту явну схему:

$$v_m^{k+1} = \sigma^2 v_{m-1}^k + 2(1-\sigma^2)v_m^k + \sigma^2 v_{m-1}^k - v_m^{k-1} + \tau^2 f_m^k. \qquad (14.21)$$

Розв'язок $v_m^0 = \varphi_m$, $m = 0, ..., M$ на нульовому часовому шарі відомий з початкової умови (14.17). На першому шарі наближений розв'язок також можна обчислити за початковими умовами. Найпростіший спосіб полягає в заміні похідної лівою скінченною різницею, тобто

$$\left.\frac{\partial u(x_m, t)}{\partial t}\right|_{t=0} = \frac{u_m^1 - u_m^0}{\tau} + O(\tau).$$

Звідси отримаємо значення чисельного розв'язку на першому шарі:

$$v_m^1 = v_m^0 + \tau\psi_m = \varphi_m + \tau\psi_m, \quad m = 1, ..., M-1. \qquad (14.22)$$

Оскільки в другій умові (14.17) похідну за t замінили з похибкою порядку τ, рівняння (14.22) апроксимує другу умову (14.17) з тим же порядком. Граничні умови першого роду для прямокутної області апроксимуються без похибки:

$$v_0^k = \xi^k, \quad v_M^k = \phi^k, \quad k = 1, ..., K. \qquad (14.23)$$

Отже, різницева схема (14.19), (14.21), (14.22) і (14.23) апроксимує вихідну крайову задачу (14.16), (14.17) і (14.18) із порядком лише $O(\tau + h^2)$. Її порядок апроксимації можна підвищити. Для цього слід увести фіктивний часовий шар із вузлами $\{x_m, -\tau\}$, $m = 1, ..., M-1$, що лежатиме за межами області сітки. Значення сіткової функції на цьому шарі позначимо через v_m^{-1}, $m = 1, ..., M-1$. Для апроксимації похідної за часом у вузлах нульового часового шару використаємо центральну симетричну різницю:

$$\left.\frac{\partial u(x_m, t)}{\partial t}\right|_{t=0} = \frac{u(x_m, t_1) - u(x_m, t_{-1})}{2\tau} + O(\tau^2).$$

Тоді різницеве рівняння для другої умови (14.17) запишемо у вигляді

$$\frac{v_m^1 - v_m^{-1}}{2\tau} = \psi_m,$$

звідки $v_m^{-1} = v_m^1 - 2\tau\psi_m$. Щоб виключити значення функції у фіктивному вузлі v_m^{-1}, використаємо різницеве рівняння (14.21), записавши його на нульовому шарі для $k = 0$:

$$v_m^1 = \sigma^2 v_{m-1}^0 + 2(1-\sigma^2)v_m^0 + \sigma^2 v_{m-1}^0 - v_m^{-1} + \tau^2 f_m^0.$$

Виключивши з останнього рівняння v_m^{-1} і використавши початкову умову (14.22), отримаємо явну формулу для обчислення наближеного розв'язку на першому часовому шарі:

$$v_m^1 = 0,5[\sigma^2\varphi_{m-1} + 2(1-\sigma^2)\varphi_m + \sigma^2\varphi_{m-1}] + \tau\psi_m + 0,5\tau^2 f_m^0. \qquad (14.24)$$

Із умовою (14.24) різницева схема (14.19), (14.22), (14.23) апроксимує початкову крайову задачу (14.16), (14.17) і (14.18) з порядком $O(\tau^2 + h^2)$. Обчислення наближеного розв'язку починаються з першого нульового шару за формулою (14.24), а потім виконуються послідовно для наступних часових шарів за формулою (14.21).

14.2.2. Стійкість явної різницевої схеми

Стійкість явної різницевої схеми будемо досліджувати за допомогою спектральних методів. Розв'язок у вигляді $v_m^k = \lambda^k e^{ima}$ підставимо в рівняння (14.21) і, припускаючи, що $f_m^k = 0$, отримаємо після скорочення на $\lambda^{k-1} e^{ima}$

$$\begin{aligned}\lambda^2 &= \lambda\sigma^2 e^{-ia} + 2\lambda(1-\sigma^2) + \lambda\sigma^2 e^{ia} - 1 = \\ &= \lambda\sigma^2(e^{-ia} + e^{ia}) + 2\lambda(1-\sigma^2) - 1 = 2\lambda\sigma^2\cos a + 2\lambda(1-\sigma^2) - 1 = \\ &= 2\lambda - 2\lambda\sigma^2(1-\cos\alpha) - 1 = 2\lambda - 4\lambda\sigma^2\sin^2 a - 1.\end{aligned}$$

Звідси отримаємо рівняння для годографа спектра λ:

$$\lambda^2 - 2\lambda(1 - 2\sigma^2\sin^2 a) + 1 = 0. \qquad (14.25)$$

За теоремою Вієта добуток його коренів $\lambda_1\lambda_2 = 1$. Якщо корені рівняння дійсні й один з них, наприклад, $|\lambda_1| < 1$, то другий має бути $|\lambda_2| > 1$. Тому для виконання умови стійкості $|\lambda_1| < 1$ корені рівняння з дійсними коефіцієнтами (14.25) повинні бути комплексно-спряженими, тобто дискримінант рівняння не повинен бути додатним:

$$D = (1 - 2\sigma^2\sin^2 a)^2 - 1 \leqslant 0.$$

Звідки отримаємо нерівність:

$$|1 - 2\sigma^2\sin^2 a| \leqslant 1.$$

Щоб нерівність виконувалася для будь-яких a, необхідно і достатньо виконання умови Куранта:

$$\sigma^2 < 1, \quad \text{тобто} \quad a\tau < h. \qquad (14.26)$$

Отже, різницева схема «хрест» є умовно стійкою.

14.2.3. Збіжність явної різницевої схеми

Із наведених вище викладок випливає, що в разі виконання умови Куранта різницева схема (14.19), (14.22), (14.23) сходиться до точного розв'язку зі швидкістю, пропорційною $O(\tau^2 + h^2)$. Вона забезпечує достатню точність наближеного розв'язку і дозволяє знайти негладкі й навіть розривні розв'язки, хоча в останньому випадку точність результатів невисока.

Умова стійкості (14.26) досить зручна, оскільки для одержання високої точності можна вибирати $a\tau \approx h$. Тому схему «хрест» часто використовують для практичних розрахунків. Безумовно стійкі різницеві схеми для гіперболічних рівнянь існують, але всі вони є неявними.

14.2.4. Неявна різницева схема

Якщо схема є умовно стійкою, то навіть невелике випадкове порушення стійкості може призвести до швидкого наростання похибки. Щоб запобігти такій ситуації та забезпечити надійність обчислень, рекомендують використовувати безумовно стійкі неявні схеми.

Розглянемо неявну схему з ваговими коефіцієнтами, шаблон якої наведений на рис. 14.10:

$$\frac{v_m^{k+1} - 2v_m^k + v_m^{k-1}}{\tau^2} = a^2 \left[\frac{\gamma(v_{m-1}^{k+1} - 2v_m^{k+1} + v_{m+1}^{k+1}) + (1-2\gamma)(v_{m-1}^k - 2v_m^k + v_{m+1}^k)}{h^2} + \right.$$
$$\left. + \frac{\gamma(v_{m-1}^{k-1} - 2v_m^{k-1} + v_{m+1}^{k-1})}{h^2} \right] + f_m^k. \qquad (14.27)$$

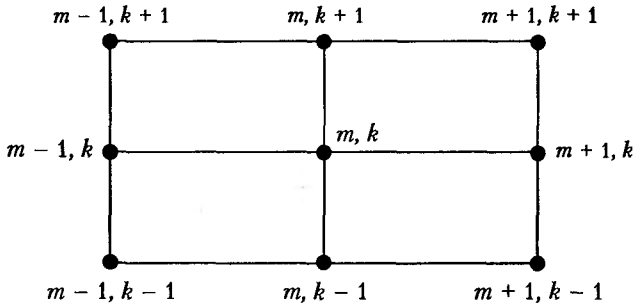


Рис. 14.10. Шаблон неявної різницевої схеми з ваговими коефіцієнтами

Щоб усі вагові коефіцієнти були додатними, варто вибирати $0 \leqslant \gamma \leqslant 0{,}5$. У граничних вузлах розв'язок визначається з умов (14.23).

Дослідимо схему (14.27). Значення розв'язку на нульовому і першому шарах обчислюються за формулами (14.22) і (14.24). На наступних шарах схема (14.27) із граничними умовами (14.23) являє собою систему лінійних рівнянь із тридіагональною матрицею. Розв'язок цієї системи існує, він єдиний і обчислюється методом прогону.

Розкладанням розв'язку за формулою Тейлора можна встановити, що за умови існування неперервної четвертої похідної функції розв'язку схема (14.27) апроксимує рівняння (14.16) з похибкою $O(\tau^2 + h^2)$ для всіх γ.

Стійкість схеми (14.27) перевіряють спектральним методом. Після підстановки $v_m^k = \lambda^k e^{ima}$ у рівняння отримаємо квадратне рівняння стосовно λ:

$$\lambda^2 - \frac{2(1-2(1-2\gamma)\sigma^2 \sin^2 a/2)}{1+4\gamma\sigma^2 \sin^2 a/2}\lambda + 1 = 0.$$

За теоремою Вієта добуток його коренів $\lambda_1\lambda_2 = 1$. На підставі тих же міркувань, що й у підрозділі 14.2.2, можна зробити висновок, що для виконання умови стійкості $|\lambda| < 1$ корені рівняння з дійсними коефіцієнтами (14.25) мають бути комплексно-спряженими, тобто дискримінант рівняння не повинен бути додатним. Звідси випливає умова стійкості схеми (14.27):

$$\sigma^2(1-4\gamma) \leqslant 1. \tag{14.28}$$

Із нерівності (14.28) видно, що коли $\gamma \geqslant 1/4$, схема (14.27) безумовно стійка. Коли $\gamma \leqslant 1/4$, схема є умовно стійкою, якщо

$$\sigma \leqslant \frac{1}{\sqrt{1-4\gamma}} \quad \text{або} \quad \tau \leqslant \frac{h}{a\sqrt{1-4\gamma}}. \tag{14.29}$$

Таким чином, у разі вибору ваги $1/4 \leqslant \gamma \leqslant 1/2$, неявна схема (14.27) є, безумовно, стійкою і має точність $O(\tau^2 + h^2)$. Коли $\gamma = 0$, схема перетворюється на схему «хрест», а умова стійкості (14.28) на умову Куранта $\tau \leqslant h/a$.

Приклад 14.3

Знайдемо розв'язок мішаної задачі для хвильового рівняння, яке описує коливання струни із закріпленими кінцями:

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2}, \quad 0 < x \leqslant L, t > 0$$

з початковими умовами

$$u(x,0) = \begin{cases} x, & \text{для } 0 \leqslant x \leqslant 0{,}6L, \\ 1{,}5(L-x), & \text{для } 0{,}6L \leqslant x \leqslant L, \end{cases}$$

$$\frac{\partial u(x,0)}{\partial t} = 0$$

і граничними умовами

$$u(0,t) = 0, \quad u(L,t) = 0.$$

Розв'яжемо задачу, використовуючи різницеву схему (14.21). Введемо вихідні дані, початкові і граничні умови та отримаємо функцію, яка інтерполює наближений розв'язок:

```
In[]:= m = 20; k = 25; l = 1.0; h = l/(m - 1); a = 1.0;
       Array[w, {k,m}, 0]; τ = h; s = a τ^2/h^2;
       Do [w[0,j] = If [j h < 0.6, N[j h], 1.5 (l - j h)]; w[1, j] = w[0,j], {j, 0, m-1} ];
       Do [w[i,0] = 0; w[i,m-1] = 0, {i, 1, k-1} ];
       Do [ Do [w[i+1,j] = -w[i-1,j] + s w[i,j-1] + 2 (1 - s) w[i,j] + s w[i,j+1],
            {j, 1, m-2} ], {i, 1, k-1} ];
       W = Array[w, {k,m},0]; ListPlot3D[W, AxesLabel →{"m", "k", "u(x,t)"}];
```

Тепер можна побудувати поверхню, що інтерполює наближений розв'язок (рис 14.11).

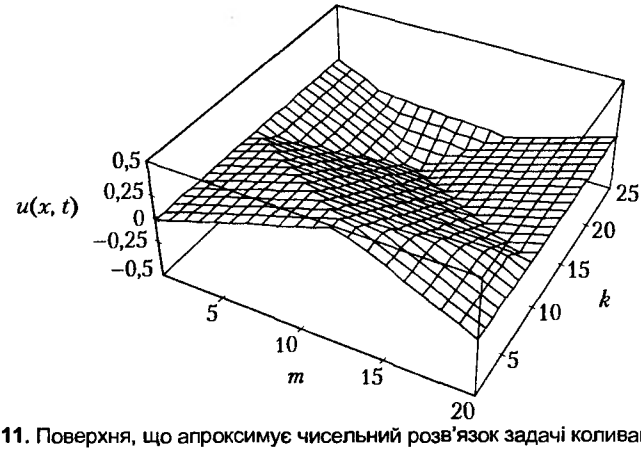


Рис. 14.11. Поверхня, що апроксимує чисельний розв'язок задачі коливань струни

На рис. 14.12 показані графіки розв'язку як функції змінної x для фіксованих моментів часу (рис. 14.12, *а*) та як функції змінної часу, задані у фіксованих точках простору (рис. 14.12, *б*). На правому рисунку видно, як хвиля початкового збурення розповсюджується вправо і вліво зі швидкістю, що дорівнює a, та як вона відображається від кінців відрізка.

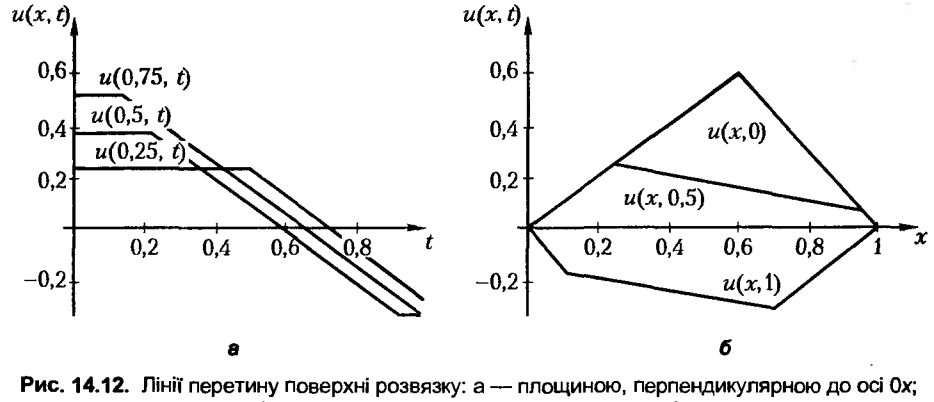


Рис. 14.12. Лінії перетину поверхні розвязку: а — площиною, перпендикулярною до осі 0x; *б* — площиною, перпендикулярною до осі 0*t*

Відмітимо, що стан струни в момент $t = 0$ збігається з функцією $u(x, 0)$.

14.3. Метод характеристик

Метод характеристик тісно пов'язаний із методами розв'язання задачі Коші. Характеристичні поверхні (лінії) визначаються як поверхні (лінії), на яких розв'язок задачі Коші або не існує, або не є єдиним.

У підрозділі 14.1 було введене поняття характеристик для найпростішого двовимірного рівняння переносу. Розглянемо більш складну систему

$$\begin{cases} \dfrac{\partial u_1}{\partial t} + a_1 \dfrac{\partial u_1}{\partial x} = 0, \\ \dfrac{\partial u_2}{\partial t} + a_2 \dfrac{\partial u_2}{\partial x} = 0, \end{cases} \tag{14.30}$$

що складається з двох незалежних рівнянь. Розв'язок першого з них має вигляд $u = f(x - a_1 t)$, другого — $u_2 = g(x - a_2 t)$. Початкові умови для системи задамо на відрізку ab осі $0x$ (для $t = 0$) у вигляді

$$u_1(x, 0) = \varphi(x), \quad u_2(x, 0) = \psi(x), \quad x \in [a, b].$$

На рис. 14.13 на площині (x, t) зображені ті напівсмуги $(t \geqslant 0)$, на яких можна задати значення $u_1(x, t)$, $u_2(x, t)$. Для наочності вибрані різні знаки коефіцієнтів $a_1 > 0$, $a_2 < 0$.

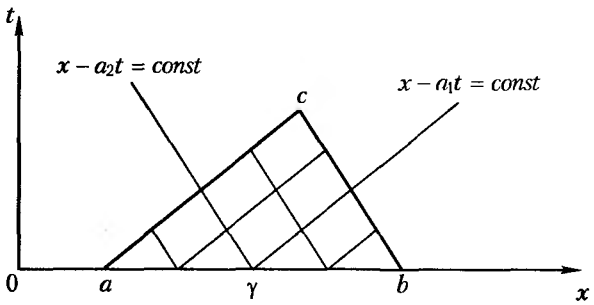


Рис. 14.13. Характеристики системи рівнянь 14.30

Зрозуміло, що розв'язок системи може бути однозначно визначений тільки всередині трикутника abc, що є областю перетину обох смуг, які спираються на відрізок ab. Прямі

$$x - a_1 t = const,$$
$$x - a_2 t = const$$

називаються характеристиками системи, а трикутник abc, обмежений характеристиками, — характеристичним трикутником. Цей простий приклад дозволяє пояснити введені вище визначення і полегшує розуміння основної задачі.

Розгляд методу характеристик почнемо з квазілінійної системи диференціальних рівнянь першого порядку з двома незалежними змінними:

$$\begin{cases} a_{11}\dfrac{\partial u_1}{\partial t} + a_{12}\dfrac{\partial u_2}{\partial t} + b_{11}\dfrac{\partial u_1}{\partial x} + b_{12}\dfrac{\partial u_2}{\partial x} = f_1, \\ a_{21}\dfrac{\partial u_1}{\partial t} + a_{22}\dfrac{\partial u_2}{\partial t} + b_{21}\dfrac{\partial u_1}{\partial x} + b_{22}\dfrac{\partial u_2}{\partial x} = f_2, \end{cases} \tag{14.31}$$

де $a_{ij}(x,t)$, $b_{ij}(x,t)$, $f_i(u_1, u_2, x, t)$, $i = 1, 2$, $j = 1, 2$.

Систему (14.31) можна записати в матричній формі

$$\mathbf{A}\frac{\partial \mathbf{U}}{\partial t} + \mathbf{B}\frac{\partial \mathbf{U}}{\partial x} = \mathbf{F}, \tag{14.32}$$

де введені позначення

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}, \quad \mathbf{U} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}.$$

Припустимо, що система має гладкий розв'язок у деякій області G. Вибравши в цій області точку (x_0, t_0), проведемо через неї криву. Вектор нескінченно малого зсуву вздовж цієї кривої з точки (x_0, t_0) позначимо через (dx, dt). Припустимо, що значення $\mathbf{U}$ вздовж кривої γ відомі і що за цими значеннями і за рівняннями системи треба знайти значення $\mathbf{U}$ в деякому околі γ.

Задача знаходження розв'язку системи в околі кривої γ за значеннями цього розв'язку на кривій називається задачею Коші для системи. Звузимо поставлену задачу, а саме обмежимося пошуком лише похідної вектор-функції $\mathbf{U} = (u_1, u_2)$ за нормаллю до кривої γ у точці (x_0, t_0), що лежить на цій кривій.

Оскільки u_1, u_2 уздовж кривої відомі, а отже, відомі похідні від них уздовж кривої, то, знаючи нормальні похідні, можна обчислити похідні за будь-яким напрямком, у тому числі й похідні

$$\frac{\partial u_1}{\partial t}, \frac{\partial u_2}{\partial t}, \frac{\partial u_1}{\partial x}, \frac{\partial u_2}{\partial x}$$

у точках кривої. І навпаки, знання цих чотирьох похідних дозволяє обчислити похідні за будь-яким напрямком, у тому числі й за напрямком нормалі до кривої. Отже, можна сформулювати таку задачу: знаючи уздовж кривої γ значення вектор-функції $\mathbf{U}$, знайти в точках цієї кривої похідні $\partial\mathbf{U}/\partial t$, $\partial\mathbf{U}/\partial x$. Обчислювати ці похідні будемо за значенням диференціала $d\mathbf{U}$, що відповідає зсуву dt, dx уздовж кривої. Запишемо $d\mathbf{U}$ за допомогою частинних похідних від $\mathbf{U}$:

$$\begin{cases} dt\dfrac{\partial u_1}{\partial t} + dx\dfrac{\partial u_1}{\partial x} = du_1, \\ dt\dfrac{\partial u_2}{\partial t} + dx\dfrac{\partial u_2}{\partial x} = du_2. \end{cases}$$

У цих рівняннях підкреслені відомі диференціали, що визначають зсув уздовж γ. Поєднуючи ці два рівняння з рівняннями системи (14.31), отримаємо чотири лінійних рівняння з чотирма невідомими $\partial u_1/\partial t$, $\partial u_2/\partial t$, $\partial u_1/\partial x$, $\partial u_2/\partial x$:

$$\begin{cases} a_{11}\dfrac{\partial u_1}{\partial t} + a_{12}\dfrac{\partial u_2}{\partial t} + b_{11}\dfrac{\partial u_1}{\partial x} + b_{12}\dfrac{\partial u_2}{\partial x} = f_1, \\ a_{21}\dfrac{\partial u_1}{\partial t} + a_{22}\dfrac{\partial u_2}{\partial t} + b_{21}\dfrac{\partial u_1}{\partial x} + b_{22}\dfrac{\partial u_2}{\partial x} = f_2, \\ dt\dfrac{\partial u_1}{\partial t} + dx\dfrac{\partial u_1}{\partial x} = du_1, \\ dt\dfrac{\partial u_2}{\partial t} + dx\dfrac{\partial u_2}{\partial x} = du_2. \end{cases} \tag{14.33}$$

У матричній формі рівняння для $\partial\mathbf{U}/\partial t$, $\partial\mathbf{U}/\partial x$ мають вигляд

$$\begin{cases} \mathbf{A}\dfrac{\partial \mathbf{U}}{\partial t} + \mathbf{B}\dfrac{\partial \mathbf{U}}{\partial x} = \mathbf{F}, \\ dt\,\mathbf{E}\dfrac{\partial U}{\partial t} + dx\,\mathbf{E}\dfrac{\partial \mathbf{U}}{\partial x} = d\mathbf{U}, \end{cases} \tag{14.34}$$

де $\mathbf{E}$ — одинична матриця. З цих рівнянь шукані похідні можуть бути знайдені за умови

$$\det\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ dt\,\mathbf{E} & dx\,\mathbf{E} \end{pmatrix} = \begin{vmatrix} a_{11} & a_{12} & b_{11} & b_{12} \\ a_{21} & a_{22} & b_{21} & b_{22} \\ dt & 0 & dx & 0 \\ 0 & dt & 0 & dx \end{vmatrix} \neq 0.$$

Лінії, що задаються диференціалами dx, dt, уздовж яких визначник

$$\begin{vmatrix} a_{11} & a_{12} & b_{11} & b_{12} \\ a_{21} & a_{22} & b_{21} & b_{22} \\ dt & 0 & dx & 0 \\ 0 & dt & 0 & dx \end{vmatrix} = 0, \tag{14.35}$$

називаються характеристиками системи

$$\mathbf{A}\frac{\partial \mathbf{U}}{\partial t} + \mathbf{B}\frac{\partial \mathbf{U}}{\partial x} = \mathbf{F}.$$

Розкриємо визначник (14.35) і отримаємо рівняння для обчислення кутових коефіцієнтів дотичних до відповідних характеристик:

$$a\lambda^2 + b\lambda + c = 0, \tag{14.36}$$

де

$$\lambda = \frac{dx}{dt}, \quad a = |\mathbf{A}|, \quad b = |\mathbf{B}|, \quad c = \begin{vmatrix} a_{12} & b_{11} \\ a_{22} & b_{12} \end{vmatrix} - \begin{vmatrix} a_{11} & b_{12} \\ a_{21} & b_{22} \end{vmatrix}.$$

Розв'язавши (14.36), отримаємо два рівняння для характеристик, що проходять через всі точки площини (x, t):

$$\frac{dx}{dt} = \lambda_1 = \frac{-c + \sqrt{c^2 - 4ab}}{2a}, \quad \frac{dx}{dt} = \lambda_2 = \frac{-c - \sqrt{c^2 - 4ab}}{2a}. \tag{14.37}$$

Нехай крива γ є характеристикою. Незважаючи на те, що визначник дорівнює 0, система (14.34) має розв'язок, оскільки за припущенням в області G існує розв'язок системи (14.32), який приймає на кривій γ задані значення. Це означає, що ранг розширеної матриці

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} & \mathbf{F} \\ dt\,\mathbf{E} & dx\,\mathbf{E} & d\mathbf{U} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & b_{11} & b_{12} & f_1 \\ a_{21} & a_{22} & b_{21} & b_{22} & f_2 \\ dt & 0 & dx & 0 & du_1 \\ 0 & dt & 0 & dx & du_2 \end{pmatrix}$$

дорівнює рангу виродженої матриці

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ dt\,\mathbf{E} & dx\,\mathbf{E} \end{pmatrix},$$

створеної з коефіцієнтів за невідомих $\partial\mathbf{U}/\partial t$, $\partial\mathbf{U}/\partial x$. Отже, вектор $d\mathbf{U}$ вздовж характеристики не може бути довільним. Він має задовольняти співвідношення

$$\operatorname{rang}\begin{pmatrix} \mathbf{A} & \mathbf{B} & \mathbf{F} \\ dt\,\mathbf{E} & dx\,\mathbf{E} & d\mathbf{U} \end{pmatrix} = \operatorname{rang}\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ dt\,\mathbf{E} & dx\,\mathbf{E} \end{pmatrix}.$$

Це співвідношення і є диференціальним рівнянням уздовж характеристики. Визначник матриці справа дорівнює нулю. Отже, має дорівнювати нулю і визначник матриці, що стоїть в лівій частині й отриманий з розширеної матриці відкиданням від неї чотирьох довільних стовпців. Тобто повинен дорівнювати нулю нижче наведений визначник.

$$\begin{vmatrix} a_{11} & a_{12} & b_{11} & b_{12} & f_1 \\ a_{21} & a_{22} & b_{21} & b_{22} & f_2 \\ dt & 0 & dx & 0 & du_1 \\ 0 & dt & 0 & dx & du_2 \end{vmatrix} = 0.$$

Він отриманий виключенням із розширеної матриці її четвертого (передостаннього) стовпця:

$$p\,du_1 + (a\lambda_i + q)\,du_2 - g\,dx + r\,dt = 0, \quad i = 1, 2, \tag{14.38}$$

де

$$p = \begin{vmatrix} a_{11} & b_{11} \\ a_{21} & b_{21} \end{vmatrix}, \quad q = \begin{vmatrix} a_{12} & b_{11} \\ a_{22} & b_{21} \end{vmatrix}, \quad g = \begin{vmatrix} a_{11} & f_1 \\ a_{21} & f_2 \end{vmatrix}, \quad r = \begin{vmatrix} b_{11} & f_1 \\ b_{21} & f_2 \end{vmatrix}.$$

Для того, щоб проілюструвати поняття характеристик і співвідношення на них, розглянемо систему, яка описує нестаціонарні процеси, що протікають у довгій електричній лінії.

Приклад 14.4

Система телеграфних рівнянь, що описує електричні коливання в лінії з розподіленими параметрами, має такий вигляд:

$$\begin{cases} L\dfrac{\partial i(x,t)}{\partial t} + \dfrac{\partial u(x,t)}{\partial x} = -Ri(x,t), \\ \dfrac{\partial i(x,t)}{\partial x} + C\dfrac{\partial u(x,t)}{\partial t} = 0, \end{cases} \tag{14.39}$$

де $i(x,t)$ – сила струму, $u(x,t)$ – напруга, R і L опір та індуктивність, розраховані на одиницю довжини. Наведемо матричну форму запису системи рівнянь (14.39):

$$\begin{pmatrix} L & 0 \\ 0 & C \end{pmatrix} \frac{\partial}{\partial t} \begin{pmatrix} i(x,t) \\ u(x,t) \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \frac{\partial}{\partial x} \begin{pmatrix} i(x,t) \\ u(x,t) \end{pmatrix} = \begin{pmatrix} -R & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} i(x,t) \\ u(x,t) \end{pmatrix}.$$

Система (14.33) для даного прикладу матиме такий вигляд:

$$\begin{pmatrix} L & 0 & 0 & 1 \\ 0 & C & 1 & 0 \\ dt & 0 & dx & 0 \\ 0 & dt & 0 & dx \end{pmatrix} \begin{pmatrix} \dfrac{\partial i(x,t)}{\partial t} \\ \dfrac{\partial u(x,t)}{\partial t} \\ \dfrac{\partial i(x,t)}{\partial x} \\ \dfrac{\partial u(x,t)}{\partial x} \end{pmatrix} = \begin{pmatrix} -R & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} i(x,t) \\ 0 \\ di(x,t) \\ du(x,t) \end{pmatrix}.$$

Рівняння характеристик визначається виразом

$$\begin{vmatrix} L & 0 & 0 & 1 \\ 0 & C & 1 & 0 \\ dt & 0 & dx & 0 \\ 0 & dt & 0 & dx \end{vmatrix} = 0.$$

Розкриваючи цей визначник, отримаємо:

$$L\begin{vmatrix} C & 1 & 0 \\ 0 & dx & 0 \\ dt & 0 & dx \end{vmatrix} - \begin{vmatrix} 0 & C & 1 \\ dt & 0 & dx \\ 0 & dt & 0 \end{vmatrix} = LC\,dx^2 - dt^2 = 0.$$

Таким чином, рівняння характеристик мають вигляд:

$$dx = \pm\frac{dt}{\sqrt{LC}}$$

і звідси

$$x \mp \frac{t}{\sqrt{LC}} = const.$$

Перейдемо до розрахунків співвідношень для характеристик розглянутої системи. Для цього необхідно, щоб ранг матриці

$$\begin{pmatrix} L & 0 & 0 & 1 & -RL \\ 0 & C & 1 & 0 & 0 \\ dt & 0 & dx & 0 & di(x,t) \\ 0 & dt & 0 & dx & du(x,t) \end{pmatrix}$$

уздовж характеристики дорівнював рангу матриці, складеної з її перших чотирьох стовпців. Визначник цієї останньої матриці дорівнює нулю. Тому має дорівнювати нулю і визначник матриці, складеної з чотирьох довільних стовпців. Наприклад, повинен дорівнювати нулю визначник

$$\begin{vmatrix} L & 0 & 0 & -RL \\ 0 & C & 1 & 0 \\ dt & 0 & dx & di(x,t) \\ 0 & dt & 0 & du(x,t) \end{vmatrix} = 0,$$

отриманий викиданням із розширеної матриці її четвертого (передостаннього) стовпця. Розкриємо цей визначник:

$$L\begin{vmatrix} C & 1 & 0 \\ 0 & dx & di(x,t) \\ dt & 0 & du(x,t) \end{vmatrix} + Ri(x,t)\begin{vmatrix} 0 & C & 1 \\ dt & 0 & dx \\ 0 & dt & 0 \end{vmatrix} = LC\,du(x,t)\,dx + Li(x,t)\,dt + Ri(x,t)\,dt^2 = 0.$$

Розділимо на dt отримане рівняння:

$$LC\,du(x,t)\frac{dx}{dt} + Ri(x,t)\,dt + Li(x,t) = 0.$$

Отже, вздовж першої характеристики $dx/dt = 1/\sqrt{LC}$ має виконуватися таке співвідношення:

$$\sqrt{LC}\,du(x,t) + Ri(x,t)\,dt + Li(x,t) = 0. \qquad (14.40)$$

Для другої характеристики $dx/dt = -1/\sqrt{LC}$ аналогічно одержуємо співвідношення:

$$-\sqrt{LC}\, di(x,t) + Ri(x,t)dt + Li(x,t) = 0. \qquad (14.41)$$

Таким чином, система диференціальних рівнянь із частинними похідними (14.39) зводиться до еквівалентної системи звичайних диференціальних рівнянь (14.40) і (14.41).

14.3.1. Чисельні методи розв'язання системи гіперболічних диференціальних рівнянь першого порядку

Візьмемо в площині (x, t) дві досить близькі точки 1 з координатами (x_1, t_1) і 2 $-(x_2, t_2)$ (рис. 14.14), що не лежать на одній характеристиці. Нехай у цих точках відомі значення шуканих функцій $(u_1^{(1)}, u_2^{(1)})$ і (u_1^2, u_2^2), що задовольняють систему рівнянь (14.32) гіперболічного типу. Через першу точку проведемо пряму в напрямку першої характеристики (або характеристики першої множини). Через точку 2 проведемо пряму в напрямку другої характеристики (або характеристики другої множини), що виходить із точки 2. Позначимо через $\lambda_1^{(1)}$ кутовий коефіцієнт дотичної до першої характеристики в точці 1, а через $\lambda_2^{(2)}$ — кутовий коефіцієнт дотичної до другої характеристики в точці 2. Ці прямі перетнуться в деякій точці 3. Координати (x_3, t_3) цієї точки є розв'язком системи рівнянь

$$\begin{cases} x_3 - x_1 = \lambda_1^{(1)}(t_3 - t_1), \\ x_3 - x_2 = \lambda_2^{(2)}(t_3 - t_2). \end{cases} \qquad (14.42)$$

Рівняння (14.42) можна отримати з відповідних рівнянь (14.37), які визначають напрямок характеристик, за формулою Ейлера.

Рис. 14.14. Ілюстрація методу Массо

Заміняючи диференціали, що входять у співвідношення на відповідних характеристиках (14.38), кінцевими різницями, отримаємо систему рівнянь, щоб знайти значення $u_1^{(3)}$, $u_2^{(3)}$ у точці 3, які позначимо через u_{13}, u_{23}. Ця система має такий вигляд:

$$\begin{cases} p_1(u_{13} - u_{11}) + (a_1\lambda_1^{(1)} + q_1)(u_{23} - u_{21}) - g_1(x_3 - x_1) + r_1(t_3 - t_1) = 0, \\ p_2(u_{13} - u_{21}) + (a_2\lambda_2^{(2)} + q_2)(u_{23} - u_{22}) - g_2(x_3 - x_2) + r_2(t_3 - t_2) = 0, \end{cases} \qquad (14.43)$$

де a_i, g_i, p_i, q_i, r_i, $i = 1, 2$ — значення визначників (14.36) і (14.38) у точці i. Це наближення може виявитися недостатньо точним, оскільки характеристики, що виходять із точок 1 і 2, були замінені відрізками прямих, у той час як точка 3

має бути точкою перетину, власне кажучи, криволінійних характеристик. Тому може виникнути необхідність в уточненні координат точки 3 і значень u_{13}, u_{23} у цій точці. Для уточнення можна використовувати формулу Ейлера з повторним обчисленням (див. формулу (8.35)).

Розв'язуючи елементарну задачу пошуку координат точки $(x_3, y_3, u_{13}, u_{23})$ за координатами двох відомих точок $(x_1, y_1, u_{11}, u_{21})$ і $(x_2, y_2, u_{12}, u_{22})$, можна отримати чисельний розв'язок для різних задач системи (14.31). Розглянемо деякі з них.

14.3.2. Розв'язання задачі Коші

Ця задача полягає у знаходженні розв'язку системи (14.31), якщо функції u_1, u_2 задані на деякій дузі гладкої кривої γ, яка не має характеристичних напрямків у жодній точці. Опишемо чисельне розв'язання цієї задачі за методом Массо. На дузі кривої вибирається ряд досить близьких точок. На рис. 14.15 ці точки позначені числами 1, 2, 3, 4, 5. По точках 1 і 2 методом Массо, розглянутим вище, знаходимо координати точки 6 і значення u_{16}, u_{26}. Потім по точках 2 і 3 знаходимо координати точки 7, по точках 3 і 4 — координати точки 8, по точках 4 і 5 — координати точки 9. Тепер ряд точок 6, 7, 8, 9 розглядаємо як вихідний і продовжуємо побудову. Процес може тривати доти, поки не буде заповнений «трикутник» acb, в якому сторона ac являє собою ламану лінію, що є деяким наближенням до першої характеристики, яка виходить із точки a, а ламана bc — це наближення до другої характеристики, яка виходить із точки b. Подібну побудову можна виконати і з іншого боку кривої γ. При цьому отримаємо «трикутник» adb, сторони якого є відповідно наближеннями до другої характеристики, що проходить через точку a, і до першої характеристики, яка проходить через точку b.

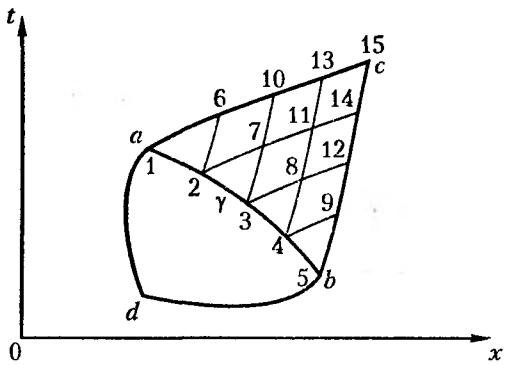


Рис. 14.15. Метод характеристик розв'язання задачі Коші

Таким чином, у явному вигляді визначається область, яка містить наближений розв'язок системи для початкових умов, заданих на кривій ab. Для лінійної системи сітка характеристик може бути заздалегідь побудована, і потрібно буде тільки знаходити значення u_1, u_2 в точках їх перетину, використовуючи співвідношення на характеристиках.

14.3.3. Розв'язання задачі Гурса

Потрібно знайти розв'язок системи (4.31), коли на двох характеристиках ab і ac, що виходять з однієї точки a, задані значення функції u_1, u_2, причому значення відповідних функцій, заданих на характеристиках, збігаються в загальній точці a. Передбачається, що задані функції u_1, u_2 на кожній характеристиці такі, що диференціальні рівняння характеристик задовольняються.

Наведемо опис чисельного розв'язання цієї задачі методом Массо. На дугах характеристик обирається ряд близьких точок (рис. 14.16), тобто точки 1, 2, ..., 8 у нашому випадку. В цих точках значення u_1, u_2 відомі. Так само, як і для задачі Коші, по точках 4 і 5 спочатку знаходимо точку 9 і значення u_{19}, u_{29}.

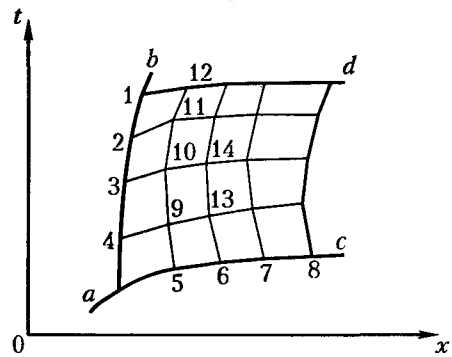


Рис. 14.16. Ілюстрація методу Массо для розв'язання задачі Гурса

По точках 3 і 9 знаходимо точку 10, по точках 2 і 10 — точку 11, по точках 11 і 1 — точку 12. Далі, вважаючи ряд точок 5, 9, 10, 11, 12 за новий ряд, продовжуємо побудову. При цьому заповнюємо елементарними чотирикутниками «чотирикутник», що апроксимує криволінійний чотирикутник, дві сторони якого, ab і ac, є заданими дугами характеристик, а дві інші, $1d$ і $8d$, — побудованими ламаними, що приблизно відповідають характеристикам, які виходять із точок 1 і 8.

14.3.4. Перша мішана задача

Ця задача полягає в побудові розв'язку u_1, u_2 системи (4.31), коли на дузі, що є характеристикою, й на дузі ac, що у жодній точці не має характеристичного напрямку, задані значення u_1, u_2. При цьому передбачається, що в загальній точці a значення відповідних функцій погоджені, і друга характеристика (або характеристика другої множини), яка виходить із точки a, лежить усередині кута bac (рис. 14.17).

Розв'язання першої мішаної задачі зводиться до послідовного розв'язання задачі Коші і задачі Гурса викладеним вище методом, потрібно тільки починати з розв'язання задачі Коші з початковими даними на дузі ac. При цьому зможемо побудувати розв'язок у «трикутнику», обмеженому дугою ac і ламаною, яка апроксимує характеристики різних множин, що виходять із точок a і c. У результаті розв'язання задачі Коші наближено визначиться друга характеристика ad, а також значення u_1, u_2 у вузлах цієї характеристики. Далі розв'язання за-

дачі в області dab зводиться до розв'язання задачі Гурса, оскільки u_1, u_2 будуть відомі на обох характеристиках, що виходять із точки a.

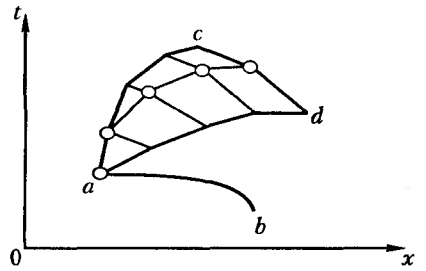


Рис. 14.17. Розв'язання першої мішаної задачі

14.3.5. Друга мішана задача

Ця задача полягає у знаходженні розв'язку системи (14.31), якщо відомі значення u_1 і u_2 на характеристиці ab і відома лінійна комбінація $\alpha u_1 + \beta u_2 = g$ на кривій ac, яка не має характеристичних напрямків, де α, β, g — задані функції. При цьому вважається, що друга характеристика, яка виходить із точки a, лежить поза кутом bac, і у точці a кривої ab виконуються співвідношення $\alpha u + \beta v = g$.

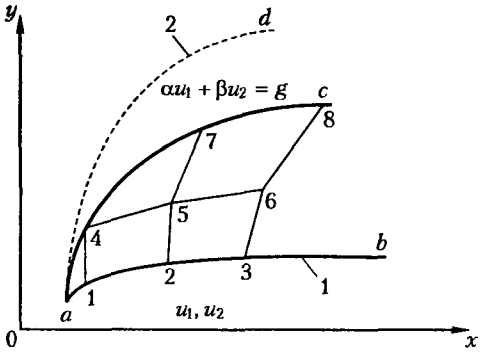


Рис. 14.18. Розв'язання другої мішаної задачі

Розв'яжемо цю задачу в такий спосіб. На дузі характеристики ab оберемо ряд точок 1, 2, 3, ... (рис. 14.18). Із точки 1 у напрямку характеристики другої множини проведемо пряму до перетинання з кривою ac у точці 4. Із диференціальної умови на характеристиці другої множини і граничної умови на кривій ac знаходимо u_4 й v_4 у цій точці. По знайденій точці 4 і точці 2 звичайним шляхом знайдемо точку 5, по точках 5 і 3 — точку 6 і т. д. Ряд точок 4, 5, 6, ... вважаємо за вихідний ряд і повторюємо процес. Таким чином, можна заповнити сітку області, обмеженої кривими ab, ac і характеристикою другої сім'ї, проведеною з точки b до перетинання з кривою ac. Далі розв'язання задачі в області dac зводиться до розв'язання задачі Коші, оскільки u_1, u_2 будуть відомі на прямій ac, що не є характеристикою.

Приклад 14.5

Розглянемо процес відключення електричної лінії з розподіленими параметрами з активним навантаженням $R_н$. Система телеграфних рівнянь, яка описує нестаціонарні коливання в лінії, наведена в прикладі 14.4 і має такий вигляд:

$$\begin{cases} L\dfrac{\partial i(x,t)}{\partial t} + \dfrac{\partial u(x,t)}{\partial x} = -Ri(x,t), \\ \dfrac{\partial i(x,t)}{\partial x} + c\dfrac{\partial u(x,t)}{\partial t} = 0, \quad 0 < x < l, 0 < t \end{cases} \tag{14.44}$$

де $i(x, t)$ — сила струму, $u(x, t)$ — напруга, R, c і L опір, ємність та індуктивність, розраховані на одиницю довжини. До лівого кінця лінії $x = 0$ підключена напруга, що дорівнює u_0. До правого кінця лінії $x = l$ підключене активне навантаження з опором $R_н$, що в момент $t = 0$ відключається.

У початковий момент лінія постійного струму працювала в стаціонарному режимі, тобто початковими умовами є

$$i(x, 0) = \frac{u_0}{Rl + R_н}, \quad u(x, 0) = u_0 - Ri(x,t)x, \quad 0 \leqslant x \leqslant l. \tag{14.45}$$

До лівого кінця лінії $x = 0$ підключається джерело напруги, що дорівнює u_0, а до правого кінця — активне навантаження з опором R, що у момент $t = 0$ відключається. Граничні умови для розглянутого випадку мають такий вигляд:

$$u(0, t) = u_0, \quad i(l, t) = 0. \tag{14.46}$$

Наведемо розв'язання цієї задачі в пакеті Mathematica, відмовившись від спрощення, тобто від припущення, що $R \neq 0$. Скористаємося стандартним оператором NDSolve (див. приклад 13.5).

Введемо початкові дані:

```
In[]:= L = 10; R = 0.025;
       c = 0.2; R1 = 450; T = 6000; l = 1000;
       u0 = 1000; i0 = u0/(R1+lR);
```

Запишемо оператор розв'язання мішаної задачі для системи рівнянь (14.39):

```
In[]:= NDSolve{L*D[i[x,t],t] + D[u[x,t],x] == -R*i[x,t], D[i[x,t],x] + c*D[u[x,t],t] == 0,
               u[x,0] == u0 - i0*R*x,
               i[x,0] == i0,
               u[0,t] == u0,
               i[l,t] == i0 e^(-1000*t)}, {u,i}, {x,0,l}, {t,0,T}
Out[]= {{u → InterpolatingFunction[{{0.,1000.}, {0.,6000.}}, <>],
         i → InterpolatingFunction[{{0.,1000.}, {0.,6000.}}, <>]}}
```

Результатом є функції $u(x, t)$ і $i(x,t)$. Побудуємо поверхню розв'язку $u(x, t)$ (рис. 14.19). Коливання напруги, зумовлені раптовим відключенням навантаження, поширюються до початку лінії, згасаючи, на відміну від коливань, що описані в розв'язанні задачі 14.3. Побудуємо графіки зміни напруги в точках на відстанях 100, 500 і 1000 м від початку лінії.

```
In[]:= Gr = Plot[{U[1000,t], U[500,t], U[100,t]}, {t,0,T},
       PlotRange → {945,1000}, AxesLabel → {"t", "U(x,t)"},
       DisplayFunction → Identity];
       Show[Gr, SH, DisplayFunction → $DisplayFunction]
```

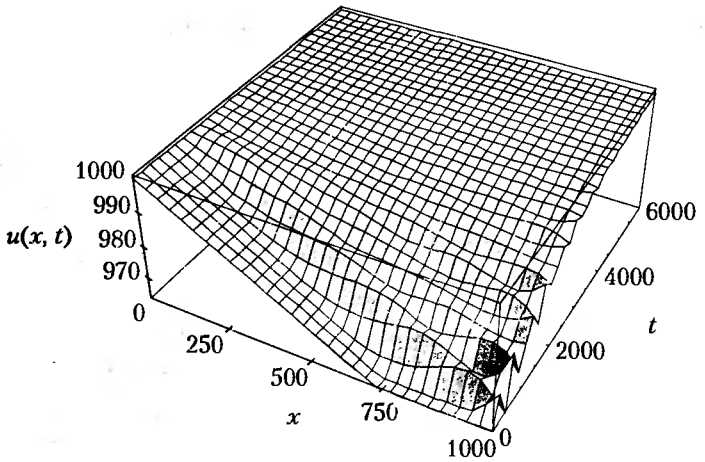


Рис. 14.19. Графік функції розв'язку $u(x,t)$

На рис. 14.20 показано стрибок напруги, зумовлений відключенням навантаження, запізнення в поширенні цього збурення і його згасання. На рис. 14.21 зображені графіки зміни сили струму в точках на відстані 0, 500 і 1000 м від початку, на яких чітко помітні процеси запізнення і згасання коливань. Графіки відображають залежність розв'язків від просторової змінної і часу та ілюструють розповсюдження хвилі початкового збурення і її відбиття від кінців.

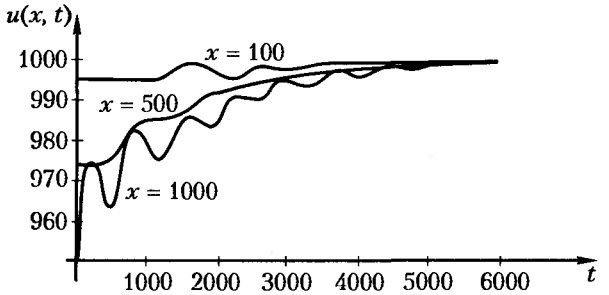


Рис. 14.20. Графіки зміни напруги в точках на відстані 100, 500 і 1000 м від початку лінії

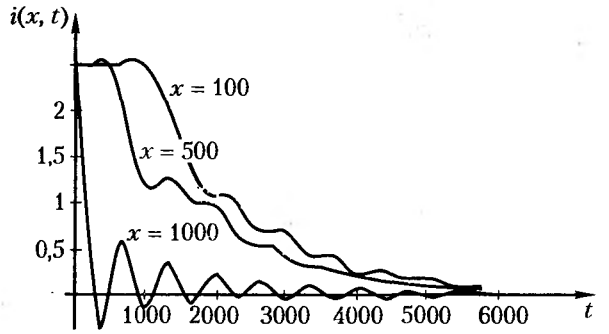


Рис. 14.21. Графіки зміни сили струму в точках на відстані 100, 500 і 1000 м від початку лінії

Висновки

1. Гіперболічні рівняння часто зустрічаються у фізичних задачах, пов'язаних із процесом переносу енергії, розповсюдженням електромагнітних хвиль у провідниках, електромагнітними і акустичними коливаннями в атмосфері, рухом стиснутого газу тощо. Чисельне розв'язання мішаних гіперболічних задач виконується звичайно явними і неявними різницевими методами або методом характеристик. Рідше використовується метод прямих.

2. Стійкість різницевих схем визначається умовами Куранта, які не накладають таких жорстких обмежень на співвідношення між величиною кроків сітки за часом і простором, як для параболічних рівнянь. Тому для рівнянь цього типу неявні різницеві схеми не мають помітної переваги перед явними. Однак більш надійні результати можна отримати, використовуючи безумовно стійкі різницеві схеми. Ці схеми забезпечують достатню точність обчислень для гладких функцій. Вони придатні й для знаходження негладких функцій розв'язку, але точність розрахунків при цьому невелика. Якщо ж схема умовно стійка, то невелике випадкове порушення стійкості може призвести до швидкого і необмеженого наростання похибок.

3. Розв'язок лінійної гіперболічної системи рівнянь може мати розриви, якщо їх мають початкові чи граничні умови. У квазілінійному рівнянні навіть за неперервних і досить гладких початкових умов можуть виникати розриви розв'язків. У разі застосування звичайних різницевих схем для розв'язання таких задач можна отримати розв'язки, які істотно відрізняються від точних. Для розв'язання рівнянь з розривними функціями слід використовувати консервативні різницеві схеми [8, 16].

4. Метод характеристик є найбільш придатним для розв'язання гіперболічних рівнянь. Для одержання гладких розв'язків гіперболічних рівнянь можна використовувати і метод прямих.

Контрольні запитання та завдання

1. Визначте порядок локальної апроксимації диференціального рівняння

$$\frac{\partial u(x,t)}{\partial t} - a\frac{\partial u(x,t)}{\partial x} = 0$$

різницевим рівнянням

$$\frac{v_i^{j+1} - \dfrac{v_{i+1}^j + v_{i-1}^j}{2}}{\tau} - a\frac{v_{i+1}^j - v_{i-1}^j}{2h} = 0.$$

2. Для рівняння

$$\frac{\partial u(x,t)}{\partial t} - a\frac{\partial u(x,t)}{\partial x} = f(x,t), \quad a = const,\ a > 0$$

перевірте стійкість різницевих схем, побудованих за шаблонами рис. 14.3.

3. Для задачі

$$2\frac{\partial u(x,t)}{\partial t} + \frac{\partial u(x,t)}{\partial x} = 0, \quad t > 0,$$
$$u(x,0) = \sin(\pi x), \quad 0 \leqslant x \leqslant 1,$$
$$u(0,t) = t, \quad 0 \leqslant t \leqslant 2$$

визначте коректність її постановки; встановіть область існування розв'язку. Знайдіть розв'язок методом характеристик.

4. Для рівняння

$$3\frac{\partial u(x,t)}{\partial t} + 4\frac{\partial u(x,t)}{\partial x} = 0, \quad t > 0;$$
$$u(x,0) = 1 - x, \quad 0 \leqslant x \leqslant 1;$$
$$u(0,t) = \cos(\pi t), \quad 0 \leqslant t \leqslant 1$$

визначте коректність; встановіть область існування розв'язку; знайдіть розв'язок методом характеристик і різницевим методом. Порівняйте отримані результати.

5. Визначте порядок апроксимації хвильового рівняння (14.16) неявною різницевою схемою (14.27) і дослідіть стійкість неявної різницевої схеми (14.27).

6. Методом кінцевих різниць розв'яжіть рівняння, що описує коливання струни:

$$\frac{\partial^2 u(x,t)}{\partial t^2} = 2\frac{\partial^2 u(x,t)}{\partial x^2}, \quad 0 \leqslant x \leqslant 1, t > 0$$

з початковими умовами

$$u(x,0) = \sin(\pi x), \quad \frac{\partial u(x,0)}{\partial t} = 0$$

і граничними умовами

$$u(0,t) = 0, \quad u(1,t) = 0.$$

Розділ 15
Інтегральні рівняння

- Типи інтегральних рівнянь
- Чисельні методи розв'язання інтегральних рівнянь
- Методи апроксимуючих функцій

Розглядаються дві основні групи методів розв'язання інтегральних рівнянь — прямі й ітераційні. Прямі методи дозволяють звести розв'язання інтегральних рівнянь до розв'язання системи алгебраїчних рівнянь за допомогою квадратурних схем чисельного інтегрування апроксимації розв'язку. Ітераційні ж методи базуються на обчисленні послідовності наближених розв'язків, які сходяться до єдиного розв'язку рівняння за обраного початкового наближення. У цьому розділі також описано методи, що дозволяють розв'язувати рівняння з виродженими ядрами.

15.1. Класифікація інтегральних рівнянь

Необхідність у чисельному розв'язанні інтегральних рівнянь виникає під час дослідження об'єктів багатьох типів і систем, наприклад у механіці (скажімо, аналіз міцності конструкцій), фізиці (кристалографія, спектроскопія, теорія плазми) і радіотехніці (оптимальна лінійна фільтрація). Взаємозв'язок інтегральних рівнянь і задачі Коші для звичайних диференціальних рівнянь дозволяє записувати останні в еквівалентній інтегральній формі. Крайові задачі для багатовимірних рівнянь у частинних похідних також зводяться до інтегральних рівнянь меншої розмірності, ядра яких є функціями Гріна.

Інтегральним будемо називати рівняння виду

$$\varphi(x) - \lambda \int_a^b K(x,t)\,\varphi(t)\partial t = f(x), \qquad (15.1)$$

де $\varphi(x)$ — функція, що є розв'язком рівняння, λ — відомий параметр рівняння, $K(x,t)$ — ядро інтегрального рівняння, $f(x)$ — вільний член, який іноді називають правою частиною рівняння, $f(x)$ і $K(x,t)$ — відомі функції.

Будемо вважати, що нижня та верхня границі a і b постійні. Параметр λ і функції $\varphi(x)$, $K(x,t)$ та $f(x)$ можуть приймати як дійсні, так і комплексні зна-

чення. У деяких постановках задачі параметр λ не вказується (його приймають рівним одиниці), а добуток $\lambda K(x,t)$ позначають через $K_1(x,t)$ і розглядають $K_1(x,t)$ як нове ядро.

Тип інтегрального рівняння визначається властивостями його ядра. Загалом виділяють три типи інтегральних рівнянь.

1. Рівняння Фредгольма. Ядро $K(x,t)$ неперервне для $a \leqslant x \leqslant b$ і $a \leqslant t \leqslant b$ або має такі розриви, що подвійний інтеграл $\int_a^b \int_a^b |K^2(x,t)|\,\partial x\,\partial t$ є обмеженим.
2. Рівняння зі слабкою особливістю. Ядро рівняння (15.1) може бути задано у вигляді $K(x,t) = H(x,t)/|x-t|^{\alpha}$, де $H(x,t)$ — обмежена функція, а α — постійний параметр, який задовольняє умову $0 < \alpha < 1$.
3. Сингулярні інтегральні рівняння. Ядро має вигляд $K(x,t) = A(x,t)/(x-t)$, де чисельник $A(x,t)$ — диференційована функція за x і t.

Інтегральне рівняння (15.1) називається однорідним, якщо його права частина завжди дорівнює нулю, в іншому разі — неоднорідним. Інтегральне рівняння (15.1) називають інтегральним рівнянням другого роду, а інтегральне рівняння

$$\int_a^b K(x,t)\,\varphi(t)\,\partial t = f(x) \tag{15.2}$$

називають рівнянням першого роду.

Під час розв'язання деяких задач математичної фізики застосовують рівняння Фредгольма, що мають особливий вигляд і називаються *рівняннями Вольтери*. Це рівняння, ядра яких обмежені і тотожно дорівнюють нулю для $t > x$. Такі рівняння називають рівняннями Вольтери другого роду, якщо їх можна записати у вигляді

$$\varphi(x) - \lambda \int_a^x K(x,t)\,\varphi(t)\,\partial t = f(x), \tag{15.3}$$

як і рівняння Вольтери першого роду, що мають вигляд

$$\int_a^x K(x,t)\,\varphi(t)\,\partial t = f(x). \tag{15.4}$$

Задачі Коші відповідають інтегральні рівняння Вольтери, а крайовій задачі — рівняння Фредгольма.

Наближені методи розв'язання інтегральних рівнянь можна розділити на два основні типи: чисельні та апроксимуючих функцій. Спочатку ми розглянемо методи першого типу.

15.2. Чисельні методи розв'язання інтегральних рівнянь

15.2.1. Рівняння з виродженим ядром

Умови існування розв'язку, тобто функції $\varphi(x)$, що задовольняє інтегральні рівняння (15.1)–(15.2) для $a \leqslant x \leqslant b$, визначаються властивостями ядра. Найбільш поширеними наближеними методами розв'язання інтегральних рівнянь, які не мають особливостей, є методи квадратурних сум (скінченних сум), сіток, ітерацій, апроксимуючих функцій. Існує важливий клас інтегральних рівнянь, які можна розв'язати дуже просто зведенням їх до системи алгебраїчних рівнянь. Ядра вказаних інтегральних рівнянь мають бути виродженими, тобто повинна існувати можливість записати їх у вигляді суми обмеженої кількості доданків, кожний з яких є добутком двох множників, причому один множник залежить тільки від x, а інший — тільки від t:

$$K(x,t) = \sum_{i=1}^{n} a_i(x)\, b_i(t). \tag{15.5}$$

Інтегральне рівняння з виродженим ядром можна записати як

$$\varphi(x) - \lambda \sum_{i=1}^{n} a_i(x) \int_a^b b_i(t)\, \varphi(t)\, \partial t = f(x). \tag{15.6}$$

Розв'язок такого рівняння шукають у вигляді:

$$\varphi(x) = f(x) + \lambda \sum_{i=1}^{n} c_i a_i(x), \tag{15.7}$$

де коефіцієнти c_i обчислюють як

$$c_i = \int_a^b b_i(t)\, \varphi(t)\, \partial t, \quad i = 1, 2, ..., n. \tag{15.8}$$

Після підстановки (15.7) у (15.6) і деяких нескладних перетворень отримаємо вираз

$$\sum_{i=1}^{n} a_i(x) \left(c_i - \int_a^b b_i(t) \left[f(t) + \lambda \sum_{k=1}^{n} c_k a_k(t) \right] \partial t \right) = 0, \quad i = 1, 2, ..., n. \tag{15.9}$$

Якщо ввести позначення

$$\int_a^b b_i(t)\, f(t)\, \partial t = f_i, \quad \int_a^b b_i(t)\, a_k(t)\, \partial t = \alpha_{ik}, \quad i, k = 1, 2, ..., n, \tag{15.10}$$

то можна перейти до системи лінійних алгебраїчних рівнянь щодо невідомих c_i:

$$c_i - \lambda \sum_{k=1}^{n} \alpha_{ik} c_k = f_i, \quad i = 1, 2, ..., n. \tag{15.11}$$

Якщо розв'язок такої системи існує, то існує і розв'язок інтегрального рівняння з виродженим ядром (15.6).

Приклад 15.1

У пакеті Mathematica за допомогою методу вироджених ядер побудуємо всі можливі розв'язки однорідного рівняння з ядром $K(x, t) = \cos^2 x \cos 2t + \cos 3x \cos^3 t$ для $a = 0$ і $b = \pi$.

Спочатку опишемо ядро і допоміжні функції

```
In[]:= K[x_, t_]:= (Cos[x])^2*Cos[2t] + Cos[3x]*(Cos[t])^3;
       f[x_]:= 0; y1[x_]:= (Cos[x])^2; y2[x_]:= Cos[3x];
```

Потім обчислимо коефіцієнти α_{ik} (15.10)):

```
In[]:= A[1,1] = Integrate[(Cos[t])^2*Cos[2t],     {t,0,π}];
       A[1,2] = Integrate[(Cos[t])^2*(Cos[t])^3,  {t,0,π}];
       A[2,1] = Integrate[Cos[3t]*Cos[2t],        {t,0,π}];
       A[2,2] = Integrate[Cos[3t]*(Cos[t])^3,     {t,0,π}];
       Array[A, {2,2}]
```

$$\text{Out[]= } \left\{\left\{\frac{\pi}{4}, 0\right\}, \left\{0, \frac{\pi}{8}\right\}\right\}$$

Тепер сформуємо систему лінійних рівнянь (15.11) та розв'яжемо її:

```
In[]:= Q = IdentityMatrix[2] - λ*Array[A, {2,2}]
```

$$\text{Out[]= } \left\{\left\{1 - \frac{\pi\lambda}{4}, 0\right\}, \left\{0, 1 - \frac{\pi\lambda}{8}\right\}\right\}$$

```
In[]:= Q/. λ → π/2
```

$$\text{Out[]= } \left\{\left\{1 - \frac{\pi^2}{8}, 0\right\}, \left\{0, 1 - \frac{\pi^2}{16}\right\}\right\}$$

```
In[]:= P = Det[IdentityMatrix[2] - λ*Array[A,{2,2}]]
```

$$\text{Out[]= } 1 - \frac{3\pi\lambda}{8} + \frac{\pi^2\lambda^2}{32}$$

```
In[]:= B = Solve[P == 0, λ]
```

$$\text{Out[]= } \left\{\left\{\lambda \to \frac{\pi}{4}\right\}, \left\{\lambda \to \frac{\pi}{8}\right\}\right\}$$

Якщо в (15.11) $\lambda \to \pi/4$, то $c_2 = 0$, а c_1 — вільна константа, інакше, якщо $\lambda \to \pi/8$, то $c_1 = 0$, а c_2 — вільна константа, отже, маємо такі розв'язки:

```
In[]:= Y[1] = B[[1, 1, 2]]*C*y1[x];
       Y[2] = B[[2, 1, 2]]*C*y2[x];
       Array[Y,2]
```

$$\text{Out[]= } \left\{\frac{4\,C\,\text{Cos}[x]^2}{\pi}, \frac{8\,C\,[3x]}{\pi}\right\}$$

15.2.2. Метод квадратурних сум

Метод квадратурних сум полягає в заміні інтеграла в рівнянні (15.1) однією з квадратурних сум (див. розділ 7), наприклад:

$$\int_a^b y(t)\,dt = \sum_{i=1}^{n} A_i y(t_i), \tag{15.12}$$

де $t_1, t_2, ..., t_n$ — вузли сітки на відрізку $[a, b]$, A_i — коефіцієнти квадратурної суми. Внаслідок такої заміни отримаємо рівняння

$$\varphi(x) - \lambda \sum_{i=1}^{n} A_i\, K(x, t_i)\,\varphi(t_i) = f(x),$$

яке є апроксимацією інтегрального рівняння (15.1).

Приймаючи $x_i = t_i$, $i = 1, 2, ..., n$, одержимо систему лінійних алгебраїчних рівнянь:

$$\varphi(x_k) - \lambda \sum_{i=1}^{n} A_i\, K(x_k, x_i)\,\varphi(x_i) = f(x_k), \quad k = 1, 2, ..., n \tag{15.13}$$

з невідомими $\varphi(x_1), \varphi(x_2), ..., \varphi(x_n)$, котрі є наближеними значеннями розв'язку $\varphi(x)$ рівняння (15.1) у вузлах $x_1, x_2, ..., x_n$.

Наведемо значення коефіцієнтів і вузлів для різних квадратурних формул.

1. Формула лівих прямокутників:

$$x_i = a + (i-1)h, \quad A_i = h, \quad i = 1, 2, ..., n, \quad \text{де} \quad h = \frac{b-a}{n}.$$

2. Формула трапецій:

$$x_i = a + (i-1)h, \quad i = 1, 2, ..., n,$$
$$A_1 = A_n = \frac{h}{2}, \quad A_2 = A_3 = ... = A_{n-1} = h, \quad \text{де} \quad h = \frac{b-a}{n-1}.$$

3. Формула Сімпсона $(n = 2m + 1)$:

$$x_i = a + (i-1)h, \quad i = 1, 2, ..., n,$$
$$A_1 = A_n = \frac{h}{3}, \quad A_2 = A_4 = ... = A_{2m} = 4h/3, \quad \text{де} \quad h = \frac{b-a}{2m},$$
$$A_3 = A_5 = ... = A_{2m-1} = 2h/3.$$

Таким чином, наближеним розв'язком інтегрального рівняння (15.1) є функція

$$\varphi(x) = \lambda \sum_{i=1}^{n} A_i\, K(x, x_i)\,\varphi(x_i) + f(x). \tag{15.14}$$

Для постійного кроку $h = const$ часто використовують рекурентну формулу методу трапецій:

$$\varphi(a) = f(a),$$
$$\varphi(x_i) = \frac{f(x_i) + h\sum_{j=1}^{i-1} A_j\, K(x_i, x_j)\,\varphi(x_j)}{1 - \frac{h}{2}K(x_i, x_i)}, \qquad (15.15)$$

де $i = 2, 3, ..., n$, $x_i = a + (i-1)h$,

$$A_j = \begin{cases} 0,5 & \text{для } j = 1, \\ 1 & \text{для } j > 1. \end{cases}$$

Приклад 15.2

Інтегральне рівняння

$$(1 - xe^{2x})\cos(1) - e^{2x}\sin(1) + \int_0^{2.5} (1 - (x-t)e^{2x})\varphi(t)\,\partial t = \varphi(x)$$

має точний розв'язок [5]

$$\varphi(x) = e^x(\cos(e^x) - e^x\sin(e^x)).$$

Знайдемо чисельний розв'язок цього рівняння, використовуючи метод трапецій (15.15), і порівняємо його з точним.

Опишемо за допомогою пакета Mathematica ядро інтегрального рівняння, функцію $f(x)$ й точний розв'язок:

```
In[]:= a = 0; b = 2.5;
       h = 0.05;
       n = IntegerPart[(b-a)/h] + 1;
       K[x_,t_]:= 1 - (x - s)*Exp[2x];
       f[x_]:= (1 - x*Exp[2x])*Cos[1] - Exp[2x]*Sin[1];
       ty[x_]:= Exp[x]*(Cos[Exp[x]] - Exp[x]*Sin[Exp[x]]);
```

Визначимо межі інтегрування, задамо початкові значення і знайдемо чисельний розв'язок:

```
In[]:= x[1] = a;
       y[1] = N[f[x[1]]];
       Do [x[i] = a + (i - 1)*h;
           g = f[x[i]];
             Do [If [j == 1, A = 0.5, A = 1];
                 g = g + h*A*K[x[i], x[j]]*y[j],  {j, i-1}];
           y[i] = g/(1 - h/2*K[x[i], x[i]]),{i, 2, n}];
       Q = Table[{x[i], y[i]}, {I, n}}]
```

Побудуємо графіки точного і чисельного розв'язків (рис. 15.1):

```
In[]:= p1 = Plot[ty[x], {x, a, b}, AxesLabel → {"x", "φ(x)"}, DisplayFunction → Identity];
       p2 = ListPlot[Q, PlotJoined → False, PlotStyle → PointSize[.015],
```

```
    DisplayFunction → Identity];
Show[p1, p2, DisplayFunction → $DisplayFunction]
```

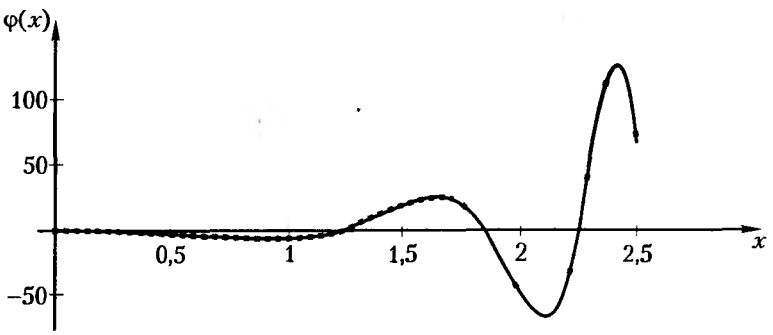


Рис. 15.1. Точний і наближений розв'язки інтегрального рівняння

Приклад 15.3

Використовуючи квадратурну формулу Сімпсона, розв'яжемо інтегральне рівняння:

$$y(x) = \frac{5}{6}x + \frac{1}{2}\int_0^1 xt\, y(t)\, dt.$$

Виберемо вузли $x_1 = 0,\ x_2 = 0,5,\ x_3 = 1$ і обчислимо значення у вузлах правої частини $f(x)$ і ядра $K(x, t) = xt$:

$$f(0) = 0,\quad f(0,5) = 5/12,\quad f(1) = 5/6,$$
$$K(0,\ 0) = 0,\quad K(0,\ 0,5) = 0,\quad K(0,\ 1) = 0,\quad K(0,5,\ 0) = 0,\quad K(0,5,\ 0.5) = 0,25,$$
$$K(0,5,\ 1) = 0,5,\quad K(1,\ 0) = 0,\quad K(1,\ 0,5) = 0,5,\quad K(1,\ 1) = 1.$$

Використовуючи формулу Сімпсона

$$\int_a^b y(t)\, dt \approx \sum_{i=1}^{n} A_i\, y(t_i) \approx \frac{1}{6}[y(0) + 4y(0,5) + y(1)],$$

побудуємо систему рівнянь (15.13) для знаходження наближеного розв'язку рівняння в обраних вузлах:

$$\begin{aligned} y_1 &= 0,\\ \frac{11}{12}y_2 - \frac{1}{24}y_3 &= \frac{5}{12},\\ -\frac{2}{12}y_2 + \frac{11}{12}y_3 &= \frac{5}{6}. \end{aligned}$$

Система має розв'язок $y_1 = 0,\ y_2 = 0,5,\ y_3 = 1$. Згідно з (15.14) наближений розв'язок можна записати у вигляді:

$$y(x) = \frac{5}{6}x + 0,5 \cdot \frac{1}{6}[0 + 4 \cdot 0,5 \cdot 0,5x + 1 \cdot 1 \cdot x] = x,$$

що збігається з точним розв'язком інтегрального рівняння.

15.2.3. Метод послідовних наближень

Даний метод полягає у пошуку послідовних наближень до розв'язку рівняння (15.1) за допомогою рекурентної формули

$$\varphi_n(x) = f(x) + \lambda \int_a^b K(x,t)\,\varphi_{n-1}(t)\,dt. \qquad (15.16)$$

Початковим наближенням $\varphi_0(x)$ може бути функція $f(x)$. Послідовність функцій $\{\varphi_n(x)\}$ збігається до розв'язку рівняння (15.1), якщо $|\lambda| < 1/B$, де

$$B = \sqrt{\int_a^b \int_a^b K(x,t)\,dx\,dt}.$$

Похибка n-го наближення визначається нерівністю

$$|\varphi(x) - \varphi_n(x)| \leqslant \frac{A}{B}|\lambda B|^n \left(\Phi + \frac{F}{1-|\lambda B|}\right), \qquad (15.17)$$

де

$$A = \sqrt{\max \int_a^b K^2(x,t)\,dt}, \quad F = \sqrt{\int_a^b f^2(x)\,dx}, \quad \Phi = \sqrt{\int_a^b \varphi_0{}^2(x)\,dx}.$$

Кількість кроків (наближень) істотно залежить від вибору початкового наближення, тобто від міри близькості його до шуканого розв'язку.

Приклад 15.4

Функція $\varphi(x) = e^x$ є точним розв'язком рівняння $1 + \int_0^7 \varphi(t)\,\partial t = \varphi(x)$ [5]. Методом послідовних наближень (15.16) знайдемо чисельний розв'язок цього рівняння з точністю не менше ніж $\varepsilon < 10^{-3}$. Опишемо відомі функції та ініціюємо початкові значення:

```
In[]:= a = 0; b = 7; h = 0.07; Eps = 0.001;
       n= IntegerPart[(b-a)/h]+1;
       K[x_,s_]:= 1;
       f[x_]:= 1;
       ty[x_]:= Exp[x];
       yk[1] = f[a];
       Do [x[i] = a + (i-1)*h; yk[i] = f[x[i]], {i,n}];    k = 0;
          pogr = Abs[yk[1]];
```

Організуємо ітераційний цикл:

```
In[]:= While [pogr > Eps, k++;
         Do [yk1[i] = yk[i]; yk[i] = f[x[i]];
         If [i≠1, Do [If [(j == 1 || j == i), A = 0.5, A = 1];
                    yk[i] = yk[i] + h*A*K[x[i], x[j]]*yk1[j], {j,i}] ];
         r[i] = Abs[(yk[i] - yk1[i])], {i,1,n} ];
         pogr = Max[Array[r,n]] ]
```

Вихід з циклу буде здійснюватися у разі досягнення заданої точності. Оскільки таблиці чисельного розв'язку досить великі, графіки чисельного і точного розв'язків зобразимо на одному рисунку (рис. 15.2):

```
In[]:= Q = Table[{x[i], yk[i]}, {i,1,n,5}];
       p1 = Plot[ty[x], {x,a,b}, AxesLabel→ {"x", "φ[x]"}, DisplayFunction → Identity]
       p2 = ListPlot[Q, PlotJoined →False, PlotStyle →PointSize[.02],
            DisplayFunction → Identity]
       Show[p1, p2, DisplayFunction → $DisplayFunction]
```

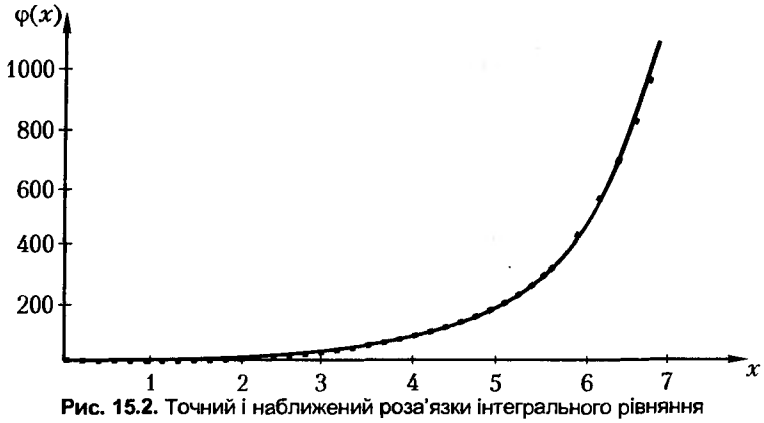


Рис. 15.2. Точний і наближений роза'язки інтегрального рівняння

15.3. Методи апроксимуючих функцій

Використовуючи методи апроксимуючих функцій, наближений розв'язок інтегрального рівняння (15.1) знаходять у вигляді:

$$\varphi(x) = \sum_{i=1}^{n} C_i U_i(x), \tag{15.18}$$

де $U_i(x)$, $i = 1, 2, ..., n$ — система базисних лінійно незалежних функцій, C_i, $i = 1, 2, ..., n$ — шукані коефіцієнти.

Збіжність наближеного розв'язку (15.18) до точного, для $n \to \infty$ забезпечується вибором системи базисних функції, що має бути повною на відрізку $[a, b]$, наприклад, як ортогональні поліноми Лежандра:

$$U_i(x) = P_i(z),$$

де $P_1(z) = 1, P_2(z) = Z, P_3(z) = (3z^2 - 1)/2, P_4(z) = (5z^3 - 3z)/2; z = \dfrac{2x - b - a}{b - a}$.

Підставляючи вирази (15.18) у рівняння (15.1), одержимо нев'язку

$$R(x, C_1, C_2, ..., C_n) = \sum_{i=1}^{n} C_i \left(U_i(x) - \lambda \int_a^b K(x,t) U_i(t)\, dt \right) - f(x), \tag{15.19}$$

де C_i, $i = 1, 2, ..., n$ у (15.18) визначаються після введення додаткових умов.

15.3.1. Метод колокацій

Вимагатимемо, щоб нев'язка $R(x, C_1, C_2, ..., C_n)$ (15.18) у вузлах колокації x_i: $a \leqslant x_1 \leqslant x_2 \leqslant ... \leqslant x_n \leqslant b$ — дорівнювала нулю

$$\begin{cases} R(x_1, C_1, C_2, ..., C_n) = 0 \\ R(x_2, C_1, C_2, ..., C_n) = 0 \\ \quad ... \quad ... \quad ... \quad ... \quad ... \\ R(x_n, C_1, C_2, ..., C_n) = 0 \end{cases} \tag{15.20}$$

Після знаходження коефіцієнтів $C_1, C_2, ..., C_n$ із системи лінійних алгебраїчних рівнянь (15.20) наближений розв'язок інтегрального рівняння (15.1) можна записати у вигляді (15.18).

Приклад 15.5

У пакеті Mathematica розв'яжемо рівняння

$$\varphi(x) = 1 + \int_{-1}^{1} (xt + x^2)\varphi(t)\,dt$$

за допомогою методу колокацій (точний розв'язок $\varphi(x) = 1 + 6x^2$).
Задамо початкові дані, допоміжні таблиці й загальний вигляд чисельного розв'язку:

```
In[]:= a = -1; b = 1;
       K[x_,s_]:= x*s + x^2;
       f[x_]:= 1;
```

Визначимо точки колокації і відповідно до їх кількості сформуємо повну систему функцій з поліномів Лежандра, які задовольнятимуть умовам ортогональності:

```
In[]:= Koll = {a,0,b}; n = Length[Koll];
       F = Table[1/2^i/i!*D[(x^2 - 1)^i, {x,i}], {i,0,n-1}]
```

Out[]= $\left\{1, x, \frac{1}{8}(8x^2 + 4(-1 + x^2))\right\}$

Розв'язок будемо шукати у вигляді:

```
In[]:= Y[x_]:= Sum[Simplify[Cf[i]*F[[i]]], {i,3}];  Y[x]
```

Out[]= Cf[1] + xCf[2] + $\frac{1}{2}$*(-1 + 3x²)Cf[3]

Для обчислення невідомих коефіцієнтів $C_1, C_2, ..., C_n$ побудуємо систему лінійних рівнянь (15.19) і розв'яжемо її:

```
In[]:= S = Table[1/2^i/i!*D[(s^2 - 1)^i, {s,i}], {i,0,n-1}];
       Do [Ur[j] = 0;
          Do [Ur[j] = Ur[j] + Cf[i]*(F[[i]] - Integrate[K[Koll[[j]],s]*S[[i]],
              {s,a,b}]), {i,3} ];
       Ur[j] = Ur[j] - f[Koll[[j]] ] /.x → Koll[[j]], {j,3} ]
       Array[Ur,n]
```

Out[]= $\left\{1 - Cf[1] - \frac{Cf[2]}{3} + Cf[3],\ -1 - Cf[1] - \frac{Cf[3]}{2},\ -1 - Cf[1] - \frac{Cf[2]}{3} + Cf[3]\right\}$

```
In[]:= V = Solve[{Ur[1] == 0, Ur[2] == 0, Ur[3] == 0},{Cf[1], Cf[2], Cf[3]}]
```

Out[]= {{Cf[1] → 3, Cf[2] → 0, Cf[3] → 4}}

Отримаємо остаточний розв'язок за методом колокацій:

```
In[]:= Y = Simplify[Sum[V[[1, i, 2]]*F[[i]], {i, 1, 3}]]
Out[]= 1 + 6x^2,
```

Побудуємо на одному графіку точний та чисельний розв'язки і покажемо точки колокації (рис. 15.3):

```
In[]:= Q = Table[{Koll[[i]], Sum[Cf[[j]]*Koll[[j]]^2, {j, 3}]}, {i, 3}]
       p1 = Plot[Y, {x, a, b}, AxesLabel → {"x", "φ(x)"}, DisplayFunction → Identity]
       p2 = ListPlot[Q, PlotJoined → False, PlotStyle → PointSize[.025],
            DisplayFunction → Identity] Show[p1, p2, DisplayFunction → $DisplayFunction]
```

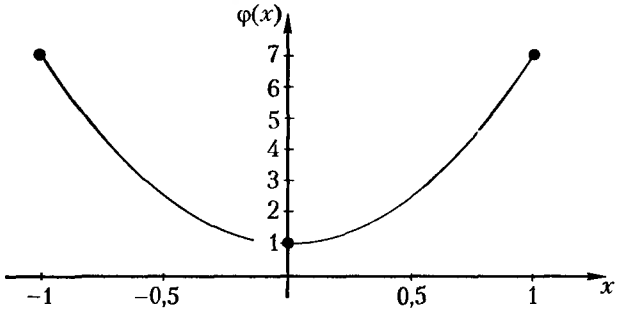


Рис. 15.3. Точний і чисельний розв'язки, що збігаються, та точки колокації

Результат свідчить, що чисельний розв'язок інтегрального рівняння повністю збігається з точним.

15.3.2. Метод найменших квадратів

Як і в методі апроксимуючих функцій, розв'язок інтегрального рівняння шукаємо у вигляді (15.18). Невідомі коефіцієнти визначаються з умови мінімізації

$$J = \sum_{i=1}^{m} R^2(x_i, C_1, C_2, ..., C_n)$$

у вузлах сітки x_i: $a \leqslant x_1 \leqslant x_2 \leqslant ... \leqslant x_m \leqslant b$ $(m \geqslant n)$. Виходячи з цієї умови отримуємо систему лінійних алгебраїчних рівнянь:

$$\sum_{i=1}^{m} R(x_i, C_1, C_2, ..., C_n) \frac{\partial R}{\partial C_k}(x_i, C_1, C_2, ..., C_n) = 0, \quad k = 1, 2, ..., n$$

чи

$$\sum_{i=1}^{m} A_{ki} C_i = B_k, \quad k = 1, 2, ..., n, \tag{15.21}$$

де

$$A_{ki} = \sum_{j=1}^{m} \left(U_k(x_j) - \lambda \int_a^b K(x_j, \xi) U_k(\xi)\, d\xi \right) \left(U_i(x_j) - \lambda \int_a^b K(x_j, \xi) U_i(\xi)\, d\xi \right),$$
$$B_k = \sum_{j=1}^{m} \left(U_k(x_j) - \lambda \int_a^b K(x_j, \xi) U_k(\xi)\, d\xi \right) f(x_j). \tag{15.22}$$

Приклад 15.6

Застосувавши пакет Mathematica, розв'яжемо інтегральне рівняння з прикладу (15.5) методом найменших квадратів.

Задамо початкові дані, допоміжні таблиці й визначимо вигляд, у якому будемо шукати розв'язок:

```
In[]:= a = -1; b = 1;
       K[x_,s_]:= x*s + x^2; f[x_]:= 1;
```

Запишемо повну систему функцій, яка задовольняє вимоги ортогональності:

```
In[]:= F = Table[x^(i-1), {i,3}]
```

Out[]= $\{1, x, x^2\}$

Розв'язок будемо шукати у вигляді:

```
In[]:= Y = Table[Sum[C[i]*F[[i]], {i,3}]]
```

Out[]= $C[1] + xC[2] + x^2C[3]$

Сформуємо відповідно до (15.22) матрицю коефіцієнтів **A** і вектор вільних членів **B**:

```
In[]:= S = Table[s^(i-1), {i,3}];
       Do [ Do [ A[i,j] = Integrate[(F[[j]] -
                          Integrate[K[x,s]*S[[j]], {s,a,b}])*(F[[i]] -
                          Integrate[K[x,s]*S[[i]], {s,a,b}]), {x,a,b}], {j,3} ];
       B[i] = Integrate[f[x]*(F[[i]] - Integrate[K[x,s]*S[[i]], {s,a,b}]), {x,a,b}], {i,3} ]
       Array[A,{3,3}]
```

Out[]= $\left\{\left\{\frac{14}{15}, 0, -\frac{2}{45}\right\}, \left\{0, \frac{2}{27}, 0\right\}, \left\{-\frac{2}{45}, 0, \frac{2}{45}\right\}\right\}$

```
In[]:= Array[B,{3}]
```

Out[]= $\left\{\frac{2}{3}, 0, \frac{2}{9}\right\}$

Розв'яжемо цю систему лінійних рівнянь відносно невідомих **C**:

```
In[]: = Cf = LinearSolve[Array[A, {3,3}], Array[B, {3}]]
```

Out[]= {1, 0, 6}

Отримаємо остаточний розв'язок за методом найменших квадратів:

```
In[]:= Y = Table[Sum[Cf[[i]]*x^(i - 1), {i,1,3}]]
```

Out[]= $1 + 6x^2$

який збігається з розв'язком, що був отриманий методом колокацій, і з точним розв'язком інтегрального рівняння.

15.3.3. Метод моментів (метод Гальоркіна)

Наближений розв'язок інтегрального рівняння (15.1) шукають у вигляді суми $f(x)$ і лінійної комбінації раніше вибраних лінійно незалежних функцій $U_i(x)$, $i = 1, 2, ..., n$. Отже, наближеним розв'язком є

$$\tilde{\varphi}(x) = f(x) + \sum_{i=1}^{n} C_i U_i(x), \qquad (15.23)$$

де коефіцієнти C_i, $i = 1, 2, \ldots, n$ визначаються у такий спосіб. У рівняння $\varphi(x) - \lambda \int_a^b K(x,t)\varphi(t)\,\partial t - f(x) = 0$ замість $\varphi(x)$ підставляють функцію $\tilde{\varphi}(x)$, і в результаті отримують вираз

$$\sum_{i=1}^{n} C_i \left(U_i(x) - \lambda \int_a^b K(x,t) U_i(t)\,dt \right) = \Phi(x, C_i), \quad i = 1, 2, \ldots, n.$$

Виходячи з умов ортогональності функції $\Phi(x, C_i)$ до всіх функцій $U_i(x)$ на відрізку $[a, b]$, отримують рівняння

$$\int_a^b U_i(x)\Phi(x, C_i)\,dx = 0, \quad i = 1, 2, \ldots, n, \tag{15.24}$$

з яких складається система лінійних алгебраїчних рівнянь відносно невідомих коефіцієнтів C_i, $i = 1, 2, \ldots, n$.

Приклад 15.7

Розв'яжемо в пакеті Mathematica рівняння з прикладу 15.5 за допомогою методу Гальоркіна.

Задамо початкові дані, допоміжні таблиці й визначимо вигляд, у якому будемо шукати чисельний розв'язок:

```
In[]:= a = -1; b = 1;
       K[x_,s_]:= x*s + x^2; f[x_]:= 1; n = 3;
       S = Table[1/2^i/i!*D[(s^2 - 1)^i, {s,i}], {i,0,n-1}]
       F = Table[1/2^i/i!*D[(x^2 - 1)^i, {x,i}], {i,0,n-1}]
```

$$\text{Out[]}= \left\{1,\ s,\ \frac{1}{8}(8s^2 + 4(-1 + s^2))\right\}$$

$$\text{Out[]}= \left\{1,\ x,\ \frac{1}{8}(8x^2 + 4(-1 + x^2))\right\}$$

Розв'язок будемо шукати у вигляді:

```
In[]:= y[x_]:= Sum[Simplify[Cf[i]*F[[i]]], {i,3}]; y[x]
```

$$\text{Out[]}= Cf[1] + xCf[2] + \frac{1}{2}(-1 + 3x^2)Cf[3]$$

Визначимо допоміжні функції та сформуємо систему лінійних рівнянь згідно з умовами (15.24):

```
In:= y[s]:= Sum[Simplify[Cf[i]*S[[i]]], {i,3}];
     Yr = f[x] + Integrate[{K[x,s]*y[s]}, {s,a,b}]
```

$$\text{Out=}\ \{1 + 2x^2 Cf[1] + \frac{2}{3}Cf[2]\}$$

```
In[]:= Yl = y[x];
       Url = Integrate[Yr*F[[1]] - Yl*F[[1]], {x,a,b}]
```

$$\text{Out[]=}\ \left\{2 - \frac{2Cf[1]}{3}\right\}$$

```
In[]:= Ur2 = Integrate[Yr*F[[2]] - Yl*F[[2]], {x,a,b}]
```

$$\text{Out[]=}\ \left\{-\frac{2Cf[2]}{9}\right\}$$

```
In[]:= Ur3 = Integrate[Simplify[Yr*F[[3]] - Yl*F[[3]]], {x,a,b}]
```

$$\text{Out[]= } \left\{\frac{8Cf[1]}{15} - \frac{2Cf[3]}{5}\right\}$$

```
In[]:= V = Solve[{Url == 0, Ur2 == 0, Ur3 == 0}, {Cf[1], Cf[2], Cf[3]}]
Out[]= {{Cf[1] → 3, Cf[2] → 0, Cf[3] → 4}}
In[]:= Simplify[Sum[V[[1,i,2]]*F[[i]], {i,1,3}]]
```

$$\text{Out[]= } 1 + 6x^2$$

Таким чином, ми отримали розв'язок, що збігається з попередніми.

Висновки

1. Сфера застосування інтегральних рівнянь досить широка, вона охоплює галузі, в яких традиційно використовуються диференціальні рівняння в частинних похідних і звичайні диференціальні рівняння. Інтегральні рівняння дозволяють знизити розмірність деяких задач дослідження суцільних середовищ і полів, ефективно розв'язувати задачі моделювання нестаціонарних систем, систем із розподіленими параметрами, задачі відновлення та фільтрації сигналів тощо. Відомі закони збереження маси, імпульсу й енергії мають інтегральне формулювання і дають змогу використовувати інтегральні рівняння як моделі конкретних процесів і явищ. При цьому такі моделі не містять похідних від функцій, що є характеристиками стану середовища, і тому допускається існування розривних розв'язків.
2. Залежно від принципу побудови розрізняють дві основні групи методів розв'язання інтегральних рівнянь — прямі й ітераційні. Прямі методи полягають у зведенні інтегральних рівнянь до більш простих рівнянь чи систем, наприклад систем лінійних алгебраїчних рівнянь, і розв'язанні їх за допомогою квадратурних схем чисельного інтегрування апроксимації розв'язку з використанням композиції базисних функцій. Ітераційні методи, як звичайно, базуються на знаходженні послідовності наближених розв'язків, що збігаються до єдиного розв'язку рівняння у разі вибору певного початкового наближення.
3. Система лінійних алгебраїчних рівнянь, що утворюється у разі розв'язання інтегрального рівняння Фредгольма чи Вольтери методом квадратурних сум, часто є погано обумовленою. Більшу стійкість має розв'язок інтегрального рівняння Фредгольма (Вольтери) другого роду.
4. Якщо ядро інтегрального рівняння вироджене, тобто представлене у вигляді суми кінцевого числа доданків, кожний з яких є добутком двох множників, і один із них залежить тільки від x, а другий тільки від t, то метод Гальоркіна у разі вибору складових ядра $a_i(x)$ як базисних функцій у виразі (15.17) гарантує точне розв'язання інтегрального рівняння другого роду.
5. У випадку коли функції, що входять в інтегральне рівняння, періодичні, з періодом $b-a$, досить точний розв'язок отримують застосуванням методу трапецій. Для неперіодичних функцій точні результати дає метод квадратури Гаусса, метод трапецій чи метод Сімпсона з екстраполяцією за Річардсоном.

Контрольні запитання та завдання

1. Знайдіть наближений розв'язок інтегральних рівнянь методом квадратурних сум із використанням квадратурних формул чисельного інтегрування прямокутників, трапецій, формули Сімпсона:

$$\varphi(x) = 1 + \int_0^1 xt^2\,\varphi(t)\,dt;$$

$$\varphi(x) = x + 4 * \int_0^1 x^2t^2\,\varphi(t)\,dt;$$

$$\varphi(x) = (5/6)x + 1/2\int_0^1 xt\,\varphi(t)\,dt;$$

$$\varphi(x) = 1 + \int_{-1}^1 (xt + x^2)\,\varphi(t)\,dt;$$

$$\varphi(x) = 1 + (4/3)x + \int_{-1}^1 (xt^2 - x)\,\varphi(t)\,dt.$$

2. Знайдіть наближений розв'язок інтегральних рівнянь методом послідовних наближень. Розв'язок визначіть із точністю не менш ніж 0,01:

$$\varphi(x) = (1/2)(e^{-x} + 3x - 1) + \int_0^1 (e^{-t} - 1)x\,\varphi(t)\,dt;$$

$$\varphi(x) = e^{-x} - x - \int_0^1 x(e^{xt} - 1)\,\varphi(t)\,dt;$$

$$\varphi(x) = x + \cos(x) + \int_0^1 x(\sin(t) - 1)\,\varphi(t)\,dt;$$

$$\varphi(x) = x/2 + (1/2)\sin(x) + \int_0^1 (1 - \cos(xt))\,x\,\varphi(t)\,dt;$$

$$\varphi(x) = \sin(x) + \int_0^1 (1 - x\cos(t))\,\varphi(t)\,dt.$$

3. Знайдіть наближений розв'язок інтегральних рівнянь методами колокацій, найменших квадратів і моментів.

$$\varphi(x) = 1 - x + \int_0^1 x\,\varphi(t)\,dt;$$

$$\varphi(x) = x + \int_0^1 x^2t\,\varphi(t)\,dt.$$

Література

1. *Бахвалов Н. С., Жидков Н. П., Кобельков Г. М.* Численные методы. — М.: Лаборатория базовых знаний, 2002. — 632 с.
2. *Бахвалов Н. С.* Численные методы. — М.: Наука, 1978. — 632 с.
3. *Березин И. С., Жидков Н. П.* Методы вычислений. — Т. 1 — М.: Наука, 1966. — Т. 2 — М.: Физматгиз, 1962. — 640 с.
4. *Вержбицкий В. М.* Основы численных методов: Учебник для вузов. — М.: Высш. шк., 2002. — 840 с.
5. *Верлань А. Ф., Сизиков В. С.* Интегральные уравнения: методы, алгоритмы, программы. — К.: Наук. Думка, 1986. — 542 с.
6. *Воеводин В. В.* Матрицы и вычисления. — М.: Наука, 1984. — 318 с.
7. *Говорухин В. Н., Цибулин В. Г.* Компьютер в математическом исследовании. Учебный курс. — СПб.: Питер, 2001. — 624 с.
8. *Годунов С. К., Рябенький В. С.* Разностные схемы. Введение в теорию — М.: Наука, 1973. — 400 с.
9. *Данилина Н. И., Дубровская Н. С., Кваша О. П.* Численные методы. — М.: Высш. шк., 1976. — 386 с.
10. *Джордж А., Лю Д.* Численное решение больших разреженных систем уравнений. — М.: Мир, 1984. — 333 с.
11. *Демидович Б. П., Марон И. А.* Основы вычислительной математики. — М.: Наука, 1970. — 664 с.
12. *Дьяконов В. П.* Mathematica 4: учебный курс — СПб.: Питер, 2001. — 656 с.
13. *Зеленський К. Х., Ігнатенко В. М., Коц О. П.* Комп'ютерні методи прикладної математики. — К.: Академперіодика, 2002. — 480 с.
14. *Ильин В. П., Кузнецов Ю. М.* Трехдиагональные матрицы и их приложение. — М.: Наука, 1985. — 207 с.
15. *Калиткин Н. Н.* Численные методы. — М.: Наука, 1972 — 512 с.
16. *Каханер Д., Моулер К., Нэш С.* Численные методы и математическое обеспечение. — М.: 1998. — 575 с.
17. *Крылов В. И., Бобков В. А., Монастырский А. И.* Вычислительные методы. — М.: Наука. — Т. 1. — 1976. — 302 с. — Т. 2. — 1977. — 399 с.
18. *Ляшко И. И., Макаров В. Л., Скоробогатько А. А.* Методы вычислений. — К.: Высш. шк., 1977. — 408 с.
19. *Марчук Г.* Методы вычислительной математики. — М.: Наука, 1980. — 534 с.
20. *Норенков И. П.* Основы автоматизированного проектирования. — М.: Изд-во МГТУ им. Баумана, 2002. — 336 с.
21. *Ортега Д., Пул У.* Введение в численные методы решения дифференциальных уравнений. — М.: Наука, 1986. — 288 с.

22. *Петренко А. І.* Обчислювальна математика: Конспект лекцій. — К.: Вид-во МУРОЛ «Україна», 2002. — 210 с.
23. *Потемкин В. Г.* Система инженерных и научных расчетов MATLAB 5.x.: В 2 т. — М.: ДИАЛОГ-МИФИ, 1999. — 670 с.
24. *Плис А. И., Сливина Н. А.* MATHCAD 2000. Практикум для экономистов и инженеров. — М.: Финансы и статистика, 2000. — 656 с.
25. *Мэтьюз Д. Г., Финк К. Д.* Численные методы. Использование MATLAB. — М.: СПб.; К.: Вильямс, 2001. — 713 с.
26. *Ракитский Ю. В., Устинов С. М., Черноруцкий И. Г.* Численные методы решения жестких систем. — М.: Наука, 1979. — 208 с.
27. *Самарский А. А.* Введение в численные методы. — М.: Наука, 1982. — 272 с.
28. *Самарский А. А. Николаев Е. С.* Методы решения сеточных уравнений. — М.: Наука, 1978. — 592 с.
29. *Самарский А. А., Гулин А. В.* Численные методы. — М.: Наука, 1989. — 429 с.
30. *Самарский А. А., Гулин А. В.* Численные методы математической физики. — М.: Научный мир, 2003. — 316 с.
31. *Сигорский В. П.* Математический аппарат инженера. — К.: Техника, 1975. — 783 с.
32. *Сильвестр П., Феррари Р.* Метод конечных элементов для радиоинженеров и инженеров-электриков. — М.: Мир, 1986. — 229 с.
33. *Тихонов А. Н., Арсенин В. Я.* Методы решения некорректных задач. — М.: Наука, 1986. — 228 с.
34. *Тихонов А. Н., Самарский А. А.* Уравнения математической физики. — М.: Наука, 1966. — 724 с.
35. *Уилкинсон Д.* Алгебраическая проблема собственных значений. — М.: Наука, 1970. — 540 с.
36. *Холл Д., Уатт Д.* Современные численные методы решения обыкновенных дифференциальных уравнений. — М.: Мир, 1979. — 312 с.
37. *Фадеев Д. К., Фадеева В. Н.* Вычислительные методы линейной алгебры. — М.: Физматгиз, 1963. — 734 с.
38. *Фельдман Л. П.* Численные методы и математические пакеты. Решение задач в пакете Mathematica-3. — Донецк: ДонГТУ, 2000. — 96 с.
39. *Форсайт Д., Малькомп Н., Моулер К.* Машинные методы математических вычислений. — М.: Мир, 1980. — 281 с.
40. *Штеттер Х.* Анализ дискретизации для обыкновенных дифференциальных уравнений. — М.: Мир, 1978. — 461 с.
41. *Эстербю О., Златев З.* Прямые методы разреженных матриц.— М.: Мир, 1987. — 118 с.
42. *Chapra S. C., Canale R. P.* Numerical methods for engineers. — Singapore: McGraw-Hill Book Company, 1990. — 812 p.
43. *Dahlquist G., Björck Å.* Numerical Methods.— New Jersey: Prentice-Hall, Inc. — 1974. — 574 p.
44. *Ralston A., Rabinowitz P.* A First Course in Numerical Analysis. — 2d ed. — New York: McGraw-Hill, 1978. — 635 с.

Алфавітний покажчик

Е

Ж

З

І

К

М

Н

О

П

Навчальне видання

Фельдман Лев Петрович
Петренко Анатолій Іванович
Дмитрієва Ольга Анатоліївна

ЧИСЕЛЬНІ МЕТОДИ В ІНФОРМАТИЦІ

Підручник

Керівник проекту В. П. Пасько
Редактор С. Г. Єзерницька
Комп'ютерна верстка Д. С. Трішенкова

ТОВ «Видавнича група BHV»
Свідоцтво про внесення до Державного реєстру
суб'єктів видавничої справи України
серія ДК №175 від 13.09.2000 р.

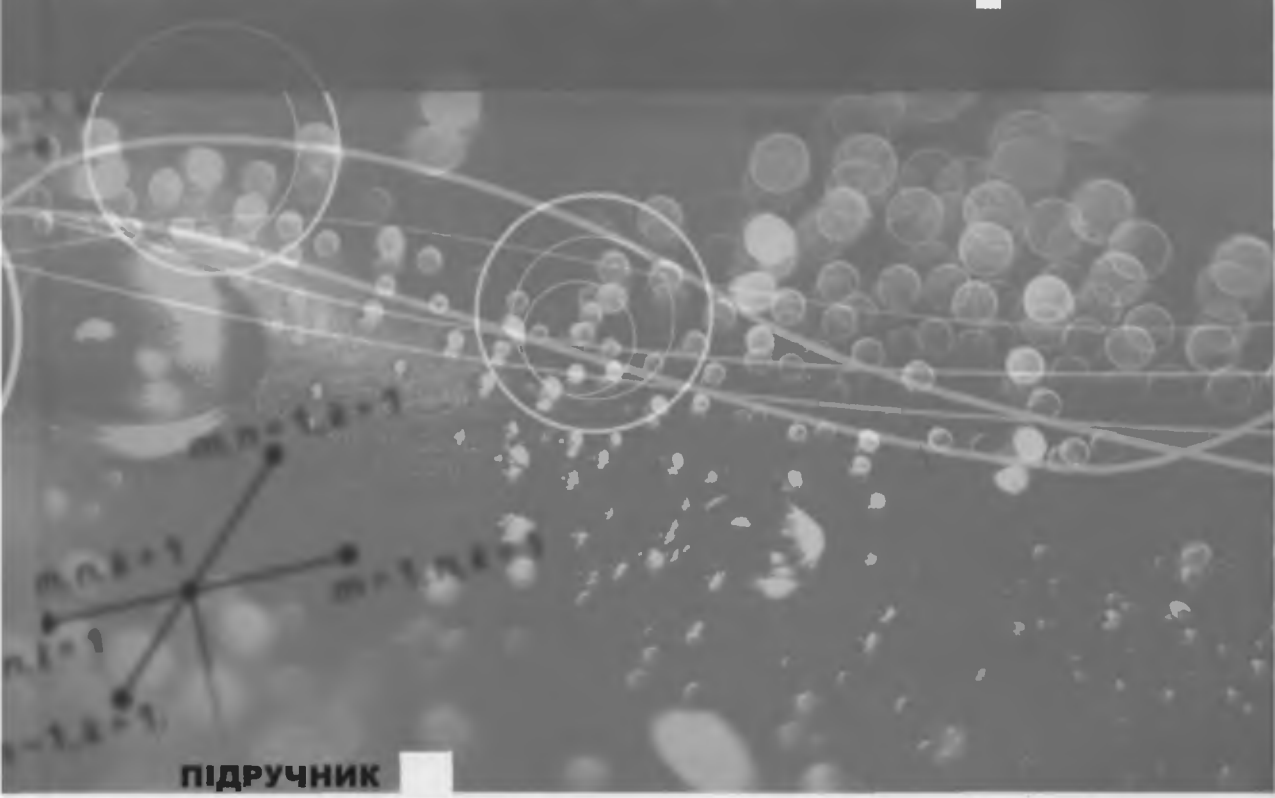
Підписано до друку 14.02.06. Формат 70×100 1/16.
Папір офсетний. Гарнітура Petersburg. Друк офсетний.
Ум. друк. арк. 38,7. Обл.-вид. арк. 37,72.
Наклад 3000 прим. Зам. №30211.

Виготовлено в ТОВ «Освітня книга»,
м. Київ, вул. Орловська, 2/7, оф. 6.
Свідоцтво про внесення до Державного реєстру
суб'єктів видавничої справи України
серія ДК № 2245 від 26.07.2005 р.

інформатика

Л. П. Фельдман, А. І. Петренко, О. А. Дмитрієва

ЧИСЕЛЬНІ МЕТОДИ В ІНФОРМАТИЦІ



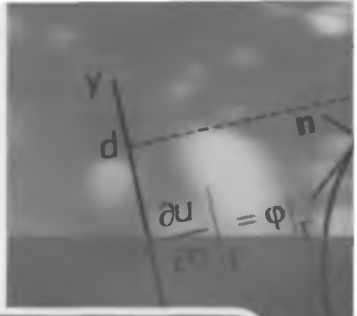
ПІДРУЧНИК

ДЛЯ ВИЩИХ НАВЧАЛЬНИХ ЗАКЛАДІВ

bhv ПИТЕР

Л. П. Фельдман, А. І. Петренко, О. А. Дмитрієва

ЧИСЕЛЬНІ МЕТОДИ В ІНФОРМАТИЦІ



У підручнику розглянуто прикладні аспекти застосування обчислювальних методів в інженерній практиці. Подано численні приклади розв'язання обчислювальних задач за допомогою пакета Mathematica, а також рекомендації з використання окремих чисельних процедур для розв'язання технічних задач.
Матеріал підготовлено на основі багаторічного досвіду авторів з викладання відповідних дисциплін та результатів науково-дослідних робіт.

Детально розглянуто такі теми:

- розв'язання систем лінійних та нелінійних рівнянь;
- обчислення власних значень і власних векторів матриць;
- розв'язання звичайних диференціальних рівнянь;
- чисельне інтегрування та диференціювання функцій;
- розв'язання диференціальних рівнянь з частинними похідними;
- наближення та інтерполяція функцій;
- розв'язання інтегральних рівнянь.

Базовий курс для студентів вищих навчальних закладів, які навчаються за напрямами: «Комп'ютерні науки», «Комп'ютерна інженерія», «Прикладна математика», «Комп'ютеризовані системи, автоматика і управління».

Зміст підручників серії відповідає навчальним планам з інформатики, що використовуються у провідних вищих навчальних закладах України

Фельдман Лев Петрович — доктор технічних наук, професор кафедри прикладної математики та інформатики Донецького національного технічного університету. Сфера наукових інтересів — паралельні чисельні алгоритми і структури обчислювальних систем.

Петренко Анвтолій Іввнович — доктор технічних наук, професор, лауреат Дврджавної премії України в галузі науки і техніки, Заслужений діяч науки і техніки України, ад'юнкт-професор Мічіганського державного університету, почвсний професор Середньо-Піеденного Університету Китаю та почвсний член IEE. Відомий у світі фахіввць у галузі моделювання складних технічних об'єктів та побудови систем автоматизованого проектування.

Дмитрієвв Ольгв Анвтолїівна — кандидат технічних наук, доцент кафедри прикладної математики і інформатики Донецького національного технічного університву. Сфера наукових інтересів — моделювання динамічних систем великої розмірності, розробка чисвльних алгоритмів, призначених для реалізації в багатопроцесорних обчислювальних системах.

За додатковою інформацією звертайтеся на сайти www.osvita.info, www.bhv.kiev.ua, www.piter.com

ISBN 966SS2155-1

bhv ПИТЕР®