МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ СЕМЕНА КУЗНЕЦЯ

Задачин В. М.
Конюшенко І. Г.

ЧИСЕЛЬНІ МЕТОДИ

Навчальний посібник

Харків. Вид. ХНЕУ ім. С. Кузнеця, 2014

УДК 517.9(075.8)
ББК 22.161.6я73
3-15

Рецензенти: докт. техн. наук, професор кафедри прикладної математики Харківського національного університету радіоелектроніки *Тевяшев А. Д.*; докт. техн. наук, професор кафедри комп'ютерної математики та математичного моделювання Національного технічного університету "ХПІ" *Любчик Л. М.*

Рекомендовано до видання рішенням вченої ради Харківського національного економічного університету імені Семена Кузнеця.
Протокол № 9 від 22.04.2014 р.

Задачин В. М.
3-15 Чисельні методи : навчальний посібник / В. М. Задачин, І. Г. Конюшенко. – Х. : Вид. ХНЕУ ім. С. Кузнеця, 2014. – 180 с. (Укр. мов.)

Викладено основні методи чисельного аналізу, а саме чисельне розв'язання нелінійних рівнянь, систем лінійних рівнянь, диференціальних рівнянь, інтегральних рівнянь тощо. Усі чисельні методи проілюстровано прикладами їх програмної реалізації та застосування для розв'язання типових математичних задач і деяких практичних задач у пакеті R.
Рекомендовано для студентів напряму підготовки 6.050101 "Комп'ютерні науки".

ISBN 978-966-676-547-6

УДК 517.9(075.8)
ББК 22.161.6я73

Зміст

Позначення

Позначення	Значення
1	2
R^1	простір дійсних чисел
$x \in G$	x належить множині G (x – елемент множини G)
$i = \overline{1,n}$	i приймає значення від 1 до n (еквівалентно $i = 1,\ldots,n$)
$\forall x \in G$	для всіх x, що належать множині G
$\exists x \in G$	існує x, що належить множині G
$K >> 1$	число K у багато разів більше, ніж 1
R^n	n-вимірний дійсний простір векторів-стовпців, тобто $x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$, де $x_i \in R^1$, $i = \overline{1,n}$
$x \in R^n$	x – n-вимірний вектор-стовпець
x^T	вектор-рядок, транспонований до x
$x^{(k)} \in R^n$	$x^{(k)}$ – n-вимірний вектор-стовпець, k-й за номером у деякій нумерації
x_i	i-та компонента вектора x
$\left\{x^{(k)}\right\}$	послідовність $x^{(k)}$, тобто $x^{(0)}, x^{(1)}, x^{(2)},\ldots$
$\left\{x \in G : T\right\}$	підмножина елементів множини G, що задовольняють умові T
$A = \left(a_{ij}\right)$	матриця з елементами a_{ij}, де a_{ij} – елемент i-го рядка і j-го стовпця
A^T	матриця, транспонована до A, тобто матриця з елементами a_{ij}
A^{-1}	матриця, обернена до A
$R^{n\times m}$	множина матриць розмірності n на m

1	2
$E \in R^{n\times n}$	одинична матриця розмірності n на n, тобто $E = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$
$f : R^1 \to R^1$	дійсна функція однієї дійсної змінної, тобто якщо $x \in R^1$ й $y = f(x)$, то $y \in R^1$
$f : R^n \to R^m$	дійсна вектор-функція n дійсних змінних, тобто якщо $x \in R^n$ й $y = f(x)$, то $y \in R^m$
$\langle x, y \rangle$	скалярний добуток векторів x, y, тобто якщо $x, y \in R^n$, то $\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i$
$\|x\|$	евклідова норма вектора x, тобто якщо $x \in R^n$, то $\|x\| = \left[\sum_{i=1}^{n} x_i^2 \right]^{\frac{1}{2}} = \sqrt{\langle x, x \rangle}$
$v_r(x^*)$	окіл точки $x^* \in R^n$ радіусу r, тобто $V_r(x^*) = \left\{ x \in R^n : \|x - x^*\| \le r \right\}$
Δ	оператор Лапласа. У випадку функції $u = u(x, y)$ двох незалежних змінних x і y він має вигляд $\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$
Δ^2	бігармонічний оператор. У випадку функції $u = u(x, y)$ двох незалежних змінних x і y він має вигляд $\Delta^2 u = \frac{\partial^4 u}{\partial x^4} + 2\frac{\partial^4 u}{\partial x^2 \partial y^2} + \frac{\partial^4 u}{\partial y^4}$

Вступ

Не буває зайвої інформації, буває
невміння нею розпорядитися.

Сучасний розвиток науки та обчислювальної техніки характеризується все більш зростаючим рівнем використання комп'ютерних моделей як для дослідження поведінки явищ та процесів, що оточують людину, так і для розв'язання практичних задач, пов'язаних з управлінням та прогнозуванням.

Вивчення навчальної дисципліни "Чисельні методи" дозволяє студентам оволодіти знаннями в галузі практичних методів рішення математичних проблем, що виникають у процесі інженерної діяльності та моделювання фізичних систем, засвоїти способи розрахунків на сучасних комп'ютерах із застосуванням пакетів спеціальних прикладних програм.

Об'єктом вивчення навчальної дисципліни є типові математичні задачі, до яких зводиться рішення практичних проблем, що виникають у ході розробки інформаційних систем та систем моделювання. Предметом вивчення навчальної дисципліни є чисельні методи розв'язання типових математичних задач.

Мета даного навчального посібника – ознайомлення студентів із постановками основних математичних задач і чисельних методів їх розв'язання, набуття студентами навичок реалізації на комп'ютері чисельних методів, навичок роботи з відомими комп'ютерними математичними пакетами.

Згідно з вимогами освітньо-професійної програми підготовки бакалаврів напряму 6.050101 "Комп'ютерні науки" студенти після вивчення навчальної дисципліни "Чисельні методи" повинні оволодіти такими компетентностями:

знати:

загальні поняття, пов'язані з чисельними методами;

постановки типових математичних задач;

чисельні методи лінійної та нелінійної алгебри;

чисельні методи наближення функцій;

методи чисельного диференціювання та інтегрування функцій;

чисельні методи розв'язання звичайних диференціальних рівнянь;

чисельні методи розв'язання інтегральних рівнянь;

чисельні методи розв'язання задач математичної фізики;

теоретичні особливості чисельних методів та можливості їх адаптації до інженерних задач;

вміти:

проектувати, програмувати, тестувати й налагоджувати програми, що реалізують чисельні методи;

розв'язувати математичні задачі з використанням математичних пакетів;

здійснювати обґрунтований вибір чисельного методу при вирішенні практичної задачі.

Одним із способів розв'язання практичної задачі є експеримент. Наприклад, будують ракету, запускають і перевіряють характеристики, які цікавлять. Якщо вони не задовольняють, то будують нову ракету і т. д. Зрозуміло, що потрібний результат (при необмежених ресурсах) буде, врешті-решт, досягнутий, проте занадто дорогою ціною.

Інший спосіб – побудова математичної (комп'ютерної) моделі об'єкта або явища, що вивчається, і проведення всіх розрахунків на комп'ютері. Для фізичних процесів математична модель зазвичай записується у вигляді набору рівнянь, в які як коефіцієнти входять характеристики тіл або речовин, що беруть участь у процесі. Наприклад, швидкість ракети $v = v(t)$ (у момент часу t після старту) при вертикальному польоті у вакуумі визначається рівнянням [10]:

$$\left(M - \int_0^t m(\tau)d\tau \right)\left(\frac{dv}{dt} + g \right) = c \times m(t), \qquad (1)$$

де M – початкова маса ракети;

$m(t)$ – задана витрата пального;

g – прискорення поля тяжіння;

c – швидкість закінчення витікання газів, що залежить від калорійності пального і середньої молекулярної ваги продуктів горіння.

Для того щоб зрозуміти місце чисельних методів у процесі розв'язання задач, виникаючих у практичній діяльності людини, з використанням комп'ютера, слід навести основні етапи цього процесу.

Етапи розв'язання практичних задач на комп'ютері.

1. Постановка задачі:
   - словесне формулювання задачі;
   - визначення кінцевої мети розв'язку.
2. Побудова математичної моделі, тобто математичне формулювання задачі.
3. Вибір чисельного методу для розв'язання математичної задачі.
4. Розроблення алгоритму.
5. Програмна реалізація алгоритму.
6. Тестування програми (налагодження на тестових задачах).
7. Проведення розрахунків на реальних даних.
8. Аналіз результатів.

Найскладнішим із перерахованих етапів є другий етап, а вивчення методів його реалізації є предметом інших навчальних дисциплін (наприклад, моделювання систем й ін.). У подальшому викладенні матеріалу даного навчального посібника передбачено, що математичне формулювання задачі вже є, потрібно тільки навчитися її розв'язувати на комп'ютері з використанням чисельних методів.

Варто зазначити, що якщо математична модель вибрана недостатньо коректно, то які б методи не застосовувалися для розрахунків з її використанням, отримані висновки будуть ненадійні, або й зовсім неправильні. Так, рівняння (1) непридатне для розрахунків запуску ракети з поверхні Землі, бо в ньому не враховується опір повітря.

Розв'язок, отриманий за допомогою чисельного методу, зазвичай є наближеним, тобто містить деяку погрішність. Джерелами погрішності (у порядку значимості) є:

- невідповідність математичної постановки задачі досліджуваному реальному явищу;
- погрішність відправних даних;
- погрішність чисельного методу розв'язання;
- помилки округлення та розрахунків.

У навчальному посібнику висвітлюються питання, пов'язані з розкриттям основних понять предметної області: математичної постановки задачі, чисельного методу, ітерації, характеристик чисельних методів, абсолютної та відносної похибок розв'язку, джерел погрішності та інших; розглянуті постановки основних математичних задач та найбільш відомі чисельні методи їх розв'язання; наведені приклади програмної реалізації чисельних методів.

У ході розв'язання конкретної практичної задачі спеціаліст повинен, перш за все, визначити, до якого типу математичної задачі належить ця практична задача, вибрати чисельний метод для її розв'язання, розробити програмну реалізацію методу самостійно чи вміти застосувати для її розв'язання один із відомих комп'ютерних математичних пакетів.

1. Вступ до чисельних методів. Загальні поняття

1.1. Сутність чисельних методів. Загальні поняття

Для розв'язання математичних задач в основному існує три групи методів:

1. **Аналітичні методи**, в яких розв'язок задачі подається у вигляді аналітичних виразів. Їх **перевагами** є: запис розв'язку у загальному вигляді; висока точність і малий об'єм комп'ютерної пам'яті для зберігання розв'язку. Основний **недолік** – неуніверсальність, бо тільки невелика частина математичних задач може бути розв'язана аналітично.

2. **Графічні методи**, в яких розв'язок задачі знаходиться візуально. Їх **перевагою** є наочність. **Недоліками** графічних методів є: велика трудомісткість; низька точність (залежить від точності побудови графіків); неуніверсальність (графіки можна побудувати тільки для невеликої розмірності та ін.).

3. **Чисельні методи**, що дозволяють звести розв'язування задачі до виконання скінченного числа арифметичних і логічних дій з числами. При цьому розв'язок визначається як набір чисел, які надалі можуть бути інтерпретовані різним способом (наприклад, подані у вигляді таблиць, графіків, анімації тощо). Їх **перевагами** є: абсолютна універсальність, бо теоретично можуть бути застосовані для розв'язання будь-яких задач; добре пристосовані для реалізації на комп'ютері. **Недоліком** є велика трудомісткість у ході ручного рахунку, що, зазвичай, не є проблемою, оскільки вони призначені для використання на комп'ютері.

Таким чином, чисельні методи є основним апаратом розв'язання математичних задач, а їх значущість тільки збільшуватиметься у міру вдосконалення комп'ютерної техніки.

Чисельні методи бувають двох типів: **прямі** та **ітераційні**. В прямих методах розв'язок задачі досягається за скінченну кількість **кроків** методу після виконання останнього кроку, в ітераційних методах виконується ряд **ітерацій** методу до отримання наближеного розв'язку із заданою точністю.

Поняття ітераційного методу.

В основному чисельні методи є **ітераційними**. **Ітерація** – це повторення сукупності операцій або процедур для покращення наявного (поточного) наближеного розв'язку задачі. Нехай x^* – розв'язок задачі, тоді ітераційний метод будує так звану **ітераційну послідовність** $\left\{x^{(k)}\right\}$ $\left(k = 0, 1, 2, \ldots\right)$ наближень розв'язку, при цьому $x^{(k)}$ повинно наближатися до x^* зі збільшенням k.

Алгоритм ітераційного методу в найзагальнішому вигляді має таку схему:

1. Задається **початкове наближення** розв'язку $x^{(0)}$ (на основі апріорних знань про задачу).

2. На k-й ітерації методу $\left(k = 0, 1, 2, \ldots\right)$ буде поточне наближення розв'язку $x^{(k)}$. Далі обчислюється наступне наближення $x^{(k+1)} = \Phi(x^{(k)})$, де Φ і є сукупністю операцій або процедур для покращення наближеного розв'язку задачі, яка є суттю конкретного чисельного методу.

3. Перевіряється **критерій останову**, тобто перевіряється: чи є отримане наближення $x^{(k+1)}$ розв'язку x^* достатньо близьким. Якщо цього немає, то відбувається перехід до наступної ітерації, тобто до пункту 2.

Варто зазначити, що вид критерію останову (тобто припинення обчислень за ітераційним методом) залежить від виду розв'язуваної математичної задачі.

1.2. Характеристики чисельних методів

Для оцінки чисельних методів, тобто порівняння між собою методів для розв'язання однієї задачі, вводять такі їх основні характеристики:

трудомісткість;

порядок методу;

збіжність;

швидкість збіжності;

стійкість до погрішностей обчислень;

стійкість до погрішностей у відправних даних.

Під **трудомісткістю** методу розуміють кількість і якість обчислень, необхідних для досягнення достатньо близького наближення розв'язку задачі.

Під **порядком** методу розуміють вимоги до знань про функції, що входять у математичне формулювання задачі (наприклад, використання в методі похідних цих функцій):

- метод **нульового порядку**, якщо він використовує тільки значення цих функцій;
- метод **першого порядку**, якщо він використовує значення функцій і їх перших похідних;
- метод **другого порядку**, якщо він використовує значення і функцій та їх перших і других похідних і т. д.

Чисельний метод називається **таким, що збігається**, якщо наближення x^k прямує до розв'язку x^* зі збільшенням k. Очевидно, що методи, які не збігаються, не цікаві з прикладної точки зору. Тому одним з найважливіших етапів при введенні нового чисельного методу є теоретичне доведення його збіжності, тобто формулювання умов, за яких метод гарантовано збігається.

В основному розрізняють такі **швидкості збіжності** методів.

1. **Лінійна збіжність**. Указують, що послідовність $\left\{x^{(k)}\right\}$ $(k = 0, 1, 2, \ldots)$ лінійно збігається до розв'язку x^* (або зішвидкістю геометричної прогресії), якщо існують числа $q \in (0, 1)$ і $k_0 > 0$ такі, що

$$\left\|x^{(k+1)} - x^*\right\| \le q\left\|x^{(k)} - x^*\right\| \text{ для всіх } k \ge k_0.$$

Тут норма $\|x - y\|$ означає відстань між x і y.

2. **Надлінійна збіжність**. Указують, що послідовність $\left\{x^{(k)}\right\}$ $(k = 0, 1, 2, \ldots)$ надлінійно збігається до розв'язку x^*, якщо існує послідовність $\{q_k\}$ $(k = 0, 1, 2, \ldots)$, $q_k \in (0, 1)$ для всіх k, така, що

$$\left\|x^{(k+1)} - x^*\right\| \le q_k\left\|x^{(k)} - x^*\right\| \text{ і } q_k \to 0 \text{ при } k \to \infty.$$

3. **Квадратична збіжність**. Указують, що послідовність $\left\{x^{(k)}\right\}$ $(k = 0, 1, 2, \ldots)$ квадратично збігається до розв'язку x^*, якщо існують числа $C > 0$ і $k_0 > 0$ такі, що

$$\left\|x^{(k+1)} - x^*\right\| \le C\left\|x^{(k)} - x^*\right\|^2 \text{ для всіх } k \ge k_0.$$

Під **стійкістю до погрішностей обчислень** розуміють те, що застосування чисельного методу приводить до розв'язку задачі на комп'ютері, незважаючи на помилки округлень і обчислень. Для цього в чисельних методах, якщо потрібно, передбачаються додаткові операції, що не змінюють суть методу, але забезпечують його стійкість до помилок обчислень.

Під **стійкістю до погрішностей у відправних даних** розуміють те, що при невеликих погрішностях у відправних даних застосування чисельного методу дозволяє отримати наближений розв'язок задачі з не дуже великою погрішністю. Стійкість до погрішностей у відправних даних досягається, як правило, шляхом модифікації чисельного методу, тобто внесенням змін до суті методу.

1.3. Похибка розв'язку

Для оцінювання точності чисельного методу слід ввести поняття абсолютної і відносної погрішності розв'язку.

Якщо x^* – точний розв'язок задачі, а $\tilde{x}$ – наближений розв'язок, то **абсолютною погрішністю** наближення $\tilde{x}$ називається деяка величина $\Delta(\tilde{x})$, про яку відомо, що

$$\left\|\tilde{x} - x^*\right\| \le \Delta(\tilde{x}).$$

Тут норма $\left\|\tilde{x} - x^*\right\|$ означає відстань між $\tilde{x}$ і x^*. Таким чином, якщо відомі наближений розв'язок задачі $\tilde{x}$ та абсолютна погрішність $\Delta(\tilde{x})$, то невідомий точний розв'язок задачі x^* знаходиться від $\tilde{x}$ на відстані не більше, ніж $\Delta(\tilde{x})$.

Також варто зазначити, що абсолютна погрішность розв'язку – це не різниця між точним і наближеним розв'язком задачі, а оцінка цієї різниці.

Відносною погрішністю наближення $\tilde{x}$ називають деяку величину $\delta(\tilde{x})$, про яку відомо, що

$$\frac{\left\|\tilde{x} - x^*\right\|}{\left\|\tilde{x}\right\|} \le \delta(\tilde{x}).$$

Варто звернути увагу на те, що наближений розв'язок задачі $\tilde{x}$, отриманий за допомогою чисельного методу, зазвичай залежить від вибраних параметрів цього методу. Без обмеження загальності, можна вважати, що параметр у методу один (умовно можна позначити його через h) і при цьому він задовольняє умову $0 < h < 1$. Тоді абсолютна погрішність $\Delta(\tilde{x})$ також залежить від h, тобто $\Delta(\tilde{x}) = \Delta(h)$.

Якщо існують деякі числа $M > 0$ і $k > 0$ (не обов'язково цілі) такі, що виконується нерівність $\Delta(h) \le M h^k$, то говорять, що абсолютна погрішність $\Delta(h)$ має порядок $O(h^k)$. Цим хочуть підкреслити **якісну** (на відміну від кількісної) залежність погрішності наближеного розв'язку задачі $\tilde{x}$ від параметра чисельного методу h. Наприклад, це показує, що зменшення значення параметра h на один порядок призводить до зменшення величини абсолютної погрішності $\Delta(\tilde{x})$ на k порядків.

1.4. Похибка округлення у ході розрахунків на комп'ютері з плаваючою крапкою

Одним із основних джерел обчислювальних погрішностей є наближене подання дійсних чисел у комп'ютері, обумовлене скінченністю розрядної сітки. При розв'язанні обчислювальних задач зазвичай використовують подання чисел у **формі з плаваючою крапкою** *(комою)*. Число a у формі з плаваючою крапкою подається у вигляді:

$$a = sign(a) M r^p,$$

де r – основа системи числення;

p – порядок числа a;

M – мантиса числа a;

$sign(a)$ – знак числа a.

При цьому повинна виконуватися умова нормування $r^{-1} \le M \le 1$, тобто перша цифра в мантисі повинна бути відмінною від нуля.

Усі персональні комп'ютери і більшість робочих станцій підтримують **IEEE-стандарт двійкової арифметики** (тобто $r = 2$). Стандарт передбачає два типи чисел із плаваючою крапкою: *числа звичайної точності* (подаються 4 байтами – 32 бітами)

1	8	23
Знак	Порядок	Мантиса

і *числа подвійної точності* (подаються 8 байтами – 64 бітами)

1	11	52
Знак	Порядок	Мантиса

Через скінченність розрядної сітки (кількість знаків у мантисі) в комп'ютері можна подати не всі числа. Число a, яке не можна подати в комп'ютері, піддається округленню, тобто замінюється близьким числом $\tilde{a}$, яке подається в комп'ютері точно.

Слід знайти оцінку *відносної погрішності подання числа з плаваючою крапкою* [3]. Припускається, що застосовується просте округлення – відкидання всіх розрядів числа, що виходять за межі розрядної сітки. Нехай потрібно записати число, що є нескінченним двійковим дробом (система числення – двійкова):

$$a = \pm \underbrace{2^{p}}_{\text{порядок}} \underbrace{\left(\frac{a_1}{2} + \frac{a_2}{2^2} + \cdots + \frac{a_t}{2^t} + \frac{a_{t+1}}{2^{t+1}} + \cdots \right)}_{\text{мантиса}}, \qquad (1.1)$$

де $a_j = \begin{cases} 0 \\ 1 \end{cases}, (j = 1, 2, \ldots)$ – цифри мантиси;

t – розрядність мантиси (23 або 52 залежно від типу числа).

Тоді, після відкидання зайвих розрядів, буде отримане округлене число

$$\tilde{a} = \pm 2^p \left(\frac{a_1}{2} + \frac{a_2}{2^2} + \cdots + \frac{a_t}{2^t} \right).$$

Погрішність округлення в цьому випадку дорівнює

$$\tilde{a} - a = \pm 2^p \left(\frac{a_{t+1}}{2^{t+1}} + \frac{a_{t+2}}{2^{t+2}} + \cdots \right).$$

Найбільша погрішність буде у випадку, коли $a_{t+1}=1$, $a_{t+2}=1$, …. Тоді **абсолютна погрішність округлення**

$$|\tilde{a} - a| \le 2^p \frac{1}{2^{t+1}} \underbrace{\left(1 + \frac{1}{2} + \frac{1}{2^2} + \cdots \right)}_{=2} = 2^{p-t}.$$

З умови нормування виходить, що $M \ge \frac{1}{2}$, а значить завжди $a_1 = 1$. Тоді з (1.1) отримуємо, що $|a| \ge 2^p \times 2^{-1} = 2^{p-1}$ і тому $\frac{|\tilde{a} - a|}{|a|} \le 2^{-t+1}$, тобто відносна погрішність дорівнює 2^{-t+1}.

У реальності застосовують точніші методи округлення (ніж просте відкидання розрядів), тому **відносна погрішність** подання чисел в пам'яті комп'ютера дорівнює:

$$\frac{|\tilde{a} - a|}{|a|} \le 2^{-t}, \qquad (1.2)$$

тобто точність подання чисел визначається розрядністю мантиси t.

Слід зазначити, що з (1.2) витікає, що наближено подане в комп'ютері число $\tilde{a}$ можна записати в вигляді $\tilde{a} = a(1 \pm \varepsilon)$, де $|\varepsilon| \le 2^{-t}$. Тому відносну погрішність подання чисел 2^{-t} ще називають "машинним епсілоном".

Більш детально похибки у ході розрахунків на комп'ютері з плаваючою крапкою розглянуті в роботах [1; 3; 18].

1.5. Математичні пакети

За останні 50 – 60 років, пов'язаних із використанням комп'ютерної техніки, в усьому світі накопичено велику кількість програмного забезпечення у вигляді бібліотек комп'ютерних підпрограм (написаних, у першу чергу, мовами Fortran та C), призначених для розв'язання типових математичних задач. Крім того, розроблено ряд універсальних математичних пакетів, за допомогою яких можна достатньо швидко як розв'язати багато відомих математичних задач, так і провести попередні розрахунки з подальшою реалізацією вже в проектованій комп'ютерній системі.

Найбільш відомими є такі математичні пакети: R, Mathematica, Maple, MatLab, MathCad [3; 21].

Пакет **Mathematica** найпопулярніший у наукових колах, особливо у теоретиків. Він надає широкі можливості під час проведення символічних (аналітичних) перетворень, дозволяє швидко й ефективно розв'язувати багато задач обчислювальної математики. В пакеті Mathematica більшість задач розв'язується в діалоговому режимі, без традиційного програмування з використовуванням стандартних операторів. Крім того, в цьому пакеті реалізована мова програмування високого рівня, яка дозволяє створювати достатньо складні програми. Також цей пакет дозволяє конвертувати документи у формат LaTex, який є стандартним форматом переважної більшості наукових видавництв світу.

Пакет **Maple** – найдавніший серед пакетів символьної математики – також дуже популярний у наукових колах. Окрім здійснення аналітичних перетворень, з його допомогою можна розв'язувати різноманітні математичні задачі чисельно. Графічний редактор пакета дозволяє одержувати зображення тривимірних фігур із перерізами та конвертувати документи у формат LaTex.

Пакет **MatLab**, як і згадані пакети, є своєрідною мовою програмування високого рівня, що орієнтована переважно на інженерні розрахунки теорії управління, електро- і радіотехніки, а також моделювання технічних систем. Пакет MatLab став фактично міжнародним стандартом сучасного навчального програмного забезпечення і використовується в багатьох провідних університетах світу [21].

Пакет **MathCad** відрізняється від інших такою особливістю, як використання звичайних математичних позначень. Документ пакета MathCad на екрані монітору виглядає як звичайний математичний розрахунок. Крім того, MathCad є середовищем візуального програмування, тобто не вимагає знання спеціального набору команд. Хоча пакет орієнтований, у першу чергу, на проведення чисельних розрахунків, він має вбудований символьний процесор Maple, що дозволяє виконувати аналітичні перетворювання. Також у пакеті MathCad передбачена можливість створювати зв'язки його документів із документами пакета MatLab.

Система **R** відрізняється від наведених пакетів в основному тим, що є вільним програмним продуктом, який розповсюджується на умові ліцензії GNU. Вона включає в себе об'єктно-орієнтовану мову програмування R і середовище в сукупності з великим набором бібліотек (більш ніж 1 600), які доступні для усіх основних платформ – Linux, Windows і MacOS. Постійно виходять нові версії та розширення, які доступні на офіційному сайті http://cran.r-project.org [33], розробники пишуть для системи R пакети, адаптовані до конкретних галузей. Важливим показником популярності цього програмного продукту є і те, що постійно публікуються нові книги з описом застосування R для різних видів аналізу даних [25; 26]. Мова R вбудовується в усілякі системи. Наприклад, корпорація Oracle інтегрувала мову програмування R у СУБД Oracle Database 11g.

З цих причин саме пакет R використовується у цьому навчальному посібнику для програмування, виконання розрахунків та наочного подання результатів розв'язання. Математичний пакет R все частіше застосовується у ході проведення лабораторних занять із різних навчальних дисциплін, тому тим, хто бажає більш досконально опанувати цей пакет, пропонується звернутися до додаткових джерел [28 – 31].

Треба зазначити, що останнім часом позначилася тенденція до зближення та інтеграції різних математичних пакетів. Так, останні версії пакетів Mathematica і Maple мають потужні засоби для візуального програмування; в пакети MatLab та MathCad включена бібліотека аналітичних перетворень Maple; MathCad дозволяє працювати разом із MatLab і т. д. Тому, за великим рахунком, не має значення, який

математичний пакет застосовується у навчальному процесі та засвоюється студентами. Головне – принципово навчитися використовувати будь-який математичний пакет для розв'язання практичних задач.

1.6. Висновки

1. Чисельні методи є одним із основних апаратів розв'язання математичних задач.

2. Універсальних чисельних методів розв'язання математичної задачі, як правило, не існує. Тому треба розглядати кілька методів.

3. Вибір чисельного методу для розв'язання математичної задачі треба здійснювати з урахуванням його характеристик.

4. Є ряд математичних пакетів, за допомогою яких можна достатньо швидко розв'язати багато відомих математичних задач.

1.7. Контрольні запитання та завдання

1. Назвіть основні групи методів розв'язання математичних задач та їх характеристики.

2. Назвіть основні характеристики чисельних методів. Розкрийте їх суть.

3. Які є основні швидкості збіжності ітераційних методів?

4. Назвіть та охарактеризуйте основні етапи розв'язання практичних задач на комп'ютері.

5. Що називають "ітерацією"? Наведіть загальну схему ітераційного методу.

6. Які методи розв'язання математичних задач називають ітераційними?

7. Які методи розв'язання математичних задач називають чисельними?

8. Дайте визначення поняттям абсолютної та відносної погрішностей.

9. Дайте визначення лінійної і квадратичної швидкості збіжності ітераційного методу.

10. Що є причинами появи погрішності при арифметичних обчисленнях на комп'ютері?

11. Як подані числа в комп'ютері? Вкажіть розмір абсолютної та відносної погрішностей округлення чисел у комп'ютері.

12. Які універсальні математичні пакети ви знаєте? Дайте їх порівняльну характеристику. Для чого вони використовуються?

2. Прямі та ітераційні методи розв'язання систем лінійних алгебраїчних рівнянь

2.1. Постановка задачі

Розглядається система алгебраїчних рівнянь виду:

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

де $a_{ij} \in R^1$, $i = \overline{1,n}$, $j = \overline{1,n}$;

$b_i \in R^1$, $i = \overline{1,n}$ – задані;

$x_j \in R^1$, $j = \overline{1,n}$ – невідомі.

Якщо ввести позначення:

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{pmatrix}, \; b = \begin{pmatrix} b_1 \\ b_2 \\ \cdots \\ b_n \end{pmatrix}, \; x = \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{pmatrix},$$

де A – матриця розмірності $n \times n$;

b – вектор-стовпець розмірності n ($b \in R^n$);

x – вектор-стовпець розмірності n ($x \in R^n$),

то систему (2.1) можна записати в матричному вигляді:

$$A\,x = b. \qquad (2.2)$$

Розв'язати систему (2.2) – це означає знайти таке $x^* \in R^n$ ($x^* = \left(x_i^*\right)_{i=1}^n$), при якому значення лівої частини рівняння тотожно дорівнює правій.

Визначення. Нев'язкою системи (2.2), що відповідає довільному вектору x, називається вектор $r = A\,x - b$. Очивидно, що для розв'язку $x^* \in R^n$ нев'язка ($A\,x - b$) дорівнює нулю.

Для того щоб система (2.2) мала єдиний розв'язок, необхідно й достатньо, щоб $det\,A \neq 0$ [10]. У цьому випадку розв'язок системи (2.2) може бути отриманий за формулами Крамера:

$$x_i = \frac{det(A^{(i)})}{det(A)},\ i = \overline{1,n},$$

де матриця $A^{(i)}$ утворюється з матриці A заміною її i-го стовпця стовпцем вільних членів b.

Але такий спосіб розв'язання системи лінійних рівнянь з n невідомими призводить до обчислення $(n+1)$-го визначника порядку n, що є дуже трудомісткою операцією при великих n.

Чисельні методи розв'язання систем лінійних алгебраїчних рівнянь (2.2) можна розбити на дві групи: прямі (точні) й ітераційні (наближені).

Визначення. Точними (прямими) **методами** називаються методи, які в припущенні, що обчислення ведуться точно (без округлень), приводять за скінчене число кроків до точних значень x_i^*. Оскільки обчислення на комп'ютері ведуться з округленнями, то розв'язок неминуче міститиме погрішності. До прямих методів відносяться, наприклад, метод Гауса, метод Холецького та ін. [3; 5; 6; 21].

Визначення. Наближеними (ітераційними) **методами** називаються такі методи, які навіть у припущенні, що обчислення ведуться без округлень, дозволяють отримати розв'язок x^* тільки із зазначеною точністю. Точний розв'язок системи в цьому випадку може бути отриманий теоретично як результат нескінченного (ітераційного) процесу. До наближених методів відносяться, наприклад, метод ітерацій, метод Зейделя та ін. [3; 5; 6; 21]. Кожний із цих методів не завжди є

збіжним у застосуванні до конкретного класу систем лінійних алгебраїчних рівнянь.

2.2. Метод виключення Гауса

Найбільш відомим із прямих методів розв'язання системи (2.2) є **метод виключення Гауса**, ідея якого полягає в послідовному виключенні невідомих із рівнянь. Спочатку відправна система (2.2) приводиться до системи трикутного вигляду (прямій хід), а потім невідомі визначаються за простими формулами (зворотний хід).

Розрахункові формули методу Гауса:

Нехай $a_{ij}^{(0)} = a_{ij}$, $i = \overline{1,n}$, $j = \overline{1,n}$; $b_i^{(0)} = b_i$, $i = \overline{1,n}$.

Прямий хід: k-й крок

при $k = \overline{1,n-1}$:

$$a_{kj}^{(k)} = \frac{a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}},\ j = \overline{k+1,n};\ b_k^{(k)} = \frac{b_k^{(k-1)}}{a_{kk}^{(k-1)}};$$

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - a_{ik}^{(k-1)}\, a_{kj}^{(k)},\ j = \overline{k+1,n},$$

$$b_i^{(k)} = b_i^{(k-1)} - a_{ik}^{(k-1)}\, b_k^{(k)},\ i = \overline{k+1,n};$$

при $k = n$:

$$b_k^{(k)} = \frac{b_k^{(k-1)}}{a_{kk}^{(k-1)}}.$$

Зворотний хід:

$$x_n = b_n^{(n)};$$

$$x_k = b_k^{(k)} - \sum_{j=k+1}^{n} a_{kj}^{(k)}\, x_j,\ k = \overline{n-1,1}. \qquad (2.3)$$

Тут $A^{(k)} = (a_{ij}^{(k)})$ – це проміжна матриця, а $b^{(k)} = (b_i^{(k)})$ – проміжний вектор на k-му кроці прямого ходу методу Гауса, причому

$A^{(0)} = A$, $b^{(0)} = b$. Елементи $a_{kk}^{(k)} = 1$ для усіх k формально, але виконувати це обчислювальня фактично не потрібно. Також не потрібно обчислювати і піддіагональні (нульові) елементи матриці.

Рядки, що містять одиницю на діагоналі, називаються **виділеними рядками**. Процес отримання виділених рядків (приведення системи до трикутного вигляду) називається **прямим ходом**, а процес знаходження невідомих шляхом використання виділених рядків – **зворотним ходом** методу Гауса.

Приклад 2.1. Розв'язати систему рівнянь методом Гауса.

$$\begin{cases} 5x_1 + x_2 + 2x_3 = 10, \\ 6x_1 + 18x_2 + 6x_3 = 54, \\ 10x_1 + 20x_2 + 40x_3 = 160. \end{cases}$$

Розв'язання в математичному пакеті R

Процедуру розв'язання системи лінійних рівнянь методом Гауса можна записати так:

```
MGaussSLE = function(A,b,n)
{
     for(k in 1:(n-1))
     {
          for(j in (k+1):n)
               A[k,j] = A[k,j]/A[k,k]
          b[k] = b[k]/A[k,k]
          for(i in (k+1):n)
          {
               for(j in (k+1):n)
                    A[i,j] = A[i,j] - A[i,k]*A[k,j]
               b[i] = b[i] - A[i,k]*b[k]
          }
     }
     b[n] = b[n]/A[n,n]
     for(k in (n-1):1)
     {
          for(j in (k+1):n)
               b[k] = b[k] - A[k,j]*b[j]
     }
      return(b)
}
```

Відправними даними задачі (2.1) є матриця A коефіцієнтів при невідомих, вектор-стовпець b вільних членів і кількість невідомих n:

```
> A = cbind(c(5,6,10), c(1,18,20), c(2,6,40))
> b = c(10,54,160)
> n = 3
> A
     [,1] [,2] [,3]
[1,]    5    1    2
[2,]    6   18    6
[3,]   10   20   40
> b
[1]  10  54 160
> n
[1] 3
```

Результат розв'язання заданої системи лінійних рівнянь з використанням записаної процедури:

```
> x = MGaussSLE(A,b,n)
> x
[1] 0.4444444 1.8666667 2.9555556
```

Таким чином, розв'язком заданої системи лінійних рівнянь є знайдені значення невідомих: $x_1 = 0.444$, $x_2 = 1.867$, $x_3 = 2.956$. Для перевірки отриманих результатів знайдені значення невідомих були підставлені у систему:

```
> y = A%*%x
> y
     [,1]
[1,]   10
[2,]   54
[3,]  160
```

Кількість арифметичних операцій, необхідних для реалізації метода Гауса, визначається формулою [6]:

$$K(n) = \frac{2n(n+1)(n+2)}{2} + n(n-1),$$

де n – розмірність системи (2.2), тобто пропорційна кубу числа невідомих ($O(n^3)$).

Слід зазначити, що з логічної точки зору кращою є модифікація методу Гауса, що зветься **метод Гауса – Жордана**. Його суть полягає в тому, що зворотний хід виконується не за формулами (2.3), а проводиться виключення невідомих у зворотному порядку з рівнянь, отриманих після виконання прямого ходу методу Гауса. При цьому система приводиться до діагонального вигляду з одиницями на діагоналі, а розв'язок системи виявляється на місці вектора b. Таким чином, зворотний хід дійсно є зворотним, тобто виконуються операції симетричні відносно прямого ходу.

2.3. Метод Гауса з вибором головного елемента

Стандартний метод Гауса може стати чисельно нестійким, якщо серед $a_{kk}^{(k-1)}$, на які проводиться ділення, виявляються дуже малі числа по абсолютній величині, хоча і відмінні від нуля (не говорячи вже про нульові значення). Тоді при діленні на них отримуються великі числа з великими абсолютними погрішностями. В результаті цього отриманий розв'язок сильно відрізнятиметься від дійсного розв'язку, тобто метод Гауса буде нестійким до помилок обчислень.

Щоб цього уникнути, на практиці застосовують **метод виключення Гауса з вибором головного елемента.** Діагональний елемент $a_{kk}^{(k-1)}$, на який проводиться ділення на k-му кроці, називається **головним** (ведучим) елементом. Якщо головний елемент близький до нуля по абсолютній величині, то можна знайти у відповідному (k-му) стовпці максимальний за модулем елемент і переставити рядки місцями так, щоб цей елемент став головним. Але така перестановка теж не завжди забезпечує стійкість. Тому при програмній реалізації зазвичай вибирають максимальний за модулем елемент не в k-му стовпці, а в усій матриці, що залишилася. Потрібно зазначити, що якщо доводиться переставляти місцями стовпці матриці, то ці перестановки потрібно запам'ятовувати, а потім (після отримання розв'язку) проводити їх у зворотному порядку вже у векторі отриманого розв'язку.

Сенс вибору головного елемента полягає в тому, щоб зробити якомога меншими по абсолютній величині числа $a_{kj}^{(k)} = \dfrac{a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}}$ і, тим

самим, зменшити погрішність обчислень і округлень. Тому для реалізації методу Гауса на комп'ютері зазвичай використовують саме схему з вибором головного елемента.

2.4. LU-розкладання матриці, метод Холецького

Метод LU-розкладання в принципі еквівалентний методу Гауса, відмінність полягає тільки в порядку дій.

У методі LU-розкладання матриця A системи (2.2) спочатку подається у вигляді **LU-розкладання**, тобто у вигляді добутку двох матриць:

$$A = L\,U, \tag{2.4}$$

де L – нижньотрикутна матриця;

U – верхньотрикутна матриця з одиницями на діагоналі.

Тоді розв'язання системи (2.2) проводиться в два етапи: спочатку розв'язується система:

$$L\,y = b, \tag{2.5}$$

відносно $y \in R^n$, а потім вже знаходиться шуканий розв'язок x^* шляхом розв'язання системи:

$$U\,x = y. \tag{2.6}$$

Оскільки матриці L, U – трикутні, то знаходження розв'язків систем (2.5) та (2.6) проводиться за простими формулами, аналогічними формулам зворотного ходу методу Гауса.

Можна ввести позначення:

$$L = \begin{pmatrix} l_{11} & 0 & 0 & \cdots & 0 \\ l_{21} & l_{22} & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & l_{nn} \end{pmatrix}, \; U = \begin{pmatrix} 1 & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & 1 & u_{23} & \cdots & u_{2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Тоді зі співвідношення (2.4) будуть отримані формули для визначення елементів матриць L і U:

$$l_{i1} = a_{i,1},$$

$$l_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}, \quad i \geq j > 1$$

i

$$u_{1j} = \frac{a_{1j}}{l_{11}},$$

$$u_{ij} = \frac{1}{l_{ii}}(a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}), \; 1 < i < j.$$

Приклад 2.2. Розв'язати систему рівнянь з прикладу 2.1 методом LU-розкладання.

Розв'язання в математичному пакеті R

Процедуру розв'язання системи лінійних рівнянь методом LU-розкладання можна записати так:

```
MetLUSLE = function(A,b,n)
{
     L = matrix(0, nrow=n, ncol=n)
     U = matrix(0, nrow=n, ncol=n)

     for(i in 1:n)
          L[i,1] = A[i,1]
     for(j in 1:n)
          U[1, j] = A[1,j]/L[1,1]
     for(i in 1:n)
     {
          for(j in 1:n)
          {
               S = 0
               for(k in 1:i)
                    S = S + L[i,k]*U[k,j]
               if( (i >= j) & (j > 1) )
                    L[i,j] = A[i,j] - S
               if( (j > i) & (i > 1) )
                    U[i,j] = (A[i,j] - S)/L[i,i]
          }
          U[i,i] = 1
     }
      y = solve(L, b)
     x = solve(U, y)
      return(x)
}
```

Відправними даними задачі (2.1) є матриця A коефіцієнтів при невідомих, вектор-стовпець b вільних членів і кількість невідомих n:

```
> A = cbind(c(5,6,10), c(1,18,20), c(2,6,40))
> b = c(10,54,160)
> n = 3
> A
     [,1] [,2] [,3]
[1,]    5    1    2
[2,]    6   18    6
[3,]   10   20   40
> b
[1]  10  54 160
```

Результат розв'язання заданої системи лінійних рівнянь з використанням записаної процедури:

```
> x = MetLUSLE(A,b,3)
> x
[1] 0.4444444 1.8666667 2.9555556
```

Таким чином, розв'язком заданої системи лінійних рівнянь є знайдені значення невідомих: $x_1 = 0.444$, $x_2 = 1.867$, $x_3 = 2.956$.

Слід зазначити, що якщо A – симетрична знакопевна матриця [23], то розкладання (2.4) може бути записане у вигляді:

$$A = U^T D U, \tag{2.7}$$

де U^T – нижньотрикутна матриця, транспонована до U;

D – діагональна матриця $D = \begin{pmatrix} d_{11} & 0 & 0 & \cdots & 0 \\ 0 & d_{22} & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & d_{nn} \end{pmatrix}$, всі d_{ii} мають однаковий знак.

Застосування розкладання (2.7) для розв'язання системи (2.2) має назву **методу Холецького**.

Оскільки проблема розв'язання системи лінійних алгебраїчних рівнянь виду (2.2) з'являється, як правило, як проміжна задача при розв'язанні більш складних задач (розв'язання систем нелінійних рівнянь, оптимізації та ін.), то використання розкладань (2.4) та (2.7)

часто буває корисним для отримання додаткової інформації про відправну задачу. Наприклад, коли матриця A обчислюється як матриця других похідних деякої функції $f(x)$ n змінних у деякій точці $x^{(0)} \in R^n$, то якщо всі діагональні елементи матриці D з розкладання (2.7) додатні, то це означає, що функція $f(x)$ опукла в околі точки $x^{(0)}$.

Розкладання (2.4) може бути використане і для обчислення визначника ($det\, A$) матриці A. Так $det\, A = det(L)\, det(U)$ [10], але визначник трикутної матриці дорівнює добутку діагональних елементів. Тому $det(U) = 1$, оскільки матриця U – трикутна і має одиниці на діагоналі, а $det(L) = \prod_{i=1}^{n} l_{ii}$. Тоді $det(A) = det(L) = \prod_{i=1}^{n} l_{ii}$.

2.5. Метод ітерацій

Як приклад ітераційного (наближеного) методу розв'язання системи (2.2) варто розглянути **метод ітерацій.**

Для застосування методу ітерацій систему (2.2) необхідно подати у вигляді:

$$x = C\,x + d, \tag{2.8}$$

де $C \in R^{n \times n}$, $d \in R^n$.

Алгоритм методу ітерацій.

1. Задається $\varepsilon > 0$ – точність розв'язку задачі і початкове наближення розв'язку $x^{(0)} \in R^n$ (наприклад, $x^{(0)} = d$).

2. На k-й ітерації методу ($k = 0, 1, 2, \ldots$) обчислюється наступне наближення:

$$x^{(k+1)} = C\,x^{(k)} + d. \tag{2.9}$$

3. Перевіряється критерій останову $\frac{q}{1-q}\left\| x^{(k+1)} - x^{(k)} \right\| \le \varepsilon$, де $0 < q < 1$ (визначається з умови збіжності).

Метод простої ітерації збігається тільки при виконанні умови:

$$\|C\| \le q < 1, \qquad (2.10)$$

де $0 < q < 1$ [3; 5; 6; 19; 21]. При цьому як норму матриці C можна розглядати величину

$$\|C\| \equiv \underbrace{max}_{j} \sum_{i=1}^{n} |c_{ij}| \text{ або } \|C\| \equiv \underbrace{max}_{i} \sum_{j=1}^{n} |c_{ij}|. \qquad (2.11)$$

Тоді метод ітерацій збігається з лінійною швидкістю збіжності, тобто $\left\|x^{k+1} - x^*\right\| \le q\left\|x^k - x^*\right\|$ для всіх $k \ge 0$. Справедлива також оцінка $\left\|x^{k+1} - x^*\right\| \le \frac{q}{1-q}\left\|x^{k+1} - x^k\right\|$ для всіх $k \ge 0$, яка і використовується в критерії останову [4].

Приведення системи (2.2) до виду (2.8) можна здійснити різними способами, важливо тільки, щоб виконувалася умова збіжності (2.10) і (2.11). Варто розглянути один із них.

Якщо діагональні елементи матриці відмінні від нуля, тобто $a_{ii} \ne 0$, $i = \overline{1,n}$, то систему (2.1) можна записати у вигляді

$$\begin{cases} x_1 = \frac{1}{a_{11}}(-a_{12}x_2 - a_{13}x_3 - \ldots - a_{1n}x_n + b_1), \\ x_2 = \frac{1}{a_{22}}(-a_{21}x_1 - a_{23}x_3 - \ldots - a_{2n}x_n + b_2), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = \frac{1}{a_{nn}}(-a_{n1}x_1 - a_{n3}x_3 - \ldots - a_{n,n-1}x_{n-1} + b_n). \end{cases}.$$

У цьому випадку елементи матриці C дорівнюють $c_{ij} = -\frac{a_{ij}}{a_{ii}}$, $i \ne j$, $i = \overline{1,n}$, $j = \overline{1,n}$, $c_{ii} = 0$. Елементи вектора d при цьому будуть дорівнювати $d_i = \frac{b_i}{a_{ii}}$, $i = \overline{1,n}$. Тобто

$$C = \begin{pmatrix} 0 & -\frac{a_{12}}{a_{11}} & -\frac{a_{13}}{a_{11}} & \dots & -\frac{a_{1n}}{a_{11}} \\ -\frac{a_{21}}{a_{22}} & 0 & -\frac{a_{23}}{a_{22}} & \dots & -\frac{a_{2n}}{a_{22}} \\ \dots & \dots & \dots & \dots & \dots \\ -\frac{a_{n1}}{a_{nn}} & -\frac{a_{n2}}{a_{nn}} & -\frac{a_{n3}}{a_{nn}} & \dots & 0 \end{pmatrix}, \; d = \begin{pmatrix} \frac{b_1}{a_{11}} \\ \frac{b_2}{a_{22}} \\ \dots \\ \frac{b_n}{a_{nn}} \end{pmatrix}.$$

Тоді умова збіжності (2.10) і (2.11) виконуватиметься, якщо

$$\sum_{\substack{i=1 \\ i \neq j}}^{n} \left| \frac{a_{ij}}{a_{ii}} \right| \leq q < 1, \; j = \overline{1,n} \text{ або } \sum_{\substack{j=1 \\ j \neq i}}^{n} \left| \frac{a_{ij}}{a_{ii}} \right| \leq q < 1, \; i = \overline{1,n}. \qquad (2.12)$$

У свою чергу, ці нерівності будуть справедливі, якщо діагональні елементи матриці A задовольняють умову $|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|$, $i = \overline{1,n}$, тобто модулі діагональних коефіцієнтів для кожного рівняння системи (2.1) більше суми модулів решти всіх коефіцієнтів.

Слід зазначити, що описана процедура приведення системи (2.2) до виду (2.8) з подальшим застосуванням методу ітерацій називається **методом Якобі**.

Приклад 2.3. Розв'язати систему рівнянь методом Якобі з заданою точністю $\varepsilon = 10^{-5}$

$$\begin{cases} 5x_1 + x_2 + 2x_3 = 10, \\ 6x_1 + 18x_2 + 6x_3 = 54, \\ 10x_1 + 20x_2 + 40x_3 = 160. \end{cases}$$

Розв'язання в математичному пакеті R

Відправними даними задачі є матриця коефіцієнтів при невідомих A, вектор-стовпець вільних членів b, кількість невідомих n, точність

обчислення ε. Для запобігання закциклювання ітераційного процесу слід задати додатково максимальну кількість ітерацій методу $k\,max$:

```
> A = cbind(c(5,6,10), c(1,18,20), c(2,6,40))
> b = c(10,54,160)
> n = 3
> eps = 0.0001
> kmax = 100

> A
     [,1] [,2] [,3]
[1,]    5    1    2
[2,]    6   18    6
[3,]   10   20   40

> b
[1]  10  54 160
> n
[1] 3
> eps
[1] 1e-04
> kmax
[1] 100
```

Процедуру розв'язання системи лінійних алгебраїчних рівнянь виду (2.8) методом ітерацій можна записати так:

```
MetIterSLE = function(C,d,n,eps,kmax)
{
    x0 = d
    for (k in 1:kmax)
    {
        x = C%*%x0 + d
        if( norm(x-x0) <= eps )
            break
        x0 = x
    }
    return(x)
}
```

Тоді процедуру розв'язання системи лінійних алгебраїчних рівнянь виду (2.8) методом Якобі можна записати так:

```
MetJakobiSLE = function(A,b,n,eps,kmax)
{
    for (i in 1:n)
    {
        for (j in 1:n)
        {
            C[i,j] = -A[i,j]/A[i,i]
        }
        C[i,i] = 0
        d[i] = b[i]/A[i,i]
    }
    x = MetIterSLE(C,d,n,eps,kmax)
    return (x)
}
```

Результат розв'язання заданої системи лінійних рівнянь із використанням записаних процедур:

```
> x = MetJakobiSLE(A,b,n,eps,kmax)
> x
          [,1]
[1,] 0.4444534
[2,] 1.8666762
[3,] 2.9555659
```

Перевірка розв'язку – обчислення нев'язки $A\,x - b$:

```
> y = A%*%x -b
> y
              [,1]
[1,] 7.486361e-05
[2,] 2.867975e-04
[3,] 6.931325e-04
```

Таким чином, розв'язком поставленої задачі є знайдені значення невідомих: $x_1 = 0.444$, $x_2 = 1.867$, $x_3 = 2.956$. Для перевірки отриманих результатів обчислена нев'язка $(A\,x - b)$ для системи (2.2) при знайдених значеннях невідомих, тобто розв'язок отримано з точністю 10^{-4}, як і було задано.

Метод Гауса для розв'язання системи (2.2) застосовують на практиці при $n \le 10^3$, метод простої ітерації – при $10^3 < n \le 10^6$. Метод простої ітерації також застосовують для уточнення розв'язку, отриманого методом Гауса.

2.6. Метод Гауса – Зейделя

Метод Гауса – Зейделя є модіфікацією метода Якобі, який було описано у розділі 2.5 для розв'язання системи (2.2). Він також застосовується для матриць A, для яких виконується умова $a_{ii} \neq 0$ для всіх i. По суті це той самий ітераційний процес, що і (2.9), але в ньому нове наближення $x^{(k+1)}$ застосовується одразу ж, як змінився його i-й елемент. Ітераційна формула методу Гауса – Зейделя має вигляд:

$$x_i^{(k+1)} = \frac{b_i - \sum_{j=1}^{i-1} a_{ij}\, x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}\, x_j^{(k)}}{a_{ii}},\ i = \overline{1,n}.$$

Умови збіжності методу Гауса – Зейделя ті ж самі, що й у методу Якобі, тобто (2.12), але на практиці швидкість його збігу дещо вища. Тому саме цей метод застосовують найчастіше.

2.7. Обчислення оберненої матриці

Визначення. **Оберненою** до матриці A називається така матриця B, для якої виконується рівність:

$$AB = BA = E, \tag{2.13}$$

де E – одинична матриця ($E \in R^{nxn}$), тобто

$$E = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Обернену матрицю до A прийнято позначати через A^{-1}.

З умови (2.13) виходить, що обернена матриця A^{-1} вводиться тільки для квадратних матриць A, при цьому A^{-1} також буде квадратною тієї ж розмірності.

Квадратна матриця A називається **невиродженою** або **неособливою (несингулярною)**, якщо її визначник $\det A$ відмінний від нуля. Будь-яка невироджена матриця має обернену матрицю.

Якщо відома обернена матриця A^{-1}, то розв'язок системи (2.2) записується у вигляді $x = A^{-1} b$. Слід зазначити, що з практичної точки зору все ж таки ефективнішим буде розв'язання системи рівнянь, ніж обчислення оберненої матриці з подальшим множенням на вектор b.

Метод Гауса може бути застосований для обчислення оберненої матриці. Нехай дана невироджена матриця $A = (a_{ij})$ розмірності $n \times n$. Елементи шуканої оберненої матриці A^{-1} можна позначити через (x_{ij}),а також ввести позначення x^j – j-й стовпець ($x^j \in R^n$, $j = \overline{1,n}$) матриці $X = A^{-1}$, e^j – j-й стовпець ($e^j \in R^n$, $j = \overline{1,n}$) матриці E, тобто:

$$x^j = \begin{pmatrix} x_{1j} \\ x_{2j} \\ \dots \\ x_{nj} \\ \end{pmatrix}, \quad e^j = \begin{pmatrix} 0 \\ \dots \\ 1 \\ \dots \\ 0 \end{pmatrix}, \ (e^j_j = 1).$$

Тоді рівність $AX = E$ можна записати у вигляді:

$$A x^j = e^j, \ (j = \overline{1,n}), \tag{2.14}$$

тобто стовпці x^j ($j = \overline{1,n}$) можуть бути знайдені методом Гауса як розв'язок n систем лінійних рівнянь з однією і тією ж матрицею A, але з різними векторами правих частин e^j. Звідси випливає, що прямий хід методу Гауса при розв'язанні систем (2.14) буде загальним, тобто його можна проводити одночасно для всіх j, а потім вже визначити вектори x^j ($j = \overline{1,n}$), виконавши n разів зворотний хід методу Гауса.

Треба нагадати (див. розділ 2.2), що виконання зворотного ходу методу Гауса – Жордана еквівалентне приведенню трикутної системи, отриманої після прямого ходу, до системи з діагональною одиничною

матрицею. Таким чином, приводячи початкову матрицю A до одиничної матриці і виконуючи аналогічні дії з елементами матриці E, ця матриця буде перетворена в матрицю A^{-1}. Звідси випливає, що обчислення оберненої матриці ефективно проводити саме за методом Гауса – Жордана.

Приклад 2.4. Знайти обернену матрицю до матриці A з прикладу 2.1 методом Гауса – Жордана.

Розв'язання в математичному пакеті R

Відправними даними задачі є матриця A:

```
>  A = cbind(c(5,6,10), c(1,18,20), c(2,6,40))
>  A
      [,1] [,2] [,3]
[1,]    5    1    2
[2,]    6   18    6
[3,]   10   20   40
```

Процедуру обчислення оберненої матриці методом Гауса – Жордана можна записати так:

```
MGaussJord = function(A,n)
{
     # створюємо одиничну матрицю E
     E = matrix(0, nrow=n, ncol=n)
     for(i in 1:n)
          E[i,i] = 1
     # приводимо матрицю A до трикутного виду (метод Гауса)
     for(k in 1:n)
     {
          for(j in 1:n)
               E[k,j] = E[k,j]/A[k,k]
          if(k==n)
               break
          for(j in (k+1):n)
               A[k,j] = A[k,j]/A[k,k]
          for(i in (k+1):n)
          {
               for(j in (k+1):n)
                    A[i,j] = A[i,j] - A[i,k]*A[k,j]
               for(j in 1:n)
                    E[i,j] = E[i,j] - A[i,k]*E[k,j]
          }
     }
```

```
    # приводимо матрицю A до диагонального виду (метод Жордана)
    for(k in (n-1):1)
    {
        for(i in (k+1):n)
            for(j in 1:n)
                E[k,j] = E[k,j] - A[k,i]*E[i,j]
    }
     return(E)
}
```

Результат обчислення оберненої матриці з використанням записаної процедури:

```
> A = cbind(c(5,6,10), c(1,18,20), c(2,6,40))
> A
     [,1] [,2] [,3]
[1,]    5    1    2
[2,]    6   18    6
[3,]   10   20   40
> A1 = MGaussJord(A,3)
> A1
             [,1]           [,2]          [,3]
[1,]  0.22222222 -3.469447e-18 -0.011111111
[2,] -0.06666667  6.666667e-02 -0.006666667
[3,] -0.02222222 -3.333333e-02  0.031111111
> E1 = A%*%A1
> E1
              [,1]          [,2]          [,3]
[1,] 1.000000e+00 0.000000e+00 -6.938894e-18
[2,] 0.000000e+00 1.000000e+00  2.775558e-17
[3,] 2.220446e-16 2.220446e-16  1.000000e+00
> A2 = solve(A)
> A2
             [,1]        [,2]          [,3]
[1,]  0.22222222  0.00000000 -0.011111111
[2,] -0.06666667  0.06666667 -0.006666667
[3,] -0.02222222 -0.03333333  0.031111111
```

Для перевірки отриманих результатів перемножені матриці *A* і *A1*. Добуток *E1* дорівнює одиничній матриці, що підтверджує правильність результату. Також обчислена обернена матриця *A2* з використанням процедури *solve* пакета **R**. Як видно, результати збігаються з точністю до помилки обчислень.

2.8. Висновки

1. Розв'язання систем лінійних алгебраїчних рівнянь є базовою процедурою в математичних пакетах чисельного аналізу, оскільки в багатьох чисельних методах вона використовується як допоміжна. Тому питанню оптимізації базової процедури приділяється особлива увага.

2. Якщо розв'язання системи лінійних алгебраїчних рівнянь є проміжною задачею при розв'язанні більш складної задачі, то найчастіше для цього застосовують метод LU-розкладання матриць.

3. Обчислення оберненої матриці ефективно проводити за методом Гауса – Жордана.

2.9. Контрольні запитання та завдання

1. Сформулюйте постановку задачі розв'язання системи лінійних алгебраїчних рівнянь.

2. Назвіть групи методів розв'язання систем лінійних алгебраїчних рівнянь, вкажіть їх принципові відмінності.

3. У чому полягає ідея методу Гауса?

4. У чому полягає суть методу виключення Гауса з вибором головного елемента? В яких випадках його застосовують на практиці?

5. Що називається LU-розкладанням матриці?

6. Як застосовується LU-розкладання матриць для розв'язання систем лінійних алгебраїчних рівнянь?

7. Як можна використати LU-розкладання матриці для обчислення її оберненої матриці?

8. Опишіть схему обчислення оберненої матриці методом Гауса – Жордана.

9. При яких умовах збігається метод простої ітерації? Наведіть оцінки швидкості збіжності цього методу.

10. Який порядок арифметичних обчислень потрібен для розв'язання системи лінійних алгебраїчних рівнянь з n невідомими?

11. Знайдіть коефіцієнти параболи $y = ax^2 + bx + c$, яка проходить через точки (1; 8.6), (3; 30.8), (5; 65).

12. Знайдіть обернену матрицю для матриці A:

$$A = \begin{pmatrix} 1.8 & -3.8 & 0.7 & -3.7 \\ 0.7 & 2.1 & -2.6 & -2.8 \\ 7.3 & 8.1 & 1.7 & -4.9 \\ 1.9 & -4.3 & -4.9 & -4.7 \end{pmatrix}.$$

13. Знайдіть точку перетину двох функцій:

$$y = 1 - 0.3x \text{ і } x = 3 - 2.2y .$$

3. Розв'язання систем лінійних рівнянь великої розмірності

3.1. Постановка задачі

Як зазначалося у розділі 2.5, класичні прямі чисельні методи розв'язання задачі (2.2) є сенс застосовувати при розмірності системи до 1 000. Але на на практиці нерідко виникає необхідність розв'язувати системи і з більшою розмірністю. Такі системи вважаються **системами великої розмірністі**. Наприклад, такого роду системи виникають при проектуванні великих інтегральних схем та ін. При роз'вязанні систем великої розмірністі на практиці з застосуванням комп'ютерної техніки виникають такі основні проблеми:

1) великий об'єм оперативної пам'яті комп'ютера, необхідний при проведенні розрахунків із матрицею;
2) великий об'єм самих розрахунків;
3) матриця A системи може бути виродженою;
4) накопичення помилок обчислення.

Наприклад, якщо матриця має розмірність $n = 10^6$, то для її зберігання необхідно $n \times n \times 8$ байт, тобто порядку 10^{13} байт, або 10^4 Гігабайт оперативної памяті комп'ютера. Очевидно, що це можливо реалізувати тільки на суперкомп'ютерах. Великий об'єм самих розрахунків приводить до того, що для розв'язання задачі необхідно багато часу. Також при цьому відбувається накопичення помилок обчислення, що призводить до неправильного результату.

Для вирішення цих проблем застосовують деякі прийоми та спеціальні чисельні методи, які описані далі.

3.2. Види розріджених матриць

Визначення. Розріджені матриці – це матриці, кількість нульових елементів в яких є таким великим, що стає ефективним облік структури цих матриць.

Розріджені матриці дуже часто зустрічаються в інженерних задачах, наприклад, у задачах проектування виликих інтегральних схем, у задачах моделювання складних об'єктів та оптимального управління ними.

Для економії комп'ютерної пам'яті при зберіганні розріджених матриць застосовують спеціальне кодування, а саме зберігають тільки ненульові елементи та їх місця розташування в матриці.

Варто розглянути приклад. Нехай є матриця розмірності 6x6.

$$A = \begin{pmatrix} a_{11} & 0 & 0 & a_{14} & 0 & 0 \\ 0 & a_{22} & 0 & 0 & 0 & 0 \\ a_{31} & 0 & a_{33} & a_{34} & 0 & a_{36} \\ 0 & 0 & a_{43} & a_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & a_{55} & 0 \\ 0 & 0 & a_{63} & 0 & 0 & a_{66} \end{pmatrix}.$$

Для її зберігання звичайним засобом потрібно 6x6x(8 байт) = 288 байт пам'яті. Але якщо зберігати тільки ненульові елементи матриці, то буде достатньо масивів:

$$Ae = (a_{11}\ a_{31}\ a_{22}\ a_{33}\ a_{43}\ a_{63}\ a_{14}\ a_{34}\ a_{44}\ a_{55}\ a_{36}\ a_{66}),$$

$$Am = (\{1,1\}\ \{3,1\}\ \{2,2\}\ \{3,3\}\ \{4,3\}\ \{6,3\}\ \{1,4\}\ \{3,4\}\ \{4,4\}\ \{5,5\}\ \{3,6\}\ \{6,6\}).$$

В одновимірному масиві Ae зберігаються ненульові елементи матриці A по стовпцям, при цьому потрібно 12x(8 байт) = 96 байт. В масиві Am зберігаються місця розташування ненульових елементів матриці, при цьому потрібно 12x2x(2 байта) = 48 байт. Таким чином, для зберігання матриці A в кодованому вигляді потрібно загалом 144 байти, тобто у два рази менше. При великій розмірності ця різниця стає дуже великою.

Описаний спосіб зберігання розріджених матриць застосовують при відсутності будь-яких закономірностей в структурі розташування нульових та ненульових елементів. Але існують матриці спеціальних видів, котрі досить часто зустрічаються на практиці.

Види розріджених матриць. Якщо матриця $A \in R^{n \times n}$ має вигляд

$$A = \begin{pmatrix} * & 0 & * & 0 & 0 & 0 \\ 0 & * & 0 & * & 0 & 0 \\ * & 0 & * & 0 & * & 0 \\ 0 & * & 0 & * & 0 & * \\ 0 & 0 & * & 0 & * & 0 \\ 0 & 0 & 0 & * & 0 & * \end{pmatrix},$$

тобто ненульові елементи знаходяться тільки на головній діагоналі та деяких інших діагоналях матриці, то вона називається **стрічковою** [21]. В цьому випадку зберігаються значення ненульових діагоналей у вигляді матриці розмірності $n \times k$, де k – кількість цих діагоналей, та їх порядковий номер відносно головної діагоналі. Діагоналі розташовані над головною діагональю нумеруються додатніми значеннями, а під головною діагональю – від'ємними значеннями. При цьому номер головної діагоналі дорівнює нулю [21].

Якщо стрічкова матриця $A \in R^{n \times n}$ має вигляд

$$A = \begin{pmatrix} * & * & 0 & 0 & 0 & 0 \\ * & * & * & 0 & 0 & 0 \\ 0 & * & * & * & 0 & 0 \\ 0 & 0 & * & * & * & 0 \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{pmatrix},$$

тобто ненульові елементи знаходяться тільки на головній діагоналі та двох сусідних діагоналях матриці, то вона називається **трьохдіагональною.**

Якщо стрічкова матриця $A \in R^{n \times n}$ має вигляд

$$A = \begin{pmatrix} * & * & * & 0 & 0 & 0 \\ * & * & * & * & 0 & 0 \\ * & * & * & * & * & 0 \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \end{pmatrix},$$

тобто ненульові елементи знаходяться тільки на головній діагоналі та чотирьох сусідніх діагоналях матриці, то вона називається **п'ятидіагональною.**

Трьох та п'ятидіагональні матриці виникають, наприклад, при розв'язанні крайових задач для звичайних диференціальних рівнянь методом кінцевих різниць (розділ 13.2).

Якщо матриця $A \in R^{n \times n}$ має вигляд

$$A = \begin{pmatrix} * & * & * & * & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \\ 0 & 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & 0 & 0 & * \end{pmatrix},$$

тобто ненульові елементи знаходяться тільки на головній діагоналі та вище неї, то вона називається **верхньою трикутною**. Аналогічно є також і **нижні трикутні матриці.** В цьому випадку зберігаються значення ненульових елементів у вигляді одномірного масиву по стовпцях.

Слід розглянути приклад. Нехай є матриця розмірності 4x4

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{pmatrix}.$$

Для її зберігання звичайним засобом потрібно 4x4x(8 байт) = 128 байт пам'яті. Але якщо сберігати тільки ненульові елементи матриці, то буде достатньо одномірного масива

$$Ae = (a_{11} \quad a_{12} \quad a_{22} \quad a_{13} \quad a_{23} \quad a_{33} \quad a_{14} \quad a_{24} \quad a_{34} \quad a_{44})$$

і при цьому потрібно лише 10x(8 байт) = 80 байт. При великій розмірності ця різниця стає більш значущою.

Слід зазначити, що при проведенні розрахунків замість звернення до довільного елемента a_{ij} повної матриці A треба звертатися до елемента $ae_{j*(j-1)/2+i}$ вектора Ae.

3.3. Методи розв'язання систем лінійних рівнянь великої розмірності з розрідженими матрицями

Нехай матриця $A \in R^{n \times n}$ системи (2.2) є трьохдіагональною. Можна позначити її елементи таким чином:

$$A = \begin{pmatrix} \alpha_1 & \gamma_1 & 0 & \dots & \dots & 0 \\ \beta_2 & \alpha_2 & \gamma_2 & 0 & \dots & 0 \\ 0 & \beta_3 & \alpha_3 & \gamma_3 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & \beta_{n-1} & \alpha_{n-1} & \gamma_{n-1} \\ 0 & \dots & \dots & 0 & \beta_n & \alpha_n \end{pmatrix}.$$

Метод LU-розкладання та метод прогонки дозволяють розв'язувати систему (2.2) з **трьохдіагональною** матрицею, не проводячи дій з нульовими елементами матриці A. Це дає можливість навіть при великій розмірності системи економити на обчисленнях і, як наслідок цього, отримати розв'язок системи з меншою погрішністю.

LU-розкладання для трьохдіагональної матриці A, тобто $A = LU$ або

$$\begin{pmatrix} \alpha_1 & \gamma_1 & 0 & 0 & \dots & 0 \\ \beta_2 & \alpha_2 & \gamma_2 & 0 & \dots & 0 \\ 0 & \beta_3 & \alpha_3 & \gamma_3 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & \beta_{n-1} & \alpha_{n-1} & \gamma_{n-1} \\ 0 & \dots & 0 & 0 & \beta_n & \alpha_n \end{pmatrix} =$$

$$= \begin{pmatrix} \sigma_1 & 0 & 0 & \dots & 0 \\ \beta_2 & \sigma_2 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \beta_{n-1} & \sigma_{n-1} & 0 \\ 0 & \dots & 0 & \beta_n & \sigma_n \end{pmatrix} \times \begin{pmatrix} 1 & \delta_1 & 0 & \dots & 0 \\ 0 & 1 & \delta_2 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & \sigma_1 & \delta_{n-1} \\ 0 & \dots & 0 & 0 & 1 \end{pmatrix}$$

реалізується формулами:

$$\sigma_1 = \alpha_1,\ \delta_1 = \frac{\gamma_1}{\sigma_1},\ \delta_i = \frac{\gamma_i}{\sigma_i},\ \sigma_i = \alpha_i - \beta_i \delta_{i-1},\ i = \overline{2,n}.$$

Тоді розв'язання системи (2.2) виконується у два етапи: $L\,y = b$, $U\,x = y$, які реалізуються формулами:

$$y_1 = \frac{b_1}{\sigma_1},\ y_i = \frac{b_i - \beta_i y_{i-1}}{\sigma_i},\ i = \overline{2,n};$$

$$x_n = y_n,\ x_i = y_i - \delta_i x_{i+1},\ i = \overline{n-1,1}.$$

Є також варіант методу LU-розкладання для розв'язання системи (2.2) з п'ятидіагональною матрицею [21].

Метод прогонки по суті є модифікацією методу Гауса, в якій застосовується облік стрічкової структури матриці системи, тобто, як і у методі LU-розкладання, не виконуються дії з нульовими елементами матриці A.

Метод прогонки для розв'язання системи (2.2) з трьохдіагональною матрицею реалізується формулами:

а) прямий хід

$$v_1 = -\frac{\gamma_1}{\alpha_1},\ w_i = -\frac{\gamma_i}{\alpha_i + \beta_i w_{i-1}},\ i = \overline{2,n};$$

$$v_1 = \frac{b_1}{\alpha_1},\ v_i = \frac{b_i - \beta_i v_{i-1}}{\alpha_i + \beta_i w_{i-1}},\ i = \overline{2,n};$$

б) зворотний хід

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

Є також варіант методу прогонки для розв'язання системи (2.2) з п'ятидіагональною матрицею [21].

3.4. Висновки

1. Розв'язання систем лінійних алгебраїчних рівнянь є базовою процедурою в математичних пакетах чисельного аналізу, оскільки в багатьох чисельних методах вона використовується як допоміжна. Тому питанню оптимізації базової процедури приділяється особлива увага.

2. У задачах розв'язання систем лінійних алгебраїчних рівнянь великої розмірності матриця системи є, як правило, розрідженою.

3. Розв'язання крайових задач для диференціальних рівнянь методом кінцевих різниць приводить до розв'язання системи лінійних алгебраїчних рівнянь з трьохдіагональною або п'ятидіагональною матрицею великої розмірності.

4. Основними чисельними методами розв'язання систем лінійних алгебраїчних рівнянь із трьохдіагональною або п'ятидіагональною матрицею є методи LU-розкладання та метод прогонки.

3.5. Контрольні запитання та завдання

1. Сформулюйте постановку задачі розв'язання системи лінійних алгебраїчних рівнянь.
2. Назвіть групи методів розв'язання систем лінійних алгебраїчних рівнянь, вкажіть їх принципові відмінності.
3. Які матриці називаються розрідженими? Назвіть види розрідженних матриць.
4. У чому полягає суть методу прогону? В яких випадках його застосовують на практиці?
5. Що називається LU-розкладанням матриці?
6. Коли застосовується LU-розкладання матриць для розв'язання систем лінійних алгебраїчних рівнянь великої розмірності?

4. Чисельні методи розв'язання нелінійних рівнянь

4.1. Чисельні методи розв'язання нелінійних рівнянь з одним невідомим

Розглядається рівняння виду:

$$f(x) = 0, \tag{4.1}$$

де $f(x)$ – нелінійна неперервна функція однієї змінної, тобто $f : R^1 \to R^1$.

Розв'язати рівняння (4.1) – означає знайти таке $x^* \in R^1$, для якого $f(x^*) \equiv 0$. При цьому x^* називають **коренем рівняння**.

У загальному випадку рівняння (4.1) може мати багато коренів. Чисельні методи розв'язання нелінійних рівнянь, які розглянуто далі,

дозволяють знаходити один корінь на заданому відрізку $[a,b]$. При цьому на інтервалі повинен існувати тільки один корінь. Знайти відрізок, що задовольняє цю умову можна різними способами:

а) з фізичних міркувань, тобто на основі фізичних знань про задачу;

б) на основі досвіду розв'язання аналогічних задач;

в) за допомогою графічних методів;

г) шляхом відокремлення коренів.

Якщо функція $f(x)$ заздалегідь відома, то тайбільш ефективним є графічний спосіб пошуку відрізка $[a,b]$. В інших випадках, коли відрізок $[a,b]$ треба найти автоматично (не візуально), то застосовують алгоритм відокремлення коренів.

Алгоритм відокремлення коренів

Відокремити корінь – це означає вказати на осі Ох відрізок $[a,b]$, який містить лише один корінь. Алгоритм відокремлення коренів розбивається на такі кроки:

1. Розбивається деяка область на деяку кількість рівних відрізків $[x_i, x_{i+1}]$, де x_0 задано, $x_{i+1} = x_i + h$, h – крок розбиття, $i = 0,\ldots,n$.

2. Визначається знак функції $f(x)$ на кінцях кожного i-го відрізка $[x_i, x_{i+1}]$.

3. Якщо добуток $f(x_i)f(x_{i+1})$ додатний і h достатньо мале, то можна сподіватися, що на відрізку $[x_i, x_{i+1}]$ коренів рівняння немає.

4. Якщо добуток $f(x_i)f(x_{i+1})$ від'ємний, то на відрізку $[x_i, x_{i+1}]$ існує розв'язок рівняння, хоча можливо, що він не один.

5. Перевіряється, чи змінює знак похідна $f'(x)$ на кінцях відрізка $[x_i, x_{i+1}]$. Якщо h достатньо мале і знак похідної $f'(x)$ на кінцях відрізка $[x_i, x_{i+1}]$ не змінюється, то можна сподіватися, що на відрізку корінь один. Таким чином, корінь рівняння відокремлений.

6. Якщо знак похідної $f'(x)$ змінюється, то впевненості, що корінь один немає.

7. Відрізки, для яких немає упевненості в тому, що розв'язок рівняння лише один, продовжують розбивати вже з меншим кроком, тобто повторюється для них описана процедура відокремлення коренів.

Варто розглянути декілька методів розв'язання нелінійного рівняння (4.1) на відрізку $[a,b]$.

4.2. Метод дихотомії

Метод дихотомії ще називають **методом половинного поділу**. При розв'язанні нелінійного рівняння методом половинного поділу задаються відрізок $[a,b]$, на якому існує лише один розв'язок, і бажана точність $\varepsilon > 0$ розв'язання задачі.

Якщо на 0-й ітерації методу позначити $[a_0,b_0]=[a,b]$, то на k-й ($k=0,1,2,\ldots$) ітерації методу буде поточний відрізок $[a_k,b_k]$. Далі визначається середина відрізка $x_k=\frac{a_k+b_k}{2}$ і перевіряється умова $f(a_k)f(x_k)<0$. Якщо вказана умова виконується, то $b_{k+1}=x_k$, $a_{k+1}=a_k$. Якщо умова не виконується, то $a_{k+1}=x_k$, $b_{k+1}=b_k$.

Ділення відрізка $[a_k,b_k]$ навпіл триває доти, доки не виконається умова $|b_k-a_k|\le\varepsilon$. Тут x_k є наближенням розв'язку x^* на k-ій ітерації методу. Очевидно, що тоді точка $x_k=\frac{a_k+b_k}{2}$ відрізнятиметься від точного розв'язку $x^*\in[a,b]$ не більше, ніж на $\frac{\varepsilon}{2}$.

Графічна інтерпретація розв'язання нелінійних рівнянь методом половинного поділу наведена на рис. 4.1.

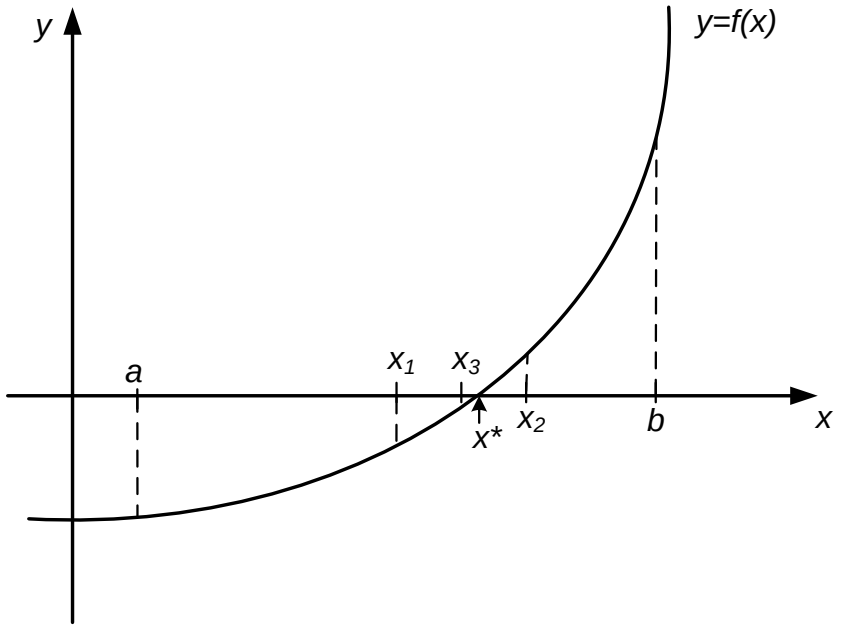


Рис. 4.1. **Графічна інтерпретація розв'язання нелінійних рівнянь методом половинного поділу**

Справедлива така **оцінка швидкості збіжності** [1]:

$$\left|x_k - x^*\right| \le \left|b_k - a_k\right| \le \frac{\left|b-a\right|}{2^{k+1}},$$

тобто метод половинного поділу збігається з лінійною швидкістю (зі швидкістю геометричної прогресії) з коефіцієнтом $q = \frac{1}{2}$.

Слід зазначити, що метод половинного поділу є методом 0-го порядку, оскільки не використовує обчислення похідних функції $f(x)$.

Приклад 4.1. Розв'язати нелінійне рівняння $x^3 + 2 = 5x$ методом дихотомії з точністю $\varepsilon = 10^{-5}$.

Розв'язання в математичному пакеті R

Оскільки метод дихотомії призначений для розв'язання нелінійного рівняння у загальному вигляді (4.1), то спершу треба привести задане рівняння до виду (4.1), тобто перенести всі члени рівняння у ліву частину.

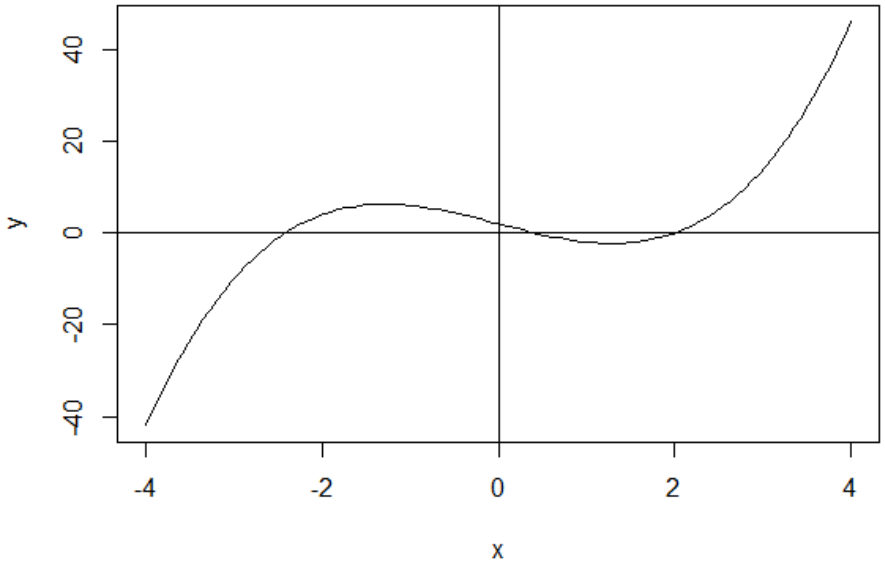
Відправними даними задачі є функція $F(x)$ та точність розв'язку ε:

$$F(x) = x^3 - 5x + 2,\ \varepsilon = 10^{-6}.$$

Оскільки метод дихотомії дозволяє знаходити розв'язки нелінійних рівнянь на відрізку, що містить тільки один корінь, то спершу необхідно визначити такі відрізки. Скористаємося графічним методом.

```
# рівняння (функція)

F = function(x)
{
  return (x^3-5*x+2)
}
# побудова графіка функції

x = seq(-4, 4, len=100)
y = F(x)
plot(x, y, type="l")
abline(h=0)
abline(v=0)
```

Графік заданої функції $F(x)$ має такий вигляд:



Розв'язками рівняння виду (4.1) є точки перетину графіка функції з віссю абсцис. Звідси задане рівняння має розв'язки на інтервалах: $[-3,-2]$, $[0,1]$, $[1,2.5]$.

Знайдені відрізки $[a,b]$ почергово будуть додатковими відправними даними.

Відправні дані (функція, точність розв'язку, знайдені відрізки):

```
F = function(x)
{
  return (x^3-5*x+2)
}
eps = 10^(-6)
a1 = -4
b1 = -2
a2 = 0
b2 = 1
a3 = 1
b3 = 3.5
```

Процедура розв'язання нелінійного рівняння методом дихотомії може бути записана так:

```
MDihotom = function(f, a, b, eps)
{  while (abs(b-a) > eps)
   {
    x = (a+b)/2
    if (f(a)*f(x) < 0) b = x
    else a = x
   }
  return ((a+b)/2)
}
```

Результат розв'язання нелінійного рівняння з використанням записаної процедури:

```
> x1 = MDihotom(F, a1, b1, eps) # виклик функції користувача
> x1
[1] -2.414214
> x2 = MDihotom(F, a2, b2, eps) # виклик функції користувача
> x2
[1] 0.4142137
> x3 = MDihotom(F, a3, b3, eps) # виклик функції користувача
> x3
[1] 2
```

Таким чином, розв'язками заданого рівняння є знайдені значення невідомого: $x1 = -2.414$, $x2 = 0.414$, $x3 = 2$.

Для перевірки отриманих результатів знайдені розв'язки підставляються в рівняння:

```
> F(x1)
[1] -1.186177e-06
> F(x2)
[1] -4.261289e-07
> F(x3)
[1] -4.172325e-07
```

Отже, знайдені значення невідомого є наближеннями точного розв'язку з заданою точністю.

4.3. Метод хорд

При розв'язанні нелінійного рівняння методом хорд задаються відрізок $[a,b]$, на якому існує лише один розв'язок, і бажана точність $\varepsilon > 0$ розв'язку задачі.

Якщо на 0-й ітерації методу позначити $[a_0,b_0]=[a,b]$, то на k-й ($k = 0,1,2,\ldots$) ітерації методу буде поточний відрізок $[a_k,b_k]$. Потім через дві точки з координатами $(a_k,f(a_k))$ і $(b_k,f(b_k))$ проводиться відрізок прямої лінії (**хорда**) і визначається точка перетину цієї лінії з віссю абсцис (точка x_k). Якщо при цьому $f(a_k)f(x_k) < 0$, то права межа інтервала переноситься в точку x_k (тобто $b_{k+1} = x_k$, $a_{k+1} = a_k$). Якщо вказана умова не виконується, то в точку x_k переноситься ліва межа інтервала ($a_{k+1} = x_k$, $b_{k+1} = b_k$). Пошук розв'язку припиняється при досягненні заданої точності: $|f(x_k)| \le \varepsilon$.

Для визначення точки перетину хорди з віссю абсцис користуються формулою (спробувати отримати формулу самостійно)

$$x_{k+1} = a_k + \left| \frac{f(a_k)}{f(b_k) - f(a_k)} \right| (b_k - a_k). \qquad (4.2)$$

Графічна інтерпретація розв'язання нелінійних рівнянь методом хорд наведена на рис. 4.2.

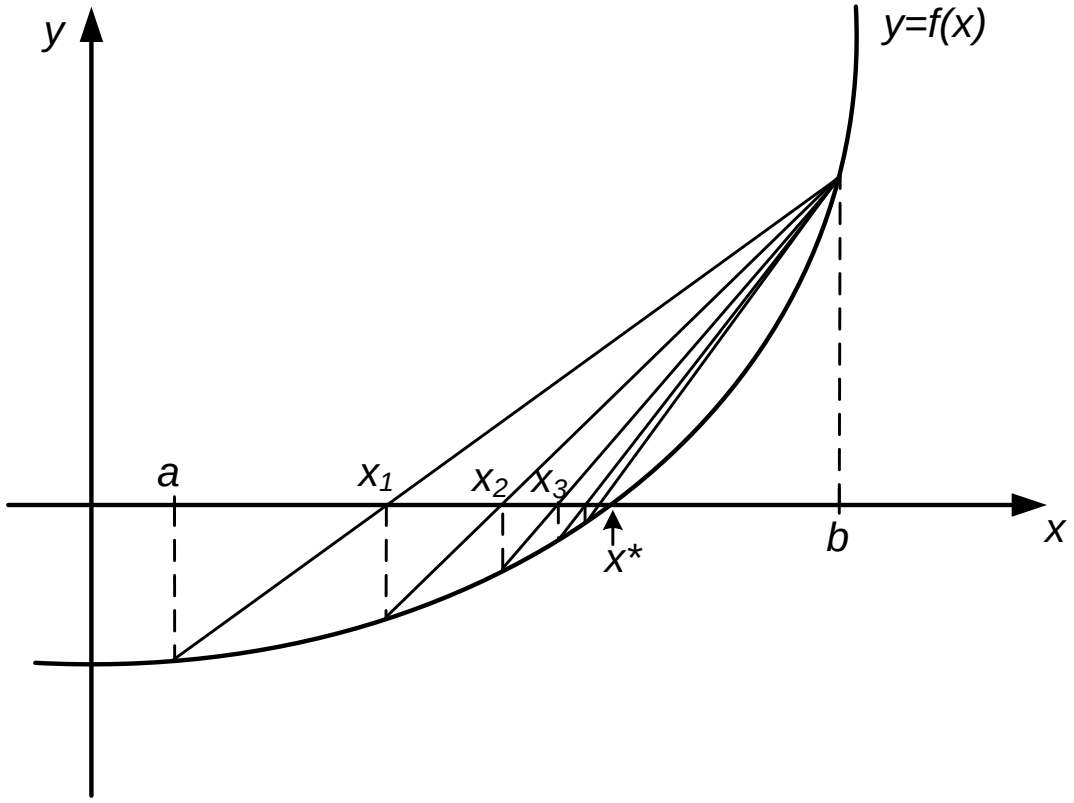


Рис. 4.2. **Графічна інтерпретація розв'язання нелінійних рівнянь методом хорд**

Слід зазначити, що метод хорд є методом *0*-го порядку, оскільки не використовує обчислення похідних функції $f(x)$.

Приклад 4.2. Розв'язати нелінійне рівняння, задане в прикладі 4.1 методом хорд.

Розв'язання в математичному пакеті R

Відправні дані – ті ж самі, що й для методу дихотомії.

Процедура розв'язання нелінійного рівняння методом хорд може бути записана так:

```
MHord = function(f, a, b, eps)
{
  x = a+abs(f(a)/(f(b)-f(a)))*(b-a)
  while (abs(f(x)) > eps)
  {
   x = a+abs(f(a)/(f(b)-f(a)))*(b-a)
   if (f(a)*f(x) < 0) b = x
   else a = x
  }
 return (x)
}
```

Результат розв'язання нелінійного рівняння з використанням записаної процедури:

```
> x1 = MHord(F, a1, b1, eps) # виклик функції користувача
> x1
[1] -2.414214
> x2 = MHord(F, a2, b2, eps) # виклик функції користувача
> x2
[1] 0.4142137
> x3 = MHord(F, a3, b3, eps) # виклик функції користувача
> x3
[1] 2
```

Таким чином, розв'язками заданого рівняння є знайдені значення невідомого: $x1 = -2.414$, $x2 = 0.414$, $x3 = 2$.

4.4. Метод Ньютона

Метод Ньютона ще називають **методом дотичних.** При розв'язанні нелінійного рівняння методом дотичних задаються відрізок $[a,b]$, на якому існує лише один розв'язок, початкове наближення розв'язку $x_0 \in [a,b]$ і бажана точність $\varepsilon > 0$ розв'язку рівняння.

На k-й ($k = 0,1,2,\ldots$) ітерації методу буде поточне наближення розв'язку $x_k \in R^1$. Наступне наближення розв'язку $x_{k+1} \in R^1$ визначається таким чином. У точці $(x_k, f(x_k))$ проводиться дотична до графіка $f(x)$ і визначається точка x_{k+1} як точка перетину дотичної з віссю абсцис. Пошук розв'язку припиняється при досягненні заданої точності $|f(x_k)| \le \varepsilon$.

Для визначення точки перетину $(k+1)$-ї дотичної з віссю абсцис користуються формулою (отримати формулу самостійно)

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}. \tag{4.3}$$

Графічна інтерпретація розв'язання нелінійних рівнянь методом Ньютона наведена на рис. 4.3.

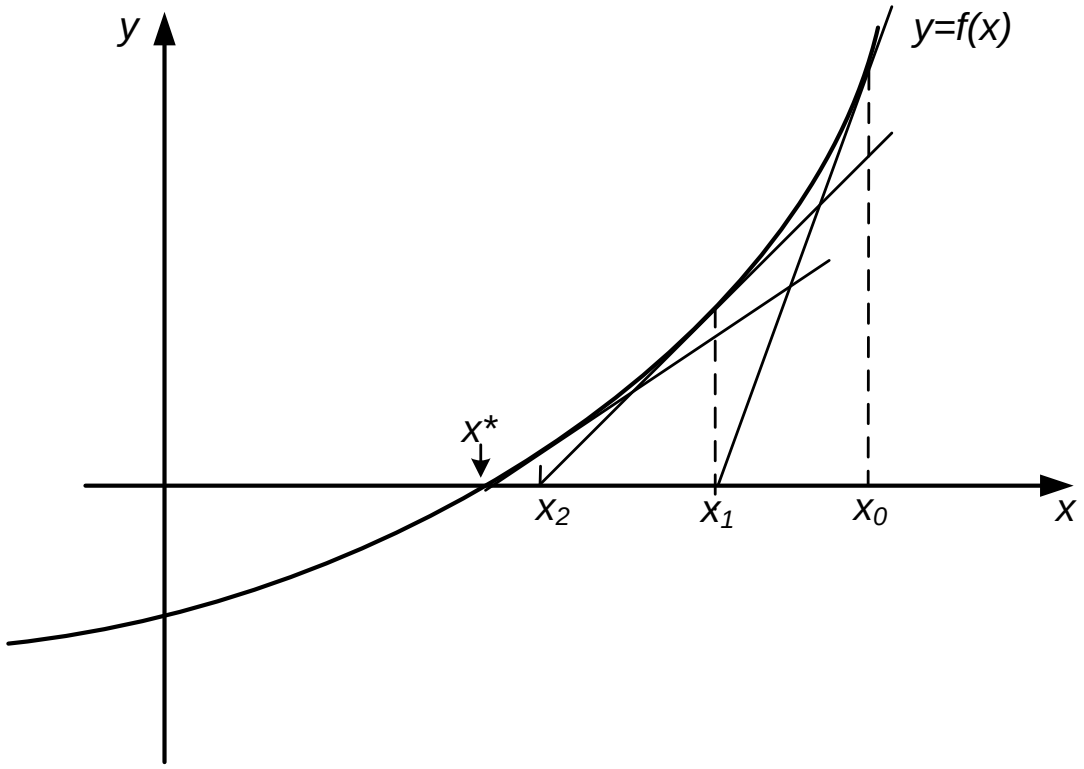


Рис. 4.3. **Графічна інтерпретація розв'язання нелінійних рівнянь методом Ньютона**

Умова збіжності методу дотичних [1] $f(x_0)f''(x_0) > 0$. При цьому швидкість збіжності буде квадратичною: $\left|x_{k+1} - x^*\right| \le M\left|x_k - x^*\right|^2$ для всіх $k > k_0$, де $k_0 > 0$, $M > 0$.

Слід зазначити, що метод дотичних є методом 1-го порядку, оскільки використовує обчислення першої похідної функції $f(x)$.

Варто зауважити, що ідея методу дотичних (як і методу хорд) полягає в наближенні нелінійної функції $f(x)$ лінійною функцією (дотичною, а в методі хорд – хордою) на кожній ітерації методу.

Приклад 4.3. Розв'язати нелінійне рівняння $x^3 - 5x + 2 = 0$ методом Ньютона з точністю $\varepsilon = 10^{-5}$.

Розв'язання в математичному пакеті R

Відправними даними задачі є функція $F(x)$ та точність розв'язку ε. Додатково задаються похідні, обчислені або задані аналітично ($DF(x)$):

```
# рівняння (функція)
F = function(x)
{
     return (x^3-5*x+2)
}

# перша похідна функції
DF = function(x)
{
     return (3*x^2-5)
}
eps = 10^(-6)
```

Початкові наближення розв'язків x_0 є додатковими відправними даними для методу Ньютона. До них ставляться такі вимоги:

1) x_0 мають належати відрізку $[a,b]$ і бути якомога ближчими до істинного розв'язку (див. приклад 4.1);

2) повинна виконуватися умова збіжності методу дотичних $f(x_0)f''(x_0) > 0$.

Задаються початкові наближення та перевіряються на збіжність:

```
# друга похідна функції
DF2 = function(x)
{
     return (6*x)
}
x01 = -2.5
x02 = 0.2
x03 = 2.1

zbig = function(f, df2, x)
{
  kx=f(x)*df2(x)
  return (kx)
}
```

```
> kx01 = zbig(F, DF2, x01)
> kx01
[1] 16.875
>
> kx02 = zbig(F, DF2, x02)
> kx02
[1] 1.2096
>
> kx03 = zbig(F, DF2, x03)
> kx03
[1] 9.5886
```

Усі значення *kx0* додатні, отже, умова збіжності виконується.

Процедура розв'язання нелінійного рівняння методом Ньютона може бути записана так:

```
MNewton = function(f, df, x0, eps)
{
   x = x0
   while (abs(f(x)) > eps)
   {
    x = x-f(x)/df(x)
   }
  return (x)
}
```

Результат розв'язання нелінійного рівняння з використанням записаної процедури:

```
> x1 = MNewton(F, DF, x01, eps)
> x1
[1] -2.414214

> x2 = MNewton(F, DF, x02, eps)
> x2
[1] 0.4142136
> x3 = MNewton(F, DF, x03, eps)
> x3
[1] 2
```

Таким чином, розв'язками заданого рівняння є знайдені значення невідомого: $x1 = -2.414$, $x2 = 0.414$, $x3 = 2$.

4.5. Метод простої ітерації

При розв'язанні нелінійного рівняння (4.1) методом ітерацій його потрібно записати у вигляді $x = \phi(x)$. Задаються початкове наближення

x_0 й точність $\varepsilon > 0$. Перше наближення розв'язку x_1 знаходиться з виразу $x_1 = \phi(x_0)$, друге – $x_2 = \phi(x_1)$ і т. д. У загальному випадку $(k+1)$-ше наближення обчислюється за формулою $x_{k+1} = \phi(x_k)$. Зазначена процедура повторюється доти, доки $|f(x_k)| = |x_k - \phi(x_k)| > \varepsilon$. Умова збіжності методу ітерацій $|\phi'(x)| \le q < 1$ для всіх $x \in [a,b]$. При цьому швидкість збіжності буде лінійною [1]: $\left|x_k - x^*\right| \le Mq^k$ для всіх $k > k_0$, де $k_0 > 0$, $M > 0$.

4.6. Висновки

1. У загальному випадку нелінійне рівняння з одним невідомим може мати безліч коренів. Тому для застосування чисельного методу необхідно вказати відрізок, на якому існує тільки один корінь.

2. Існуючі чисельні методи мають різну швидкість збіжності і кожен з них виявляється ефективним для свого класу нелінійних рівнянь з одним невідомим.

4.7. Контрольні запитання та завдання

1. Сформулюйте постановку задачі розв'язання рівняння з одним невідомим.

2. Які умови повинен задовольняти відрізок, на якому ведеться пошук розв'язку рівняння? Як його можна знайти?

3. У чому полягає метод дихотомії для розв'язання рівняння з одним невідомим? Яку він має швидкість збіжності?

4. У чому полягає метод хорд для розв'язання рівняння з одним невідомим? Яку він має швидкість збіжності?

5. У чому полягає метод Ньютона для розв'язання рівняння з одним невідомим? Яку він має швидкість збіжності? Яку ще назву має цей метод?

6. Знайдіть відрізки, на яких рівняння $x^3 - 4x + 2 = 0$ має корені методом відокремлення коренів.

7. Розв'яжіть вказане рівняння методами половинного поділу та простої ітерації.

5. Чисельні методи розв'язання систем нелінійних рівнянь

5.1. Постановка задачі

Розглядається система нелінійних рівнянь виду:

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) = 0, \\ f_2(x_1, x_2, \ldots, x_n) = 0, \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ f_n(x_1, x_2, \ldots, x_n) = 0. \end{cases}, \qquad (5.1)$$

де $f_i(x_1, x_2, \ldots, x_n)$, $i = \overline{1,n}$ – деякі нелінійні функції n змінних $x_1, x_2, \ldots, x_n$, $i = \overline{1,n}$.

Якщо ввести позначення $x = \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{pmatrix}$ – вектор-стовпець розмірності n з елементами x_i ($x \in R^n$), $F(x) = \begin{pmatrix} f_1(x) \\ \cdots \\ f_n(x) \end{pmatrix}$ – векторна функція розмірності n, елементами якої є функції $f_i(x) = f_i(x_1, x_2, \ldots, x_n)$, $i = \overline{1,n}$ ($F : R^n \to R^n$), то систему (5.1) можна записати у векторному вигляді:

$$F(x) = 0. \qquad (5.2)$$

Розв'язати систему (5.2) – означає знайти таке $x^* \in R^n$, для якого $F(x^*) \equiv 0$, тобто $f_i(x_1^*, x_2^*, \ldots, x_n^*) \equiv 0 \ \forall i = \overline{1,n}$.

Більшість математичних моделей різних процесів і явищ записуються в загальному випадку у вигляді (5.2), тому дана задача має величезне практичне значення.

Слід розглянути три основні методи розв'язання систем нелінійних рівнянь виду (5.2).

5.2. Метод Ньютона

Метод Ньютона будує ітераційну послідовність $\{x^{(k)}\}$, $(x^{(k)} \in R^n)$, $k = 0,1,2,\ldots$, наближень розв'язку x^* (початкове наближення x^0 задається) за такою ітераційною формулою:

$$x^{(k+1)} = x^{(k)} - [F'(x^{(k)})]^{-1} F(x^{(k)}), \tag{5.3}$$

де $F'(x) = \begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \ldots & \frac{\partial f_1(x)}{\partial x_n} \\ \ldots & \ldots & \ldots \\ \frac{\partial f_n(x)}{\partial x_1} & \ldots & \frac{\partial f_n(x)}{\partial x_n} \end{pmatrix}$ – матриця Якобі, тобто $F'(x^{(k)})$ – матриця розмірності $n \times n$ з елементами $F'(x^{(k)})_{ij} = \frac{\partial f_i(x^{(k)})}{\partial x_j}$.

Процес (5.3) триває доти, доки не виконається умова $\left\| F(x^{(k)}) \right\| \le \varepsilon$, де ε – задана точність розв'язку задачі (5.2).

Ідея метода Ньютона полягає в тому, що на k-й ітерації ($x^{(k)}$ – поточне наближення розв'язку) наступне наближення розв'язку $x^{(k+1)}$ знаходиться, як розв'язок системи лінійних рівнянь:

$$F_k(x) = 0, \tag{5.4}$$

де $F_k(x) \equiv F(x^{(k)}) + F'(x^{(k)})(x - x^{(k)})$ (перші два члени розкладання в ряд Тейлора функції $F(x)$ в околі точки $x^{(k)}$ [7]), тобто система (5.4) є лінеаризацією (лінійним наближенням) системи (5.2). Оскільки система (5.4) лінійна відносно x, то її розв'язок може бути знайдений аналітично:

$$F_k(x) \equiv F(x^{(k)}) + F'(x^{(k)})(x - x^{(k)}) = 0$$

або

$$F'(x^{(k)})(x - x^{(k)}) = -F(x^{(k)}).$$

Тому (через обернену матрицю)

$$(x - x^{(k)}) = -[F'(x^{(k)})]^{-1} F(x^{(k)}),$$

звідки і отримуємо розв'язок системи (5.4)

$$x = x^{(k)} - [F'(x^{(k)})]^{-1} F(x^{(k)}).$$

Якщо послідовність $\{x^{(k)}\}$, $k = 0,1,2,\ldots$, побудована згідно з (5.3), збігається, то за достатньо загальних умов [1; 3; 5; 6; 19] швидкість її збіжності буде квадратичною, тобто $\left\|x^{(k+1)} - x^*\right\| \le M\left\|x^{(k)} - x^*\right\|^2$ починаючи з деякого k, де M – деяка додатна константа.

Основними недоліками метода Ньютона є:

- збіжність тільки для достатньо близьких до розв'язку початкових наближень $x^{(0)}$;
- висока трудомісткість методу, оскільки на кожній ітерації необхідно обчислювати матриці $F'(x^{(k)})$ і $[F'(x^{(k)})]^{-1}$.

Основною перевагою методу Ньютона є висока швидкість збіжності.

Слід зазначити, що в методі Ньютона формулу (5.3) можна записати у вигляді $x^{(k+1)} = x^{(k)} + h^{(k)}$, де $h^{(k)} = -[F'(x^{(k)})]^{-1} F(x^{(k)})$. Проте при практичній реалізації методу вектор h^k ефективніше обчислювати як розв'язок системи лінійних рівнянь виду $F'(x^{(k)}) \times h = -F(x^{(k)})$.

Приклад 5.1. Розв'язати методом Ньютона систему нелінійних рівнянь

$$\begin{cases} 2x_1^2 + x_2^2 - 1 = 0, \\ x_1^3 + 6x_1^2 x_2 - 1 = 0 \end{cases}$$

з точністю $\varepsilon = 10^{-5}$ (початкове наближення $x_0 = \begin{pmatrix} 0{,}65 \\ 0{,}35 \end{pmatrix}$).

Розв'язання в математичному пакеті R

Відправними даними задачі є вектор функцій системи $F(x)$, матриця частинних похідних функцій системи $FFx(x)$, початкове наближення розв'язку $x0$, кількість невідомих n, точність розв'язку ε та максимальна кількість ітерацій $k\,max$:

```
# Вводимо векторну функцію F
Fx = function(x)
{
  f = c(1:2)
  f[1] = 2*(x[1])^2+(x[2])^2-1
  f[2] = (x[1])^3+(6*(x[1])^2)*x[2]-1
  return(f)
}
# Вводимо матрицю частинних похідних
FFx = function (x)
{
  ff = matrix(nrow=2, ncol=2)
  ff[1,1] = 4*x[1]
  ff[1,2] = 2*x[2]
  ff[2,1] = 3*(x[1]^2)+12*x[1]*x[2]
  ff[2,2] = 6*(x[1]^2)
  return(ff)
}
# Вводимо вектор початкового наближення розв'язку
x0 = c(0.65, 0.35)
N = 2 # Кількість невідомих
kmax = 100     # максимальна кількість ітерацій
eps = 0.00001  # точність розв'язку
```

Процедуру розв'язання системи нелінійних рівнянь методом Ньютона можна записати так:

```
Newton = function(fx, ffx, x, n, eps, kmax)
{
     y = fx(x)
     k = 0
     while ((sqrt(t(y)%*%y)) > eps)
     {
          yy = ffx(x)
          x = x - solve(yy)%*%y
          y = fx(x)
               if (k == kmax)
                    break
          k = k+1
     }
     x[n+1] = k
     return(x)
}
```

Результат розв'язання системи нелінійних рівнянь із використанням записаної процедури:

```
> X = Newton(Fx, FFx, x0, N, eps, kmax)
> X
[1] 0.6865944 0.2391155 4.0000000
```

Таким чином, розв'язком заданої системи нелінійних рівнянь є знайдені значення невідомих: $x_1 = 0{,}687$, $x_2 = 0{,}239$. Елемент вектора $x_3 = 4$ показує кількість виконаних ітерацій.

Для перевірки отриманих результатів слід підставити знайдені значення невідомих у систему:

```
> Fx(X)
[1]  1.176896e-10 -2.472814e-10
```

5.3. Метод простої ітерації

При розв'язуванні системи нелінійних рівнянь (5.5) методом простої ітерації її потрібно спочатку записати у вигляді $x = G(x)$, де $G(x)$ – векторна функція розмірності n від $x \in R^n$. Потім задаються початкове наближення $x^{(0)}$ і точність $\varepsilon > 0$. Метод будує ітераційну послідовність $\{x^{(k)}\}$, $k = 0,1,2,...$, наближень розв'язку x^* за такою ітераційною формулою

$$x^{(k+1)} = G(x^{(k)}). \qquad (5.5)$$

Процес (5.5) триває доти, доки не виконається умова $\left\| x^{(k)} - G(x^{(k)}) \right\| \le \varepsilon$. Цей критерій закінчення обчислень виходить з рівності $F(x) = x - G(x)$.

Метод простої ітерації (5.5) збігається, якщо $\|G'(x)\| \le q < 1$ для всіх x, що належать деякому околу $V(x^*)$ розв'язку x^* і $x^0 \in V(x^*)$. При цьому швидкість збіжності буде лінійною, тобто $\left\| x^{(k+1)} - x^* \right\| \le q \left\| x^{(k)} - x^* \right\|$, починаючи з деякого k [1; 3; 5; 6; 19].

Основними недоліками методу простої ітерації є:

- складність переходу від запису системи $F(x) = 0$ до виду $x = G(x)$, зважаючи на нелінійність функцій $f_i(x_1, x_2, ..., x_n)$, $i = \overline{1,n}$;
- збіжність тільки для достатньо близьких до розв'язку початкових наближень $x^{(0)}$;
- невисока швидкість збіжності.

Основною перевагою методу простої ітерації є невисока трудомісткість методу.

Приклад 5.2. Розв'язати методом простої ітерації систему нелінійних рівнянь з прикладу 5.1.

Розв'язання в математичному пакеті R

Відправними даними задачі є векторна функція системи $G(x)$, початкове наближення розв'язку $x0$, кількість невідомих n, точність розв'язку ε та максимальна кількість ітерацій $kmax$:

```
N = 2 # Кількість невідомих
# Вводимо векторну функцію G
Gx = function(x)
{
  g = c(1:2)
  g[1] = sqrt((1-x[2]^2)/2)
  g[2] = (1-(x[1])^3)/(6*(x[1])^2)
  return(g)
}

# Вводимо вектор початкового наближення розв'язку
x0 = c(0.65, 0.35)
kmax = 100     # максимальна кількість ітерацій
eps = 0.00001  # точність розв'язку
```

Процедуру розв'язання системи нелінійних рівнянь методом простої ітерації можна записати так:

```
MetIter=function(G,x,n,eps,kmax)
{
     k = 0
     repeat
     {
          xk = G(x)
          y = xk - G(xk)
          if (((sqrt(t(y)%*%y))<= eps)|(k == kmax))
          {
               break
          }
          x = xk
          k = k + 1
     }
     x[n+1] = k
     return(x)
}
```

Результат розв'язання системи нелінійних рівнянь із використанням записаної процедури:

```
> X = MetIter(Gx,x0,N,eps,kmax)
> X
[1]  0.6865834  0.2391370 11.0000000
```

Таким чином, розв'язком заданої системи нелінійних рівнянь є знайдені значення невідомих: $x_1 = 0{,}687$, $x_2 = 0{,}239$. Елемент вектора $x_3 = 11$ показує кількість виконаних ітерацій.

Для перевірки отриманих результатів слід підставити знайдені значення невідомих у систему:

```
> Fx(X)
[1]  1.176896e-10 -2.472814e-10
```

У результаті розв'язання системи нелінійних рівнянь двома методами з однаковою точністю отримані однакові результати. Різниця тільки в тому, що в методі Ньютона розв'язок отримано за 4 ітерації, а в методі простої ітерації – за 11 ітерацій.

5.4. Метод найменших квадратів

Пошук розв'язку задачі (5.2), по суті, еквівалентний задачі знаходження точки мінімуму функції $\Phi(x)$ виду

$$\Phi(x) \equiv \|F(x)\|^2 = \sum_{i=1}^{n} f_i^2(x). \qquad (5.6)$$

Так, якщо $x^* \in R^n$ – розв'язок задачі (5.2), то $F(x^*) = 0$, тобто $\Phi(x^*) = 0$. Але функція $\Phi(x)$ – невід'ємна, тобто $\Phi(x) \geq 0$ для всіх $x \in R^n$, тому в точці x^* вона досягає мінімального значення.

І навпаки, якщо функція $\Phi(x)$ в точці x^* досягає мінімального значення і при цьому $\Phi(x^*) = 0$, то $\|F(x^*)\|^2 = 0$, а значить і $F(x^*) = 0$, тобто x^* – розв'язок задачі (5.2).

Задачу знаходження точки мінімуму функції $\Phi(x)$ виду (5.6) називають **задачею найменших квадратів**. Слід зазначити, що задачу найменших квадратів (5.6) можна розглядати і як самостійну, тобто без прив'язки до системи (5.2). В цьому випадку кількість функцій $f_i(x)$ може бути більшим, ніж кількість змінних x_j.

Таким чином, розв'язання системи нелінійних рівнянь (5.2) зводиться до оптимізаційної задачі [7], розв'язок якої з практичної точки зору виявляється, зрештою, простішим.

Приклад 5.3. Розв'язати систему нелінійних рівнянь з прикладу 5.1 методом найменших квадратів.

Розв'язання в математичному пакеті R

Відправними даними задачі є вектор функцій системи *F(x)*, початкове наближення розв'язку *x0* та кількість невідомих *n*:

```
n = 2 # Кількість невідомих
# Вводимо векторну функцію F
Fx = function(x)
{
  f = c(1:2)
  f[1] = 2*(x[1])^2+(x[2])^2-1
  f[2] = (x[1])^3+(6*(x[1])^2)*x[2]-1
  return(f)
}
```

Розв'язання системи нелінійних рівнянь методом найменших квадратів із використанням вбудованої процедури *optim*.

```
# Визначаємо цільову функцію метода МНК
FunMNK = function(x)
{
     z = Fx(x)
     S = 0
     for(i in 1:n)
          S = S + z[i]*z[i]
     return( S )
}
> # Вводимо вектор початкового наближення розв'язку
> x0 = c(0.65, 0.35)
>
> res = optim(fn=FunMNK, par=x0)
> x1 = res$par
> x1
[1] 0.6866008 0.2390994
```

Для перевірки отриманих результатів слід підставити знайдені значення невідомих у систему:

```
> Fx(x1)
[1]  9.701967e-06 -2.408662e-05
```

5.5. Висновки

1. Більшість математичних моделей різних процесів і явищ записуються в загальному випадку у вигляді (5.1).

2. Основними методами розв'язання систем нелінійних рівнянь є метод Ньютона, метод простої ітерації та метод найменших квадратів.

5.6. Контрольні запитання та завдання

1. Сформулюйте постановку задачі розв'язання системи нелінійних рівнянь.

2. У чому полягає метод Ньютона для розв'язання систем нелінійних рівнянь? Яка ідея методу? Вкажіть основні характеристики цього методу.

3. У чому полягає метод простої ітерації для розв'язання систем нелінійних рівнянь? Вкажіть основні характеристики цього методу.

4. У чому полягає метод найменших квадратів для розв'язання систем нелінійних рівнянь?

5. Розв'яжіть систему нелінійних рівнянь

$$\begin{cases} x_1 + x_1^2 - 2x_2x_3 - 0.1 = 0, \\ x_2 - x_2^2 + 3x_1x_3 + 0.2 = 0, \\ x_3 + x_3^2 + 2x_1x_2 - 0.3 = 0 \end{cases}$$

двома методами: Ньютона та простої ітерації. Прийняти початкове наближення $x^0 = (0; 0; 0)$.

Порівняйте трудомісткість і швидкість збіжності методів.

6. Чисельні методи обчислення власних значень і власних векторів матриці

6.1. Постановка задачі

Визначення. **Власним значенням** (власним числом) дійсної матриці A розмірності $n \times n$ ($A \in R^{n \times n}$) називається таке число λ (дійсне

або комплексне), для якого знайдеться дійсний ненульовий вектор x розмірності n ($x \in R^n$), що виконується умова

$$A\,x = \lambda\,x. \qquad (6.1)$$

При цьому вектор x називається **власним вектором** матриці A, що відповідає власному значенню λ.

Матриця розмірності $n \times n$ має n власних чисел, які можуть бути кратними.

Задача обчислення власних значень і власних векторів полягає в знаходженні всіх власних значень і відповідних власних векторів квадратної матриці.

Такі задачі нерідко виникають при розв'язанні інженерних задач. Наприклад, при динамічному аналізі механічних систем власні значення відповідають власним частотним коливанням, а власні вектори характеризують моди (просторовий напрям і амплітуди) цих коливань. При розрахунку конструкцій власні значення дозволяють визначити критичні навантаження, перевищення яких призводить до втрати стійкості (руйнування).

Властивості власних значень матриці. Нехай λ_i, $i = \overline{1,n}$, – власні значення матриці A, тоді [3]:

1. Вони задовольняють **характеристичне рівняння** $det(A - \lambda E) = 0$.

2. $det(A) = \prod_{i=1}^{n} \lambda_i$.

3. Якщо матриця A є виродженою (особливою або сингулярною), тобто $det(A) = 0$, то хоча б одне її власне значення дорівнює нулю.

4. Власні значення оберненої матриці A^{-1} дорівнюють $\frac{1}{\lambda_i}$, $i = \overline{1,n}$.

5. Власні значення матриці A^k ($k = 0,1,2,...$) дорівнюють λ_i^k, $i = \overline{1,n}$.

6. Усі n власних значень дійсної симетричної матриці розміру $n \times n$ дійсні.

Добуток власного вектора матриці на скаляр є власним вектором тієї ж матриці. Як правило, власні вектори нормалізують, поділивши їх на норму вектора або на найбільший елемент.

Чисельні методи обчислення власних значень і власних векторів матриці поділяються на дві групи: **ітераційні** та **методи перетворення подібності.**

6.2. Ітераційні методи обчислення власних значень і власних векторів

Ітераційні методи засновані на багатократному використанні ітераційного алгоритму, що наближає власний вектор, який одержується в кожному циклі, до точного розв'язку.

Ці методи застосовують тоді, коли треба знайти тільки окреме власне значення (наприклад, максимальне за модулем).

Метод ітерацій. Метод ітерацій застосовується для пошуку найбільшого за модулем власного значення квадратної матриці.

Нехай $A \in R^{n \times n}$. Тоді алгоритм методу ітерацій такий [10]:

1) задається початковий вектор $z^0 \in R^n$ (початкове наближення для власного вектора матриці A, що відповідає найбільшому за модулем власному значенню λ_{max} матриці A);

2) k-та ітерація ($k = 0,1,2,...$). На ній буде поточний вектор $z^k \in R^n$. Далі проводиться нормування вектора z^k, а саме обчислюється число λ_{max}^k таке, що $\left|\lambda_{max}^k\right| = \max\limits_{i=\overline{1,n}} \left|z_i^k\right|$, і вектор $y^k = \dfrac{z^k}{\lambda_{max}^k}$, і після чого визначається наступний вектор $z^{k+1} = A y^k$;

3) обчислення припиняються, коли виконується умова $\dfrac{\left|\lambda_{max}^{k+1} - \lambda_{max}^k\right|}{\left|\lambda_{max}^{k+1}\right|} \le \varepsilon$, де ε – задана відносна точність обчислень.

Умова збіжності методу ітерацій. Можна позначити власні значення λ_i, $i = \overline{1,n}$, матриці A так, щоб виконувався порядок

$$\left|\lambda_1\right| > \left|\lambda_2\right| \ge \left|\lambda_3\right| \ge \ldots \ge \left|\lambda_n\right|,$$

тобто $\lambda_{max} = \lambda_1$.

Метод ітерацій збігається, якщо $|\lambda_1| > |\lambda_2|$. Тоді швидкість збіжності лінійна з коефіцієнтом $q = \dfrac{|\lambda_2|}{|\lambda_1|} < 1$ [21].

Якщо необхідно знайти $\lambda_{min} = \lambda_n$, то треба знайти λ^*_{max} для оберненої матриці A^{-1}. Тоді $\lambda_{min} = \dfrac{1}{\lambda^*_{max}}$.

Визначивши максимальне або мінімальне за модулем власне значення, можна знайти наступні за ним за величиною, замінивши відправну матрицю матрицею, яка містить лише решту власних значень.

Так, якщо створити нову матрицю $A^* = A - \lambda_1 X_1 X_1^T$, де X_1 – власний вектор матриці A, що відповідає власному значенню λ_1, то вона буде мати власне значення $\lambda_1 = 0$, а всі решта її власних значень будуть збігатися з власними значеннями відправної матриці A [23]. У результаті знаходиться λ_2 і далі, утворюючи A^{**} і т. д., аналогічно визначаються всі власні значення і вектори.

6.3. Методи перетворення подібності для обчислення власних значень і власних векторів

Методи перетворення подібності засновані на використанні властивості подібних матриць, що мають однакові власні значення.

Визначення. Матриця A називається **ортогональною**, якщо $A^{-1} = A^T$, де T – знак транспонування.

Визначення. Матриці A і B називаються **подібними**, якщо існує така несингулярна матриця P, що справедливе співвідношення

$$B = P^{-1} A P.$$

Властивість подібних матриць. Якщо дві матриці подібні, то їх власні значення збігаються.

З подібності A і B виходить, що $B = P^{-1} A P$. Оскільки $A x = \lambda x \Rightarrow$ $\Rightarrow P^{-1} A x = \lambda P^{-1} x$. Якщо прийняти $x = P y$, тоді $P^{-1} A P y = \lambda y$, тобто A і B не тільки мають однакові власні значення λ, але і їх власні вектори зв'язані співвідношенням $x = P y$.

Методи перетворення подібності здійснюють перетворення матриці A таким чином, щоб у отриманої подібної матриці власні значення обчислювались простіше.

Слід зазначити, що з умови (6.1) витікає, що $(A-\lambda E)x=0$, тому для того, щоб ця система мала ненульовий розв'язок власні значення λ повинні задовольняти рівнянню

$$det(A-\lambda E)=0. \tag{6.2}$$

Варто розглянути декілька прикладів матриць, власні значення яких легко знайти.

1. Нехай A **трикутна матриця**, тобто $A=\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{pmatrix}$. Тоді

$A-\lambda E=\begin{pmatrix} a_{11}-\lambda & a_{12} & \cdots & a_{1n} \\ 0 & a_{22}-\lambda & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn}-\lambda \end{pmatrix}$ і $det(A-\lambda E)=(a_{11}-\lambda)(a_{22}-\lambda)\ldots(a_{nn}-\lambda)=0$,

Звідки витікає, що $\lambda_i=a_{ii}, i=\overline{1,n}$.

2. Нехай A **блочно-трикутна матриця**, тобто $A=\begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ 0 & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & A_{nn} \end{pmatrix}$,

де блоки A_{ii} мають розмір 1x1 або 2x2. Тоді власні значення матриці A дорівнюють власним значенням блоків A_{ii}.

Треба знайти власні значення блока A_{ii}, який має розмір 2x2, тобто $A_{ii}=\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$.

Тоді $det\begin{pmatrix} a_{11}-\lambda & a_{12} \\ a_{21} & a_{22}-\lambda \end{pmatrix}=(a_{11}-\lambda)(a_{22}-\lambda)-a_{12}a_{21}=$

$=\lambda^2-\lambda(a_{11}+a_{22})+(a_{11}a_{22}-a_{12}a_{21})=0$, звідки витікає, що

$$\lambda_{1,2}=\frac{Tr(A)\pm\sqrt{(Tr(A))^2-4\,det(A)}}{2},$$

де $Tr(A)=a_{11}+a_{22}, det(A)=(a_{11}a_{22}-a_{12}a_{21})$.

Визначення: **QR-розкладанням (або QR-факторизацією)** називається подання матриці A у вигляді $A = QR$, де Q – ортогональна, а R – верхня трикутна матриці.

QR-алгоритм пошуку власних значень і власних векторів матриці [1; 24].

Нехай $A_0 = A$.

Матриця A_0 подається у вигляді $A_0 = Q_1 R_1$ і обчислюється матриця $A_1 = R_1 Q_1 = Q_1^{-1} R_1 Q_1$. Тобто матриця A_1 подібна матриці A_0.

Знову матриця A_1 подається у вигляді $A_1 = Q_2 R_2$ і обчислюється матриця $A_2 = R_2 Q_2 = Q_2^{-1} R_2 Q_2$. Тобто матриця A_2 подібна матриці A_1, а значить і A_0. Так здійснюється і далі.

Таким чином, будується послідовність матриць $\{A_k\}, k = 0,1,2,\ldots$: $A_k = Q_{k+1} R_{k+1}$, $A_{k+1} = R_{k+1} Q_{k+1}$, котрі є подібними до відправної матриці A.

Справедливе таке твердження: якщо всі головні мінори матриці A невирождені, то послідовність матриць $\{A_k\}, k = 0,1,2,\ldots$, збігаються до матриці блочно-трикутного виду (матриці Хесенберга) [23], власні значення якої легко знайти.

QR-алгоритм є стійким до похибок обчислення, тому вважається найбільш ефективним для пошуку власних значень і власних векторів матриці.

Для QR-розкладу матриці застосовують метод обертання Гівенса або перетворення Хаусхольдера [23].

6.4. Висновки

1. Задача обчислення власних значень і власних векторів квадратної матриці нерідко виникає при розв'язанні інженерних задач.

2. Чисельні методи обчислення власних значень і власних векторів матриці поділяються на дві групи: ітераційні та методи перетворення подібності.

3. Ітераційні методи застосовують тоді, коли треба знайти тільки окреме власне значення (наприклад, максимальне за модулем).

4. Методи перетворення подібності засновані на використанні властивості подібних матриць, що мають однакові власні значення.

6.5. Контрольні запитання та завдання

1. Дайте визначення власного значення та власного вектора квадратної матриці.

2. Сформулюйте постановку задачі обчислення власних значень і власних векторів квадратної матриці.

3. Для пошуку яких власних значень призначені ітераційні методи?

4. На якому принципі засновані чисельні методи перетворення подібності для обчислення власних значень і власних векторів матриці?

5. Як можна знайти мінімальне за модулем власне значення матриці?

6. Знайдіть чисельно максимальне та мінімальне за модулем власні значення матриці з системи лінійних алгебраїчних рівнянь

$$\begin{cases} 16x_1 - 8x_2 - 4x_3 = -8, \\ -8x_1 + 13x_2 - 4x_3 - 3x_4 = 7. \\ -4x_1 - 4x_2 + 9x_3 = 6, \\ -3x_2 + 3x_4 = -3. \end{cases}$$

7. Чисельні методи наближення функцій. Апроксимація, інтерполяція та екстраполяція

Задача наближення функцій виникає при розв'язанні багатьох практичних і теоретичних задач, а інколи і як самостійна. Так, наближення функцій є важливим допоміжним апаратом при розв'язанні інших задач чисельного аналізу: чисельного інтегрування і диференціювання, розв'язання диференціальних рівнянь, розв'язання систем нелінійних рівнянь, задач оптимізації та ін.

7.1. Постановка задачі. Поняття апроксимації та інтерполяції

Слід розглянути декілька постановок задачі наближення функцій.

Постановка 1. Проста задача, що приводить до наближення функцій, полягає в наступному. Нехай є деяка функція $f(x)$, $f : R^1 \to R^1$, про яку відомо, що в n точках $x_1, x_2, \ldots, x_n$ вона приймає, відповідно, значення $y_1, y_2, \ldots, y_n$, тобто $y_i = f(x_i), i = \overline{1,n}$. Потрібно відновити її

значення при інших значеннях $x \in (x_1, x_n)$. Функція $y = f(x)$ може бути як невідомою, тобто її значення тільки вимірюються (так званий "чорний ящик"), а може бути і просто дуже складною для обчислень. Наприклад, вона може використовуватися в яких-небудь фізико-технічних або суто математичних розрахунках, де її доводиться багато разів обчислювати. Слід зазначити, що якщо функція $y = f(x)$ невідома, то набір точок (x_i, y_i), $i = \overline{1,n}$, називають **табличним заданням функції**, оскільки ці значення подають у вигляді таблиці.

У цих випадках функцію $f(x)$ стараються замінити на простішу функцію $g(x;a)$, близьку до $f(x)$, тобто:

$$f(x) \approx g(x;a), \tag{7.1}$$

де $a \in R^k$, $a = (a_1, a_2, \ldots, a_k)^T$ – деякі параметри функції g, вид якої відомий. Процес наближення однієї функції іншою ще називають **апроксимацією**, а функцію g при цьому називають **апроксимуючою** для f.

Варто зазначити, що параметри $a \in R^k$ є інструментом для підгонки (наближення) функції g до функції f.

Якщо параметри $a_1, a_2, \ldots, a_k$ визначаються з умови збігу значень функції $f(x)$ і апроксимуючої функції в точках $x_1, x_2, \ldots, x_n$, тобто:

$$g(x_i;a) = f(x_i), \ \forall i = \overline{1,n},$$

то такий спосіб наближення називають **інтерполяцією** або **інтерполюванням**.

Слід зазначити, що точки x_i, $i = \overline{1,n}$ називають **вузлами інтерполяції** або ще **полюсами**, тобто "інтерполяція" – це відновлення функції між полюсами.

Постановка 2. У задачах планування експериментів виникає така проблема. Відомо вид гарного наближення функції, наприклад функція $f(x)$ добре наближається поліномом 2-го степеня. В той же час вимірювані значення функції y_i мають великі помилки. Потрібно отримати найкраще в певному значенні наближення при мінімальній кількості вимірювань. Така задача виникає при плануванні експериментів в біології, хімії, фізиці, географії, військовій справі та ін.

Постановка 3. Задача наближення функцій розв'язується і при складанні стандартних процедур обчислення елементарних і спеціальних функцій, наприклад $cos(x)$, $sin(x)$ й ін. Так:

$$e^x \approx \sum_{k=0}^{n} \frac{x^k}{k!},$$

$$sin(x) \approx x + \sum_{k=1}^{n} (-1)^k \frac{x^{2k-1}}{(2k-1)!},$$

де n підбирається так, щоб забезпечити достатню точність обчислень.

7.2. Метод найменших квадратів для апроксимації функцій

Найбільш відомим і ефективним із методів розв'язання задачі апроксимації функцій (7.1) є **метод найменших квадратів (МНК).** МНК призначений безпосередньо для знаходження параметрів a апроксимуючої функції $g(x;a)$, вид якої вже обраний. Суть метода полягає в такому.

Вводиться функція від a виду:

$$\Phi(a) \equiv \sum_{i=1}^{n} [(y_i - g(x_i;a)]^2, \qquad (7.2)$$

яку можна розглядати як міру відхилення функції $g(x;a)$ (по аргументу x) від функції $f(x)$ по сукупності точок $x_1, x_2, \ldots, x_n$. Тоді параметри a функції $g(x;a)$ можна визначити з умови найменшого відхилення $g(x;a)$ від $f(x)$, тобто параметри a знаходяться як точка, в якій функція $\Phi(a)$ досягає по $a \in R^k$ мінімального значення (точка мінімуму). Нехай a^* – шукані параметри. При цьому величини

$$r_i \equiv y_i - g(x_i;a^*),\ i = \overline{1,n}$$

називаються **залишковими відхиленнями** (або **залишками)**і використовуються для аналізу отриманого розв'язку.

Вид графіка залишків дозволяє оцінити, наскільки правильно спочатку був вибраний вид апроксимуючої функції $g(x;a)$. Якщо в графіку залишків спостерігається деяка функціональна закономірність

(наприклад, парабола), то це означає, що цю закономірність потрібно перенести в апроксимуючу функцію. Нормальною є ситуація, коли графік залишків має вид ламаної синусоїди (ще кажуть, має випадковий характер).

Слід зазначити, що метод найменших квадратів зазвичай застосовують для розв'язання задачі наближення функцій у постановці 2 (див. розділ 7.1).

Приклад 7.1. Використовуючи метод найменших квадратів, підібрати найкращу апроксимацію поліномом (визначити його степінь) для функції, заданої таблицею:

x	– 1,5	– 1	– 0,5	0	0,5	1	1,5	2	2,5
y	9,8	6,4	5,2	1,7	2,9	5,6	7,2	16,5	27,5

Розв'язання в математичному пакеті R

Відправними даними задачі є вектор значень аргумента x, вектор значень функції y та кількість точок спостереження n:

```
x = c(-1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5)
y = c(9.8, 6.4, 5.2, 1.7, 2.9, 5.6, 7.2, 16.5, 27.5)
n = 9
```

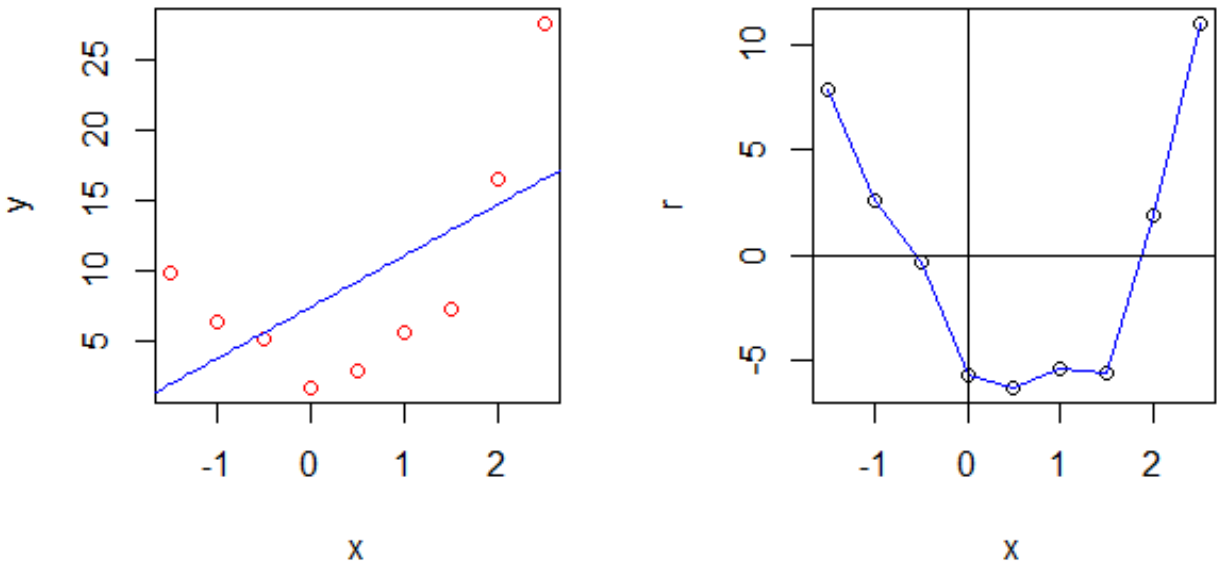
Апроксимація функції методом найменших квадратів із використанням вбудованої процедури *optim*.

1. Апроксимація лінійною функцією (поліномом 1-ї степені):

```
# Задаємо вид апроксимуючої функції
FunModel = function(x,a)
{
    return( a[1]+a[2]*x )
}
# Визначаємо цільову функцію метода МНК
FunMNK = function(a)
{
    S = 0
    for(i in 1:n)
        S = S + (y[i]-FunModel(x[i],a))^2
    return( S )
}
```

```
> # Задаємо початкове значення
> # для параметрів a апроксимуючої функції
> a0 = c(0,0)
> res = optim(fn=FunMNK, par=a0)
> a1 = res$par
> a1
[1] 7.383490 3.633125
> FunMNK(a1)
[1] 327.2633
> # Обчислюємо значення апроксимуючої функції в точках x
> ym = FunModel(x,a1)
> # Обчислюємо залишки
> r = y - ym
> a = -2; b = 3; n = 100
> # Обчислюємо n точок на відрізку [a,b]
> h = (b-a)/(n-1)
> x1 = c(1:n)
> for(i in 1:n){ x1[i] = a + h*(i-1) }
> # Обчислюємо значення апроксимуючої функції в точках x
> y1m = FunModel(x1,a1)
> par(mfrow=c(1, 2))            # задаємо дві графічні підобласті
> plot(x, y, type="p", col="red")# рисуємо перший графік у вигляді точок
> lines(x1, y1m, col="blue")  # додаємо в 1-у графічну підобласть ще лінію
> plot(x, r, type="p")          # рисуємо перший графік у вигляді точок
> lines(x, r, col="blue")     # додаємо в 2-у графічну підобласть ще лінію
> abline(h=0)
> abline(v=0)
```

Графіки функцій (заданої та апроксимуючої) і графік залишків:



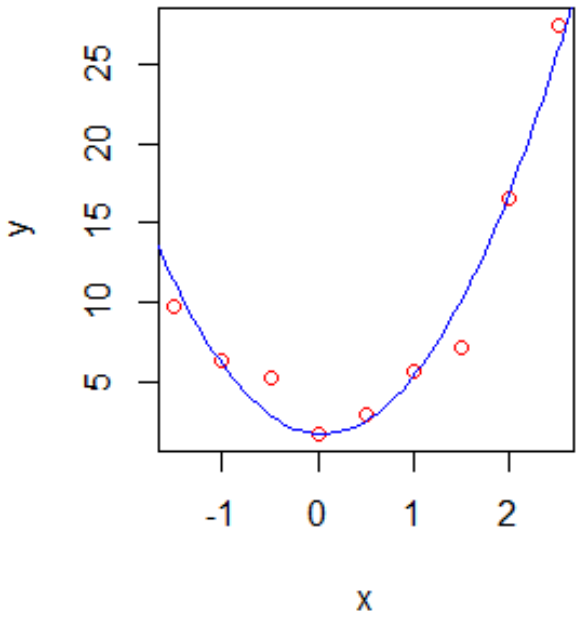
З графіків функцій видно, що апроксимуюча функція виду $y = a_0 + a_1 x$ погано наближається до функції, заданої таблицею. В

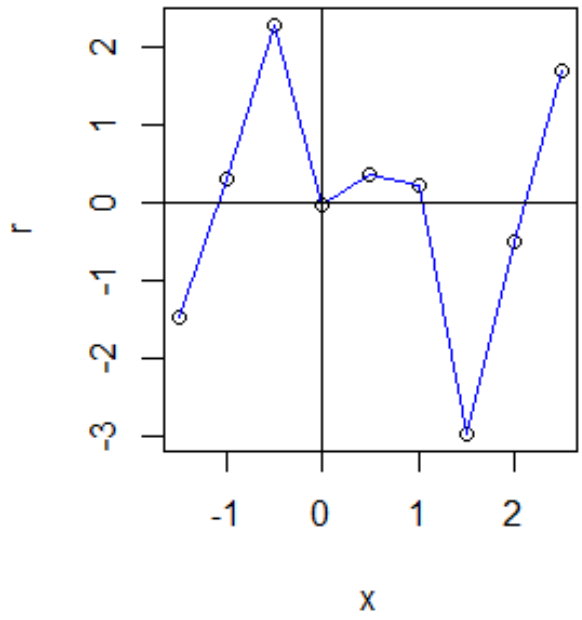
графіку залишків спостерігається закономірність у вигляді параболи, яку необхідно перенести в апроксимуючу функцію.

2. Апроксимація квадратичною функцією (поліномом 2-ї степені):

```
# Задаємо вид апроксимуючої функції
FunModel = function(x,a)
{
     return( a[1]+a[2]*x+a[3]*x^2 )
}
# Визначаємо цільову функцію метода МНК
FunMNK = function(a)
{
     S = 0
     for(i in 1:n)
          S = S + (y[i]-FunModel(x[i],a))^2
     return( S )
}
> # Задаємо початкове значення
> # для параметрів а апроксимуючої функції
> a0 = c(0,0,0)
> res = optim(fn=FunMNK, par=a0)
> a1 = res$par
> a1
[1]  1.7213365 -0.3640413  3.9971283
> FunMNK(a1)
[1] 19.66321
```

Графіки функцій (заданої та апроксимуючої) і графік залишків:



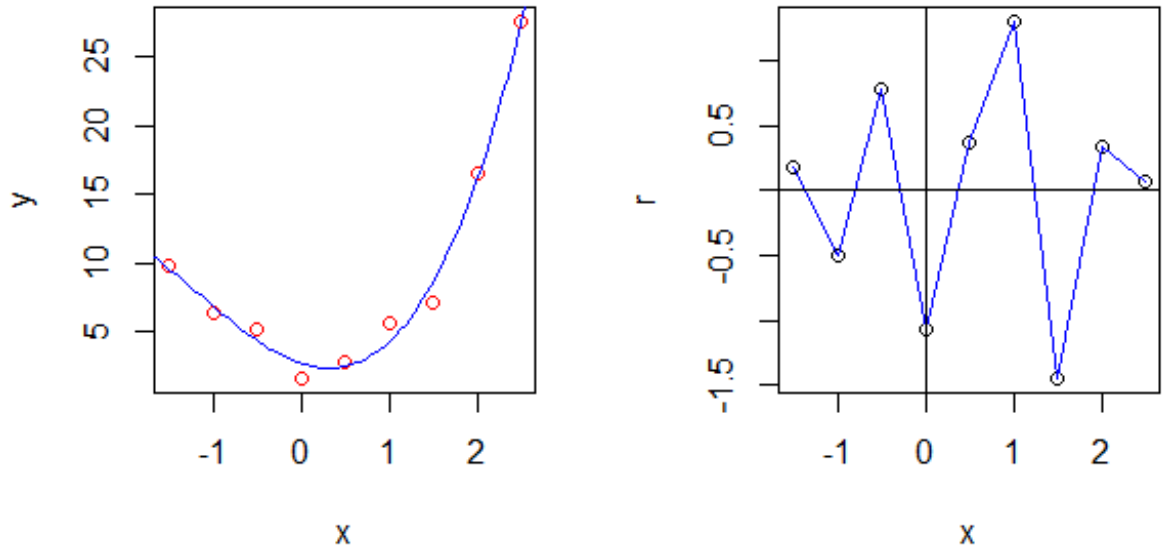


З графіків функцій видно, що апроксимуюча функція виду $y = a_0 + a_1 x + a_2 x^2$ також не дуже гарно наближається до функції, заданої таблицею. В графіку залишків спостерігається закономірність у вигляді кубічної функції, яку необхідно перенести в апроксимуючу функцію.

3. Апроксимація кубічною функцією (поліномом 3-ї степені):

```
# Задаємо вид апроксимуючої функції
FunModel = function(x,a)
{
     return( a[1]+a[2]*x+a[3]*x^2+a[4]*x^3 )
}
# Визначаємо цільову функцію метода МНК
FunMNK = function(a)
{
     S = 0
     for(i in 1:n)
          S = S + (y[i]-FunModel(x[i],a))^2
     return( S )
}
> # Задаємо початкове значення
> # для параметрів а апроксимуючої функції
> a0 = c(0,0,0,0)
> res = optim(fn=FunMNK, par=a0)
> a1 = res$par
> a1
[1]  2.7745381 -2.0843292  2.8265234  0.7812671
> FunMNK(a1)
[1] 6.071312
```

Графіки функцій (заданої та апроксимуючої) і графік залишків:



Як видно з графіків і значень функцій *FunkMNK(a1)* для різних апроксимуючих функцій, кращою виявилась кубічна функція виду $y = a_0 + a_1 x + a_2 x^2 + a_3 x^3$.

У випадках, коли функція $g(x;a)$ є лінійною по параметрам a точку мінімуму функції $\Phi(a)$ можна знайти аналітично з необхідної умови мінімуму 1-го порядку $\Phi'(a) = 0$ [1; 7], тобто:

$$\frac{\partial \Phi}{\partial a_j} = 0, \ \forall j = \overline{1,k}. \tag{7.3}$$

Слід розглянути загальний випадок, коли функція $g(x;a)$ лінійна по параметрам, тобто має вигляд:

$$g(a;x) = \sum_{m=1}^{k} a_m \psi_m(x), \tag{7.4}$$

де $\psi_m(x)$, $m = \overline{1,k}$, – деякі відомі функції від x. Тоді з (7.2) – (7.4) отримуємо $\forall j = \overline{1,k}$

$$\frac{\partial \Phi}{\partial a_j} = -2 \sum_{i=1}^{n} [y_i - g(x_i;a)] \psi_j(x_i) = -2 \sum_{i=1}^{n} \left[y_i - \sum_{m=1}^{k} a_m \psi_m(x_i) \right] \psi_j(x_i) = 0.$$

Звідки для $\forall j = \overline{1,k}$

$$\sum_{m=1}^{k} a_m \left[\sum_{i=1}^{n} \psi_m(x_i) \psi_j(x_i) \right] = \sum_{i=1}^{n} y_i \psi_j(x_i). \tag{7.5}$$

Таким чином, параметри a можуть бути знайдені як розв'язок системи лінійних рівнянь (7.5), яку можна записати в матричному вигляді:

$$Ca = b, \tag{7.6}$$

де

$$c_{jm} = \sum_{i=1}^{n} \psi_m(x_i) \psi_j(x_i), \ j = \overline{1,k}, \ m = \overline{1,k};$$

$$b_j = \sum_{i=1}^{n} y_i \psi_j(x_i), \ j = \overline{1,k}.$$

З (7.6) за умови, що матриця C має обернену, можна отримати й аналітичний вираз для a, а саме $a = C^{-1} b$.

Приклад 7.2. Використовуючи метод найменших квадратів, знайти параметри апроксимуючої функції виду $y = a_0 + a_1 x + a_2 x^2$ для таблично заданої функції шляхом розв'язання лінійної системи рівнянь.

x	– 1,5	– 1	– 0,5	0	0,5	1	1,5	2	2,5
y	9,8	6,4	5,2	1,7	2,9	5,6	7,2	16,5	27,5

Розв'язання в математичному пакеті R

Відправними даними задачі є вектор значень аргумента x, вектор значень функції y, кількість точок спостереження n, кількість параметрів апроксимуючої функції k та векторна функція $Fi(x)$ відомих функцій від x:

```
x = c(-1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5)
y = c(9.8, 6.4, 5.2, 1.7, 2.9, 5.6, 7.2, 16.5, 27.5)
n = 9
k=3
# Вводимо векторну функцію Fi
Fi = function(x)
{
  z = c(1:3)
  z[1] = 1
  z[2] = x
  z[3] = x^2
  return( z )
}
```

Процедура знаходження параметрів апроксимуючої функції шляхом розв'язання системи лінійних рівнянь може мати вигляд:

```
MNKSolve = function(x, y, n, Fi, k)
{
  # Формуємо матрицю значень векторної функції Fi
  # в точках x[i] по рядкам
  FiX = matrix(nrow=n, ncol=k)
  for(i in 1:n)
  {
    FiX[i,] = Fi(x[i])
  }
```

```
  C = matrix(nrow=k, ncol=k)
     for(j in 1:k)
     {
          for(m in 1:k)
          {
               S = 0
                    for(i in 1:n)
                         S = S + FiX[i,j]*FiX[i,m]
               C[j,m] = S
          }
     }
  # Формуємо вектор d
  d = c(1:k)
  for(j in 1:k)
     {
          S = 0
          for(i in 1:n)
          {
               S = S + y[i]*FiX[i,j]
          }
          d[j] = S
     }
  # Розв'язуємо систему
  a = solve(C, d)
  return(a)
}
```

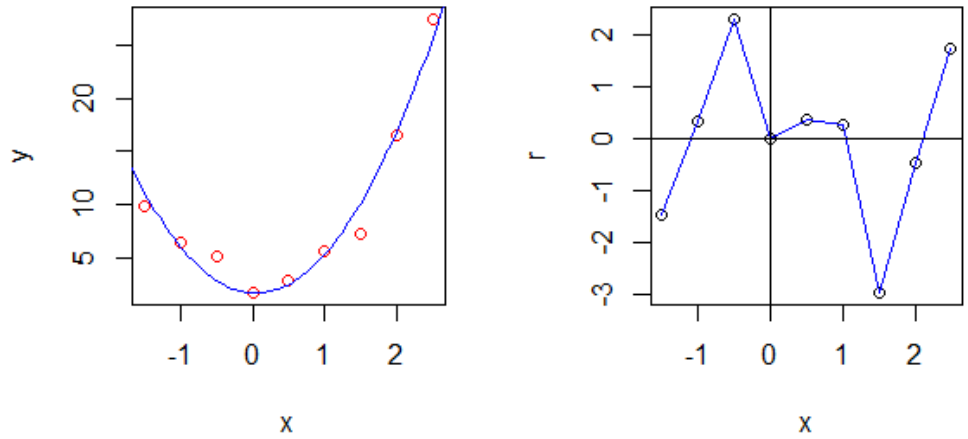
Результат знаходження параметрів апроксимуючої функції з використанням записаної процедури:

```
> a1 = MNKSolve(x, y, n, Fi, k)
> a1
[1]  1.7203463 -0.3640693  3.9974026
```

Таким чином, апроксимуюча функція має вигляд:

$$y = 1.72 - 0.364x + 3.997x^2 .$$

Графіки функцій (заданої та апроксимуючої) і графік залишків:



Варто розглянути ще випадок, коли апроксимуюча функція $g(x;a)$ є поліномом заданого $(k-1)$-го степеня. Тоді вона в загальному випадку може бути записана у вигляді:

$$g(x;a) = b_0 + b_1 x + b_2 x^2 + \cdots + b_{k-1} x^{k-1} = \sum_{i=0}^{k-1} b_i x^i ,$$

де $a \in R^k$, $a_1 = b_0$, $a_2 = b_1$, ..., $a_k = b_{k-1}$, b_j, $j = \overline{0,k-1}$, – коефіцієнти полінома. При цьому функція $g(x;a)$ є лінійною по параметрам a і має вигляд (7.4), де $\psi_1(x) = 1$, $\psi_2(x) = x$, …, $\psi_k(x) = x^{k-1}$, тобто $\psi_m(x) = x^{m-1}$, $m = \overline{1,k}$. Тоді параметри a можна знайти, розв'язавши систему лінійних рівнянь (7.6), з елементами

$$c_{jm} = \sum_{i=1}^{n} x_i^{m-1} x_i^{j-1} = \sum_{i=1}^{n} x_i^{m+j-2} , \; j = \overline{1,k} , \; m = \overline{1,k} ; \qquad (7.7)$$

$$b_j = \sum_{i=1}^{n} y_j x_i^{j-1} , \; j = \overline{1,k} . \qquad (7.8)$$

Приклад 7.3. Використовуючи метод найменших квадратів, знайти параметри апроксимуючої функції виду $y = a_0 + a_1 x + a_2 x^2 + a_3 x^3$ (поліном 3-го степеня) для таблично заданої функції шляхом розв'язання системи лінійних рівнянь.

x	– 1,5	– 1	– 0,5	0	0,5	1	1,5	2	2,5
y	9,8	6,4	5,2	1,7	2,9	5,6	7,2	16,5	27,5

Розв'язання в математичному пакеті R

Відправними даними задачі є вектор значень аргумента x, вектор значень функції y, кількість точок спостереження n та кількість параметрів апроксимуючої функції K:

```
x = c(-1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5)
y = c(9.8, 6.4, 5.2, 1.7, 2.9, 5.6, 7.2, 16.5, 27.5)
n = 9
k=4
```

Процедура знаходження параметрів апроксимуючої функції шляхом розв'язання системи лінійних рівнянь може бути записана так:

```
MNKSolvePolinom = function(x, y, n, k)
{
  # Формуємо матрицю C
  C = matrix(nrow=k, ncol=k)
     for(j in 1:k)
     {
          for(m in 1:k)
          {
               S = 0
               for(i in 1:n)
               {
                    S = S + x[i]^(m+j-2)
               }
          C[j,m] = S
          }
     }
  # Формуємо вектор d
  d = c(1:k)
  for(j in 1:k)
     {
       S = 0
       for(i in 1:n)
       {
          S = S + y[i]*x[i]^(j-1)
       }
       d[j] = S
     }
  # Розв'язуємо систему
  a = solve(C, d)
  return(a)
}
```

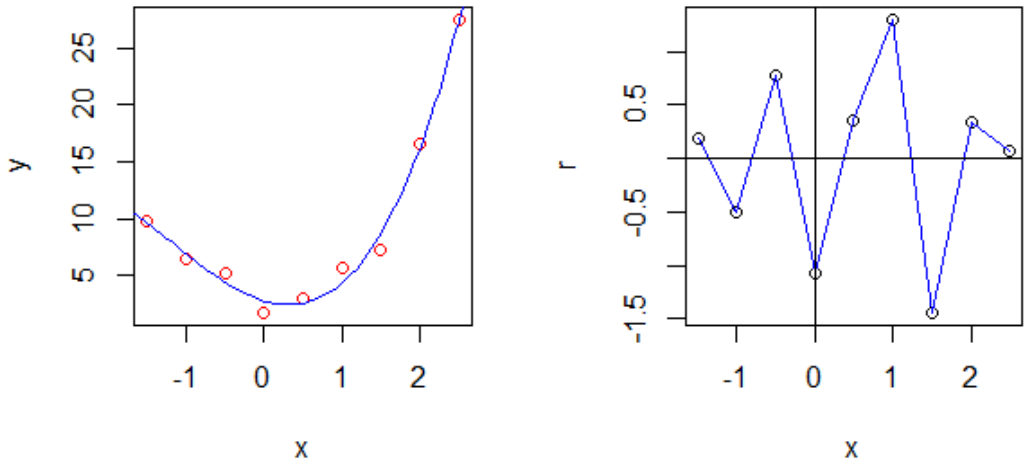
Результат знаходження параметрів апроксимуючої функції з використанням записаної процедури:

```
> # Знаходимо параметри апроксимуючої функції
> a1 = MNKSolvePolinom(x, y, n, k)
> a1
[1]  2.7748918 -2.0825878  2.8256854  0.7811448
```

Таким чином, апроксимуюча функція має вигляд:

$$y = 2.775 - 2.083x + 2.826x^2 + 0.781x^3.$$

Графіки функцій (заданої та апроксимуючої) і графік залишків:



7.3. Інтерполяція лінійна та квадратична

У випадках, коли вид функції $g(x;a)$ наперед невідомий, а значення y_i, $i=\overline{1,n}$ містять малу погрішність, задачу наближення таблично заданої функції $f(x)$ часто розв'язують як задачу інтерполяції.

Кусочно-лінійна інтерполяція

Такий вид інтерполяції полягає в тому, що на кожному окремому інтервалі (x_i, x_{i+1}) функція $f(x)$ наближається лінійною функцією $g_i(x;a_i,b_i)=a_i x+b_i$, $i=\overline{1,n-1}$ (рис. 7.1).

Оскільки значення функцій $f(x)$ і $g_i(x;a_i,b_i)$ в точках x_i і x_{i+1} повинні співпадати, то:

$$g_i(x_i;a_i,b_i)=a_i x_i+b_i=y_i,$$

$$g_i(x_{i+1};a_i,b_i)=a_i x_{+1i}+b_i=y_{i+1}.$$

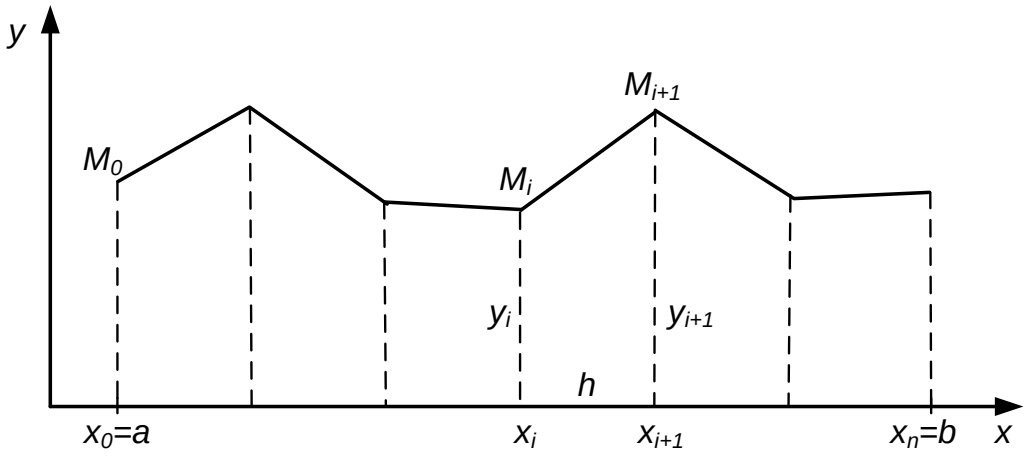


Рис. 7.1. **Графічна інтерпретація кусочно-лінійної інтерполяції**

Тоді, розв'язуючи отриману систему рівнянь відносно a_i, b_i, отримуємо:

$$a_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i},$$

$$b_i = y_i - a_i x_i.$$

Звідси випливає, що параметри a_i, b_i функції $g_i(x)$ повністю визначаються значеннями координат точок (x_i, y_i), (x_{i+1}, y_{i+1}). Неважко перевірити, що лінійна функція, яка проходить через 2 точки (x_i, y_i), (x_{i+1}, y_{i+1}), також може бути записана в зручнішому вигляді:

$$g_i(x) = \frac{(x - x_{i+1})}{(x_i - x_{i+1})} \cdot y_i + \frac{(x - x_i)}{(x_{i+1} - x_i)} \cdot y_{i+1}. \qquad (7.9)$$

Таким чином, у ході використання кусочно-лінійної інтерполяції для таблично заданої функції, для обчислення наближеного значення функції $f(x)$ у деякій точці $\bar{x} \in (x_1, x_n)$ діють таким чином. Спочатку знаходять інтервал (x_i, x_{i+1}), якому точка $\bar{x}$ належить, а потім обчислюють наближене значення функції $f(x)$ у точці $\bar{x}$ як $g_i(\bar{x})$, де функція $g_i(x)$ визначена в (7.9).

Варто зазначити, що інтерполююча функція $g(x)$ для функції $f(x)$ буде наче "зшитою" з лінійних функцій $g_i(x)$, кожна з яких розглядається на своєму "кусочку" (x_i, x_{i+1}), звідси і назва даного виду інтерполяції – кусочно-лінійна.

При цьому функція $g(x)$ по суті є наближеним розв'язком задачі наближення функції $f(x)$, тобто істинного розв'язку. Тому виникає питання про точність розв'язку цієї задачі, тобто про погрішність отриманого розв'язку. Слід розглянути випадок, коли значення y_i, $i = \overline{1,n}$ відомі точно, тобто $y_i = f(x_i)$, $i = \overline{1,n}$.

Оцінка погрішності. Нехай функція $f(x)$ двічі неперервно-диференційована на відрізку (x_1, x_n). Тоді для всіх $x \in (x_1, x_n)$:

$$|f(x) - g(x)| \le M_2 h^2,$$

де $M_2 = \max\limits_{\xi \in (x_1, x_n)} |f''(\xi)|$, $h = \max\limits_{i=1,n-1} |x_{i+1} - x_i|$.

Таким чином, погрішність кусочно-лінійної інтерполяції має порядок $O(h^2)$.

Кусочно-лінійна інтерполяція застосовується тоді, коли не вимагається великої точності наближення. Наприклад, при побудові графіків по точках тощо.

Приклад 7.4. Використовуючи кусочно-лінійну інтерполяцію, обчислити наближене значення функції, заданої таблично, в точці $x = -1.2$.

x	– 1,5	– 1	– 0,5	0	0,5	1	1,5	2	2,5
y	9,8	6,4	7,0	1,7	17,3	5,6	10,8	6,2	27,5

Розв'язання в математичному пакеті R

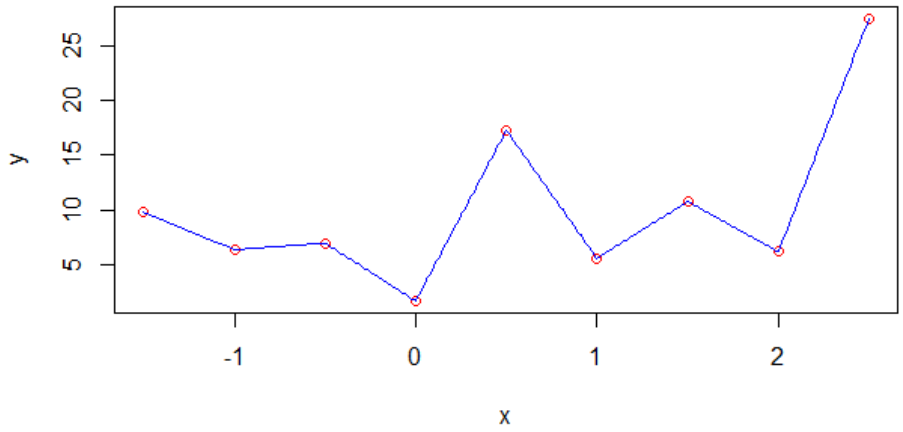
Відправними даними задачі є вектор значень аргумента x, вектор значень функції y та кількість точок спостереження n:

```
x = c(-1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5)
y = c(9.8, 6.4, 7, 1.7, 17.3, 5.6, 10.8, 6.2, 27.5)
n = 9
```

Процедуру, що реалізує кусочно-лінійну інтерполяцію, можна записати так:

```
# Кусково-лінійна інтерполяція
LinInterpol = function(x, y, n, xk)
{
  for (i in 1:(n-1))
  {
     if (xk <= x[i+1])
        break
  }
  yk = (xk-x[i+1])*y[i]/(x[i]-x[i+1])+
       (xk-x[i])*y[i+1]/(x[i+1]-x[i])

  return(yk)
}
#для побудови двох графіків
x1 = seq(-1.5, 2.5, by=0.01)
for(i in 1:length(x1))
     y1m[i] = LinInterpol(x, y, n, x1[i])
# рисуємо перший графік у вигляді точок
plot(x, y, type="p", col="red")
# додаємо в 1-у графічну підобласть ще лінію
lines(x1, y1m, col="blue")
```

Графіки заданої та інтерполюючої кусочно-лінійної функції:



Обчислення значення функції в заданій точці з використанням записаної процедури:

```
> xk = 2.1
> yk = LinInterpol(x, y, n, xk)
> yk
[1] 10.46
```

Отже, значення функції в точці $x = 2.1$, отримане методом кусочно-лінійної інтерполяції, дорівнює 10.46.

Кусочно-квадратична інтерполяція

Подібно до кусочно-лінійної інтерполяції можна використовувати для наближення на кожному окремому інтервалі (x_i, x_{i+1}) і поліноми вищого порядку, наприклад квадратичну функцію $g(x;a,b,c) = ax^2 + bx + c$. У цьому випадку для знаходження коефіцієнтів a, b, c слід використовувати додаткову точку x_{i+2} (три невідомих, тому необхідно і три рівняння для їх визначення) (рис. 7.2).

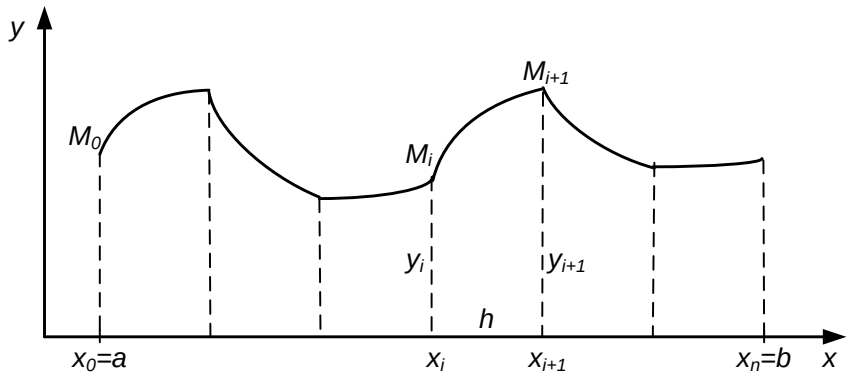


Рис. 7.2. **Графічна інтерпретація кусочно-квадратичної інтерполяції**

Неважко перевірити, що квадратична функція, що проходить через 3 точки (x_i, y_i), (x_{i+1}, y_{i+1}), (x_{i+2}, y_{i+2}), може бути записана у вигляді:

$$g_i(x) = \frac{(x - x_{i+1})(x - x_{i+2})}{(x_i - x_{i+1})(x_i - x_{i+2})} \cdot y_i + \frac{(x - x_i)(x - x_{i+2})}{(x_{i+1} - x_i)(x_{i+1} - x_{i+2})} \cdot y_{i+1} + \\ + \frac{(x - x_i)(x - x_{i+1})}{(x_{i+2} - x_i)(x_{i+2} - x_{i+1})} \cdot y_{i+2} \quad . \qquad (7.10)$$

Оцінка погрішності. Нехай функція $f(x)$ тричі неперервно-диференційована на відрізку (x_1, x_n). Тоді для всіх $x \in (x_1, x_n)$:

$$|f(x) - g(x)| \le M_3 h^3,$$

де $M_3 = \max\limits_{\xi \in (x_1, x_n)} |f^{(3)}(\xi)|$, $h = \max\limits_{i=1,n-1} |x_{i+1} - x_i|$.

Таким чином, погрішність кусочно-квадратичної інтерполяції має порядок $O(h^3)$.

Приклад 7.5. Використовуючи кусочно-квадратичну інтерполяцію, обчислити наближене значення функції, заданої таблично, в точці $x = -1.2$.

x	– 1,5	– 1	– 0,5	0	0,5	1	1,5	2	2,5
y	9,8	6,4	7,0	1,7	17,3	5,6	10,8	6,2	27,5

Розв'язання в математичному пакеті R

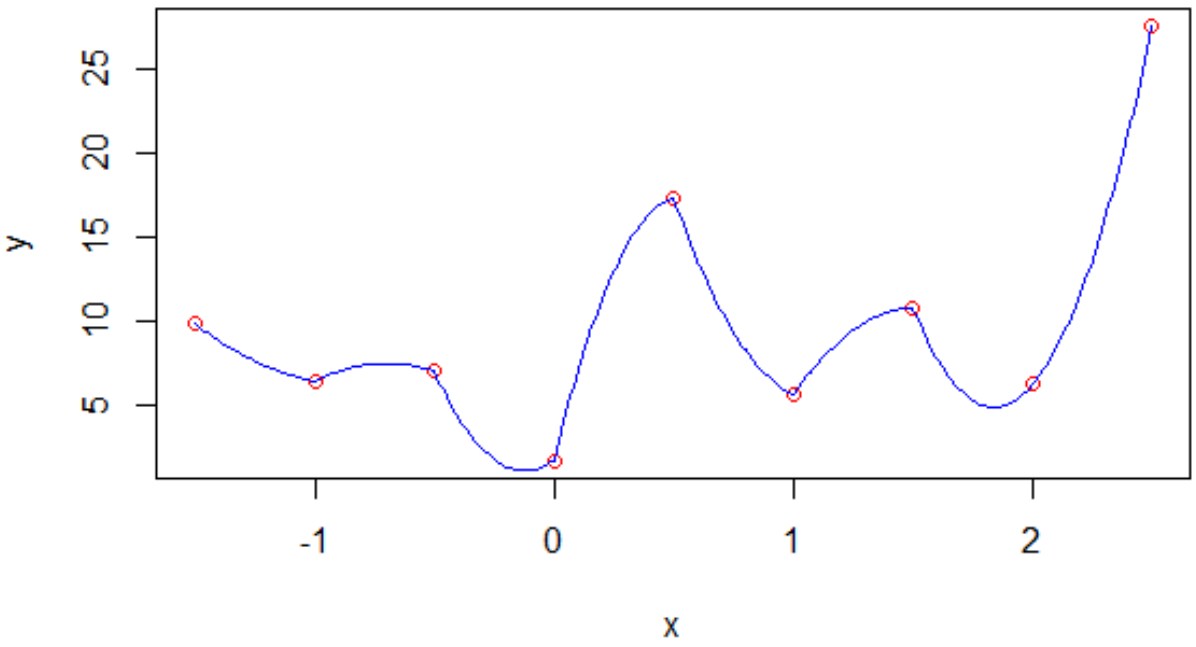
Відправними даними задачі є вектор значень аргумента x, вектор значень функції y та кількість точок спостереження n:

```
x = c(-1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5)
y = c(9.8, 6.4, 7, 1.7, 17.3, 5.6, 10.8, 6.2, 27.5)
n = 9
```

Процедура, що реалізує кусочно-квадратичну інтерполяцію, може бути записана так:

```
# Кусково-квадратична інтерполяція
KwInterpol = function(x, y, n, xk)
{
  for (i in 1:(n-2))
  {
     if (xk <= x[i+1])
        break
  }
  yk = (xk-x[i+1])*(xk-x[i+2])*y[i]/((x[i]-x[i+1])*(x[i]-x[i+2]))+
     (xk-x[i])*(xk-x[i+2])*y[i+1]/((x[i+1]-x[i])*(x[i+1]-x[i+2]))+
     (xk-x[i])*(xk-x[i+1])*y[i+2]/((x[i+2]-x[i])*(x[i+2]-x[i+1]))

  return(yk)
}
```

Графіки заданої та інтерполюючої кусочно-квадратичної функції:



Обчислення значення функції в заданій точці з використанням записаної процедури:

```
> xk = 2.1
> yk = KwInterpol(x, y, n, xk)
> yk
[1] 8.388
```

Отже, значення функції в точці $x = 2.1$, отримане методом кусочно-квадратичної інтерполяції, дорівнює 8.388.

7.4. Інтерполяційний поліном Лагранжа

Якщо відомі значення y_i функції $f(x)$ в n точках x_i, $i=\overline{1,n}$, то поліномом мінімального степеня, що інтерполює функцію $f(x)$,матиме степінь $n-1$. Якщо він записаний у вигляді:

$$L_{n-1}(x)=\sum_{i=1}^{n}\left[\prod_{\substack{j=1\\ j\neq i}}^{n}\left(\frac{x-x_j}{x_i-x_j}\right)\right]y_i, \qquad (7.11)$$

то його називають інтерполяційним багаточленом Лагранжа.

Неважко перевірити, що $L_{n-1}(x_i)=f(x_i)$ для всіх $i=\overline{1,n}$.

Погрішність наближення інтерпольованої функції багаточленом Лагранжа сильно залежить від розкиду точок x_i, тобто чим менше інтервал (x_1,x_n), тим точніше $L_{n-1}(x)$ наближає значення функції $f(x)$ в точках $x\in(x_1,x_n)$.

Оцінка погрішності. Нехай функція $f(x)$ n разів неперервно-диференційована на відрізку (x_1,x_n). Тоді [16; 21] для всіх $x\in(x_1,x_n)$:

$$\left|f(x)-L_{n-1}(x)\right|\le\frac{M_n h^n}{n!},$$

де $M_n=\max\limits_{\xi\in(x_1,x_n)}\left|f^{(n)}(\xi)\right|$, $h=\max\limits_{i=1,n-1}\left|x_{i+1}-x_i\right|$.

Таким чином, погрішність інтерполяції за допомогою багаточлена Лагранжа має порядок $O(h^n)$.

Однак між вузлами інтерполяції x_i поліном Лагранжа, як правило, нестійкий до погрішностей обчислень, причому нестійкість зростає зі збільшенням n. Крім того, навіть невеликі погрішності значень y_i можуть сильно змінити поведінку полінома між вузлами. Зазначені ефекти нестійкості виявляються вже при $n\ge 15$, а в практичних задачах, коли n близьке до 100, нестійкість полінома Лагранжа дуже велика.

Приклад 7.6. Використовуючи поліном Лагранжа, обчислити наближене значення таблично заданої функції в точці $x = -1.2$.

x	– 1,5	– 1	– 0,5	0	0,5	1	1,5	2	2,5
y	9,8	6,4	7,0	1,7	17,3	5,6	10,8	6,2	27,5

Розв'язання в математичному пакеті R

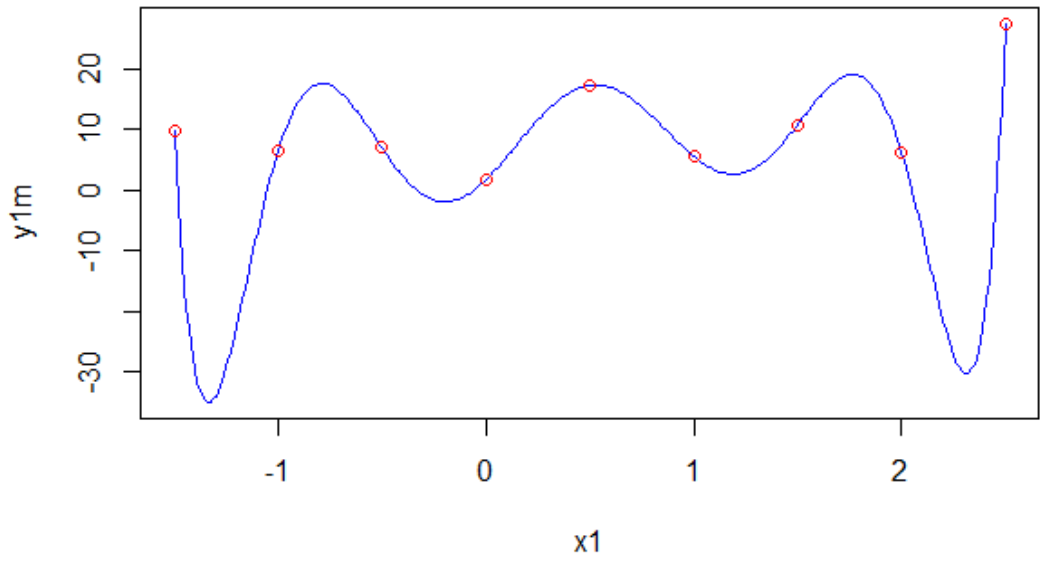
Відправними даними задачі є вектор значень аргумента x, вектор значень функції y та кількість точок спостереження n:

```
x = c(-1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5)
y = c(9.8, 6.4, 7, 1.7, 17.3, 5.6, 10.8, 6.2, 27.5)
n = 9
```

Поліном Лагранжа в загальному вигляді записується так:

```
LagrangeInterpol = function(x, y, n, xk)
{
    yk = 0
    for (i in 1:n)
    {
        P = 1
        for (j in 1:n)
        {
            if (j!=i)
                P = P*((xk-x[j])/(x[i]-x[j]))
        }
        yk = yk + P*y[i]
    }
    return(yk)
}
```

Графіки заданої функції та полінома Лагранжа:



Обчислення значення функції в заданій точці з використанням записаної процедури:

```
> xk =  2.1
> yk = LagrangeInterpol(x, y, n, xk)
> yk
[1] -6.982716
```

Отже, значення функції в точці $x = 2.1$, обчислене за формулою полінома Лагранжа, дорівнює – 6.98.

7.5. Інтерполяційний поліном Ньютона

На відміну від (7.11), поліном мінімального степеня, що інтерполює функцію $f(x)$ в n точках x_i, $i = \overline{1,n}$, може бути записаний в іншому вигляді. Вводяться такі позначення:

розділені різниці 1-го порядку:

$$f(x_1; x_2) = \frac{f(x_2) - f(x_1)}{x_2 - x_1}\ , \ldots,\ f(x_{n-1}; x_n) = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}};$$

розділені різниці 2-го порядку:

$$f(x_1; x_2; x_3) = \frac{f(x_2; x_3) - f(x_1; x_2)}{x_3 - x_1},\ldots,$$

$$f(x_{n-2}; x_{n-1}; x_n) = \frac{f(x_{n-1}; x_n) - f(x_{n-2}; x_{n-1})}{x_n - x_{n-2}};$$

розділена різниця $(n-1)$-го порядку:

$$f(x_1; x_2; \ldots; x_n) = \frac{f(x_2; x_2; \ldots x_n) - f(x_1; x_2; \ldots x_{n-1})}{x_n - x_1}.$$

Тоді розглядається поліном $(n-1)$-го степеня виду

$$\begin{aligned} H_{n-1}(x) &= f(x_1) + (x - x_1) f(x_1; x_2) + \\ &+ (x - x_1)(x - x_2) f(x_1; x_2; x_3) + \ldots + \\ &+ (x - x_1)(x - x_2)\ldots(x - x_{n-1}) f(x_1; x_2; \ldots; x_n). \end{aligned} \quad . \qquad (7.12)$$

Неважко перевірити, що $H_{n-1}(x_i) = f(x_i)\ \forall i = \overline{1,n}$, тобто поліном $H_{n-1}(x)$ є інтерполяційним для функції $f(x)$. Його називають **інтерполяційним поліномом Ньютона**.

Інтерполяційний поліном Ньютона виявляється дуже корисним при розв'язанні інших задач чисельного аналізу, що буде показано далі.

7.6. Сплайн-інтерполяція

Інтерполяція за допомогою поліномів Лагранжа або Ньютона з використанням великої кількості вузлів часто призводить до поганого наближення, що пояснюється накопиченням великої погрішності в процесі обчислень. З іншого боку, кусочно-лінійна і кусочно-квадратична інтерполяція не дозволяють добитися гарного наближення зважаючи на їх теоретичну неточність. Одним із способів добитися гарного наближення в практичних задачах є інтерполяція за допомогою сплайн-функцій.

Визначення. Сплайном (з англ. *spline* – рейка, лінійка) називається кусочно-поліноміальна функція, визначена на відрізку (x_1, x_n) і яка має на цьому відрізку не менше двох неперервних похідних. Сплайн буде позначено через $S(x)$. Інтерполяція за допомогою сплайнів називається **сплайн-інтерполяцією**.

Розглянемо **кубічну сплайн-інтерполяцію** [23], яка найчастіше застосовується на практиці, тобто на кожному інтервалі (x_i, x_{i+1}), $i = \overline{1, n-1}$ функція $S(x)$ задається поліномом третьої степені:

$$S_i(x) = y_i + c_{1i}(x - x_i) + c_{2i}(x - x_i)^2 + c_{3i}(x - x_i)^3. \qquad (7.13)$$

Очевидно, $S(x_i) = S_i(x_i) = y_i$, $i = \overline{1, n-1}$.

Слід зазначити, що у випадку застосування кубічного сплайна він повинен мати дві неперервні похідні на відрізку (x_1, x_n).

Для того щоб побудувати кубічний сплайн $S(x)$, необхідно знайти всі коефіцієнти c_{1i}, c_{2i}, c_{3i} $(i = \overline{1, n-1})$ поліномів $S_i(x)$, $i = \overline{1, n-1}$. Кількість цих невідомих коефіцієнтів $3(n-1)$.

Коефіцієнти c_{1i}, c_{2i}, c_{3i} $(i = \overline{1, n-1})$ підбираються так, щоб на межах інтервалів (x_i, x_{i+1}) забезпечити неперервність, як функції $S(x)$, так і її першої $S'(x)$ і другої $S''(x)$ похідних. Варто зазначити, що згідно з (7.13) для всіх $i = \overline{1, n-1}$

$$S_i'(x\ ) = c_{1i} + 2c_{2i}(x - x_i) + 3c_{3i}(x - x_i)^2,$$

$$S_i''(x) = 2c_{2i} + 6c_{3i}(x - x_i).$$

Таким чином, повинні виконуватися умови:

$$\begin{gathered} S_i(x_{i+1}) = S_{i+1}(x_{i+1}) = y_{i+1},\ i = \overline{1,n-2};\ S_{n-1}(x_n) = y_n; \\ S'_i(x_{i+1}) = S'_{i+1}(x_{i+1}) = c_{1,i+1},\ i = \overline{1,n-2}; \\ S''_i(x_{i+1}) = S''_{i+1}(x_{i+1}) = c_{2,i+1},\ i = \overline{1,n-2}. \end{gathered} \tag{7.14}$$

Можна переписати (7.14) в більш наочному вигляді:

$$\begin{cases} y_i + c_{1i}(x_{i+1} - x_i) + c_{2i}(x_{i+1} - x_i)^2 + c_{3i}(x_{i+1} - x_i)^3 = y_{i+1}, i = \overline{1,n-1}, \\ c_{1i} + 2c_{2i}(x_{i+1} - x_i) + 3c_{3i}(x_{i+1} - x_i)^2 = c_{1,i+1}, i = \overline{1,n-2}, \\ 2c_{2i} + 6c_{3i}(x_{i+1} - x_i) = 2c_{2,i+1}, i = \overline{1,n-2}. \end{cases} \tag{7.15}$$

Як видно, система (7.15) визначає $3(n-1)-2$ лінійних рівняння для пошуку $3(n-1)$ невідомих c_{1i}, c_{2i}, c_{3i} ($i = \overline{1,n-1}$). Тому необхідно довизначити ще два рівняння. Для цього задають так звані **граничні умови** – значення першої або другої похідної функції $f(x)$ на межах інтервалу $(x_1, x_n)(x_1, x_n)$. Якщо значення однієї з похідних на межах відомі, то задавши їх, можна отримати дуже точну інтерполяційну схему. Якщо ж ці значення невідомі, то можна задати другу похідну на межах рівною нулю і отримати також достатньо гарні результати. Необхідно розглянути простий випадок, коли:

$$f''(x_1) = 0, f''(x_n) = 0,$$

тобто

$$\begin{cases} S''_1(x_1) = 2c_{21} = 0, \\ S''_{n-1}(x_n) = 2c_{2,n-1} + 6c_{3,n-1}(x_n - x_{n-1}) = 0. \end{cases} \tag{7.16}$$

Таким чином, для обчислення коефіцієнтів c_{1i}, c_{2i}, c_{3i}, $i = \overline{1,n-1}$, необхідно розв'язати систему лінійних алгебраїчних рівнянь (7.15), (7.16).

Оцінка погрішності. Нехай функція $f(x)$ 4 рази неперервно-диференційована на відрізку (x_1, x_n). Тоді [16; 21] для всіх $x \in (x_1, x_n)$:

$$|f(x) - S(x)| \le \frac{M_4 h^4}{8},$$

де $M_4 = \max\limits_{\xi \in (x_1, x_n)} \left| f^{(4)}(\xi) \right|$, $h = \max\limits_{i=1,n-1} |x_{i+1} - x_i|$.

Таким чином, погрішність інтерполяції за допомогою кубічного сплайна має порядок $O(h^4)$. Перевагою кубічних сплайнів перед іншими наведеними способами інтерполяції є: по-перше, їх стійкість до процесу обчислень, і, по-друге, достатньо висока точність.

При використовуванні сплайн-інтерполяції для таблично заданої функції діють таким чином. Спочатку один раз визначають всі коефіцієнти c_{1i}, c_{2i}, c_{3i}, $i = \overline{1,n-1}$, а потім вже для обчислення наближеного значення функції $f(x)$ у деякій точці $\bar{x} \in (x_1, x_n)$ спочатку знаходять інтервал (x_i, x_{i+1}), якому точка $\bar{x}$ належить, а потім обчислюють наближене значення функції $f(x)$ в точці $\bar{x}$ як $S_i(\bar{x})$, де функція визначена в (7.13).

7.7. Поняття екстраполяції функцій

На відміну від задачі інтерполяції (з англ. *inter* – між), тобто відновлення функції між полюсами, задача екстраполяції (з англ. *extra* – додатково) полягає в відновленні функції за межами полюсів. Тобто, якщо для деякої функції $f(x)$, $f : R^1 \to R^1$ відомо, що в n точках $x_1, x_2, \ldots, x_n$ вона приймає, відповідно, значення $y_1, y_2, \ldots, y_n$, тобто $y_i = f(x_i)$, $i = \overline{1,n}$, то потрібно відновити її значення при $x > x_n$ або $x < x_1$.

Така задача виникає, наприклад, при прогнозуванні в часі значень деякого показника (економічного або фізичного) на основі відомих його значень у вже минулих моментах часу. В цьому випадку x – це час, y – значення показника. Відомий набір значень $y_1, y_2, \ldots, y_n$ називається **часовим рядом**. Прикладами часових рядів є: курси деяких акцій, поквартальний обсяг виробництва, річний ВВП країни, добовий обсяг

продажів, погодинний обсяг водоспоживання міста, середньорічна температура на планеті Земля та ін.

У процесі розв'язання задачі екстраполяції застосовують методи апроксимації (див. розділ 7.2), оскільки значення y_i, як правило, містять в собі значні випадкові шуми і тому методи інтерполяції дають велику похибку наближення. Для прогнозування часових рядів, окрім апроксимації (прямі методи), застосовують і спеціальні статистичні методи (адаптивні або стохастичні) [26].

7.8. Висновки

1. Наближення функцій є важливим допоміжним апаратом при розв'язанні інших задач чисельного аналізу.

2. Найбільш відомим і ефективним методом розв'язання задачі апроксимації функцій є метод найменших квадратів.

3. Одним із способів добитися гарного наближення в практичних задачах є інтерполяція за допомогою сплайн-функцій.

7.9. Контрольні запитання та завдання

1. Сформулюйте постановку задачі наближення функцій. Які постановки задачі наближення функцій ви ще знаєте?

2. Чим відрізняються задачі апроксимації, інтерполяції та екстраполяції функцій?

3. У чому полягає метод найменших квадратів для апроксимації функцій?

4. У чому полягає лінійна інтерполяція функцій? Коли її слід застосовувати?

5. У чому полягає квадратична інтерполяція функцій? Коли її слід застосовувати?

6. Що називається інтерполяційним поліномом Лагранжа? Коли його слід застосовувати?

7. Що називають сплайном?

8. У чому полягає лінійна сплайн-інтерполяція функції? Коли її слід застосовувати?

9. Використовуючи метод найменших квадратів, підберіть найкращу апроксимацію для функції, заданої таблично

x	1,6	1,8	2	2,2	2,4	2,6
y	6,6	6,7	6,2	5,8	4,3	3,4

серед функцій виду:

1) $y = a_0 + a_1 x$;

2) $y = a_0 + a_1 x + a_2 x^2$;

3) $y = a_0 + a_1 \sin(a_2 x)$.

Побудуйте графіки апроксимуючих функцій і графіки залишків.

10. Побудуйте кусочно-лінійну та кусочно-квадратичну інтерполяцію для функції, заданої таблицеюв завданні 8.

8. Чисельне диференціювання функцій

8.1. Постановка задачі

Задача чисельного диференціювання полягає в знаходженні значень похідних функції $y = f(x)$ в заданих точках у випадках, коли аналітичний вид функції $f(x)$ невідомий (задана неявно), дуже складний або функція $f(x)$ задана таблицею. Привабливість чисельного підходу пояснюється наявністю простих формул, за допомогою яких похідні в заданих точках можна приблизно обчислити за кількома значеннями функції $f(x)$ в цих та близьких до них точках.

8.2. Формули чисельного диференціювання

Формули чисельного диференціювання застосовуються в тих випадках, коли $y = f(x)$ може бути задана таблицею своїх значень $y_i = f(x_i)$ у рівновіддалених вузлах $x_i = x_0 + i \times h \; i = 0, \pm 1, \pm 2, \ldots$. Варто зазначити, що нумерація точок x_i для зручності запису проводиться відносно точки x_0, в якій обчислюється похідна. Вибравши яку-небудь множину $(n+1)$-го вузлів, функцію $y = f(x)$ наближають інтерполяційним багаточленом $P_n(x)$. Тоді похідна від цього багаточлена $P_n'(x)$ в точці x_0 застосовується для наближеного подання шуканої похідної $y'(x_0) = f'(x_0)$, а саме $f'(x_0) \approx P_n'(x_0)$.

Найбільш зручним інтерполяційним багаточленом для чисельного диференціювання є поліном Ньютона (див. розділ 7.5). На його основі отримані формули різного порядку точності залежно від кількості задіяних точок x_i [1; 21].

Слід навести кілька найпростіших формул для похідної першого порядку:

формула диференціювання вперед

$$f'(x_0) \approx \frac{f(x_0+h)-f(x_0)}{h} = \frac{y_1-y_0}{h}; \qquad (8.1)$$

формула диференціювання назад

$$f'(x_0) \approx \frac{f(x_0)-f(x_0-h)}{h} = \frac{y_0-y_{-1}}{h}; \qquad (8.2)$$

симетрична формула диференціювання

$$f'(x_0) \approx \frac{f(x_0+h)-f(x_0-h)}{2h} = \frac{y_1-y_{-1}}{2h}. \qquad (8.3)$$

Варто зазначити, що запис формул через функцію $f(x)$ відповідає постановці задачі диференціювання, коли її аналітичний вигляд невідомий або дуже складний. Це означає, що значення функції $f(x)$ можна розрахувати в потрібних точках x_i з кроком h. При цьому h називають кроком чисельного диференціювання та підбирають залежно від поведінки функції $f(x)$ в околі точки x_0.

Запис формул через y_i відповідає постановці задачі диференціювання, коли функція $f(x)$ задана таблицею. При цьому для крайніх точок x_i можна застосовувати формули диференціювання вперед та назад (відповідно до того, з якого вони боку), а для внутрішніх точок краще застосовувати симетричну формулу диференціювання.

Погрішність формул (8.1) та (8.2) порядку $O(h)$, формули (8.3) – $O(h^2)$ [3].

Для похідних 1-го порядку (додатково до формул (8.1) – (8.3)) можна застосовувати **симетричну формулу диференціювання**

$$f'(x_0) \approx \frac{-y_2+8y_1-8y_{-1}+y_{-2}}{12h}. \qquad (8.4)$$

Погрішність формули (8.4) – $O(h^4)$ [3; 16; 21].

Варто навести кілька найпростіших формул для похідної другого порядку:

симетричні формули диференціювання:

$$f''(x_0) \approx \frac{f(x_0+h)-2f(x_0)+f(x_0-h)}{h^2} = \frac{y_1-2y_0+y_{-1}}{h^2}, \qquad (8.5)$$

$$f''(x_0) \approx \frac{-y_2+16y_1-30y_0+16y_{-1}-y_{-2}}{12h^2}. \qquad (8.6)$$

Погрішність формули (8.5) порядку $O(h)$, формули (8.6) – $O(h^3)$ [3].

Треба зазначити, що крок чисельного диференціювання h не можна брати дуже малим, бо тоді, внаслідок похибок округлення на комп'ютері, похибка розрахунку похідної з застосуванням формул чисельного диференціювання може бути дуже великою. Значення h також краще брати залежно від величини x_0, наприклад $h = h_R \times |x_0|$, де h_R знаходиться у межах від 10^{-6} до 10^{-2} та підбирають залежно від поведінки функції $f(x)$ в околі точки x_0.

Приклад 8.1. Знайти чисельно першу похідну функції, заданої таблично, в усіх точках x_i.

x	– 1,5	– 1	– 0,5	0	0,5	1	1,5	2	2,5
y	9,8	6,4	5,2	1,7	2,9	5,6	7,2	16,5	27,5

Розв'язання в математичному пакеті R

Відправними даними задачі є вектор значень аргумента X, вектор значень функції Y та кількість точок спостереження N:

```
X = c(-1.5, -1, -0.5, 0, 0.5, 1, 1.5, 2, 2.5)
Y = c(9.8, 6.4, 5.2, 1.7, 2.9, 5.6, 7.2, 16.5, 27.5)
N = 9
```

Процедуру обчислення першої похідної за формулами чисельного диференціювання для таблично заданої функції в усіх точках x_i можна записати так:

```
# Похідна таблично заданої функції
PohidnaTabl = function(x, y, n)
{
     df=c(1:n)
     h = x[2]-x[1]
     for (i in 1:n)
     {
          if (i==1)
               df[i]=(y[2]-y[1])/h
          else
          {
               if (i==n)
                    df[i]=(y[n]-y[n-1])/h
               else df[i]=(y[i+1]-y[i-1])/(2*h)
          }
     }
     return (df)
}
```

Результат обчислення похідних із використанням записаної процедури:

```
>  DF = PohidnaTabl(X, Y, N)
>  DF
[1] -6.8 -4.6 -4.7 -2.3  3.9  4.3 10.9 20.3 22.0
```

Приклад 8.2. Знайти чисельно першу похідну функції $f(x) = x^3 - 5x + 2$ в точці $x = 3$, прийнявши крок диференціювання $h = 0{,}01$.

Розв'язання в математичному пакеті R

Відправними даними задачі є функція $F(x)$. Додатково задається точка X, в якій необхідно обчислити похідну, крок диференціювання h та тип формули, за якою проводити обчислення tf (при $tf = 1$ використовується формула диференціювання вперед, при $tf = -1$ – формула диференціювання назад, при $tf = 0$ – симетрична формула диференціювання):

```
F = function(x)
{
 return (x^3-5*x+2)
}
X = 3
h = 0.01
tf = 1
```

Процедуру обчислення першої похідної за формулами чисельного диференціювання для аналітично заданої функції можна записати так:

```
# Похідна аналітично заданої функції
PohidnaAnalyt = function(f, x, h, tf)
{
     if (tf==1)
          df=(f(x+h)-f(x))/h
     else
     {
          if (tf==-1)
               df=(f(x)-f(x-h))/h
          else
          {
               if (tf==0)
                    df=(f(x+h)-f(x-h))/(2*h)
          }
     }
  return (df)
 }
```

Результат обчислення похідної в заданій точці з використанням записаної процедури:

```
>  DF = PohidnaAnalyt(F, X, h, tf)
>  DF
[1] 22.0901
```

8.3. Висновки

1. Чисельне диференціювання функції в заданій точці здійснюється за простими формулами, в яких присутні значення функції в цій точці та кількох близьких до неї точках.

2. Формули чисельного диференціювання відрізняються точністю та способом застосування.

3. Крок чисельного диференціювання не можна брати дуже малим у зв'язку з похибкою округлення та розрахунків на комп'ютері.

4. При табличному задаванні функції значення похідних розраховуються тільки в вузлових точках.

8.4. Контрольні запитання та завдання

1. Сформулюйте постановку задачі диференціювання функцій.
2. У яких випадках виникає задача чисельного диференціювання?
3. Що називається інтерполяційним багаточленом Ньютона? Як його застосовують при виведенні формул чисельного диференціювання функцій?
4. На якому принципі заснований чисельний розрахунок похідних функції в точці?
5. Який інтерполяційний багаточлен є найбільш зручним для чисельного диференціювання?
6. У яких випадках краще застосовувати формули диференціювання вперед, назад та симетричні формули?
7. Знайдіть чисельно першу та другу похідну в точці $x = 3$ функції, заданої таблицею

x	2	2,5	3	3,5	4	4,5
y	5,5	4,5	3	2	4,5	7

9. Чисельне інтегрування функцій

9.1. Постановка задачі

Задача чисельного інтегрування полягає в обчисленні визначеного інтеграла

$$I = \int_a^b f(x)dx \qquad (9.1)$$

у випадках, коли аналітичне обчислення неможливе або дуже складне.

Методи чисельного обчислення інтеграла засновані на тому, що в якості наближеного значення інтеграла (9.1) береться значення інтеграла від інтерполюючої для $f(x)$ функції, побудованої по точках розбиття відрізка $[a,b]$.

Слід навести найбільш відомі та достатньо ефективні методи розв'язання задачі чисельного інтегрування функцій: метод трапецій і метод Сімпсона.

9.2. Формула трапецій

Відрізок $[a,b]$ розбивається на n частин з кроком $h=\dfrac{b-a}{n}$, при цьому точки розбиття x_i визначаються за формулою $x_i=a+i\times h$, $i=\overline{0,n}$, тобто $x_0=a$, $x_n=b$.

Метод трапецій заснований на кусочно-лінійній інтерполяції функції $f(x)$, побудованій по точках $M_i=(x_i,f(x_i))$, $i=\overline{0,n}$ (рис. 9.1).

Оскільки площа S_i трапеції $x_iM_iM_{i+1}x_{i+1}$ на інтервалі (x_i, x_{i+1}) дорівнює

$$S_i=h\frac{f(x_i)+f(x_{i+1})}{2},$$

то, додаючи площі всіх прямолінійних трапецій, і отримуємо **формулу трапецій**

$$S_n=\sum_{i=0}^{n-1}S_i=h\sum_{i=0}^{n-1}\frac{f(x_i)+f(x_{i+1})}{2}=h\left[\frac{f(x_0)+f(x_n)}{2}+\sum_{i=1}^{n-1}f(x_i)\right]. \qquad (9.2)$$

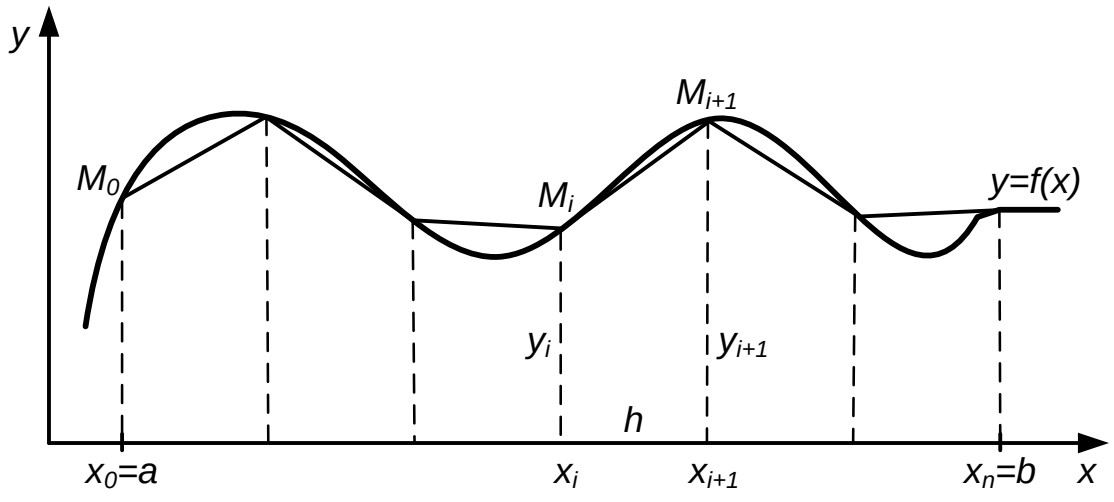


Рис. 9.1. **Графічна інтерпретація методу трапецій**

Оцінка погрішності формули трапецій [3; 5; 6]:

$$|S_n-I|\le\frac{b-a}{12}M_2h^2, \qquad (9.3)$$

де. $M_2=\max\limits_{\xi\in(a,b)}|f''(\xi)|$.

Варто зазначити, що число розбиття n відрізка $[a,b]$ є параметром формули трапецій (9.2), тобто чим більше n, тим менше h, а значить, менше погрішність.

З оцінки (9.3) виходить, що якщо функція $f(x)$ лінійна, то формула (9.2) для обчислення інтеграла (9.1) є теоретично точною.

Приклад 9.1. Обчислити значення визначеного інтеграла $\int_3^{15}\left(5x-\frac{4}{(2x-4)^3}\right)dx$ за формулою трапецій для числа розбиття інтервалу інтегрування $n=50$.

Розв'язання в математичному пакеті R

Відправними даними задачі є підінтегральна функція $F(x)$, межі інтегрування a і b, число розбиття n інтервалу $[a,b]$:

```
F = function(x)
{
  return (5*x-4/((2*x-4)^3))
}
a = 3
b = 15
n = 50
```

Процедуру обчислення визначеного інтеграла за формулою трапецій можна записати так:

```
trapec = function(f, a, b, n)
{
     h = (b-a)/n
     s = (f(a)+f(b))/2
     x = c(1:(n+1))
     for (i in 2:n)
     {
          x[i] = a + (i-1)*h
          s = s + f(x[i])
     }
     return (s*h)
}
```

Результат обчислення заданого визначеного інтеграла з використанням записаної процедури:

```
> S = trapec(F, a, b, n) # виклик функції користувача
> S
[1] 539.7444
```

9.3. Формула Сімпсона

Відрізок $[a,b]$ розбивається на n частин з кроком $h=\frac{b-a}{n}$, при цьому точки розбиття x_i визначаються за формулою $x_i = a + i \times h$, $i = \overline{0,n}$, тобто $x_0 = a$, $x_n = b$.

Метод Сімпсона заснований на кусочно-квадратичній інтерполяції функції $f(x)$, побудованій по точках $(x_i, f(x_i))$, $i = \overline{0,n}$ (рис. 9.2).

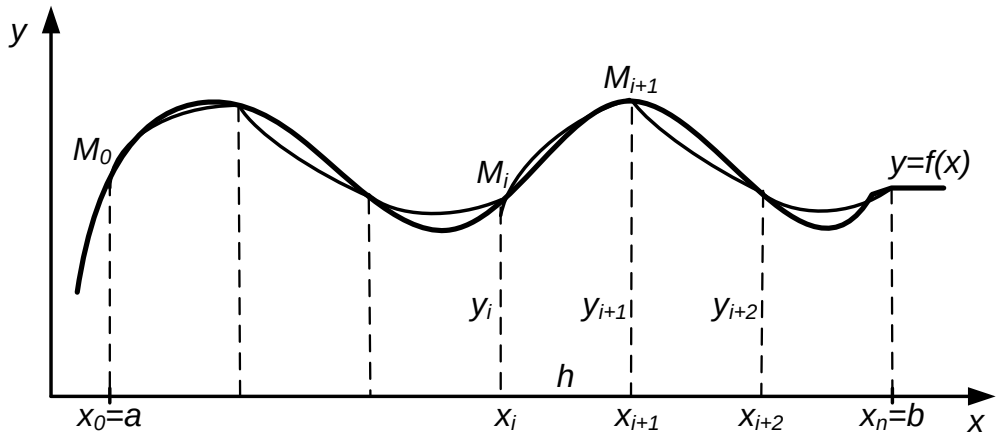


Рис. 9.2. **Графічна інтерпретація методу Сімпсона**

Оскільки на кожному інтервалі (x_i, x_{i+1}) функція $f(x)$ інтерполюється параболою (тобто функцією виду $g_i(x) = a_i x^2 + b_i x + c_i$), то площу S_i криволінійної трапеції $x_i M_i M_{i+1} x_{i+1}$ нескладно обчислити аналітично

$$S_i = \int_{x_i}^{x_{i+1}} g_i(x)\,dx .$$

Тоді наближене значення інтеграла (9.1) буде дорівнювати

$$S_n = \sum_{i=0}^{n-1} S_i .$$

Формула Сімпсона в загальному випадку має вигляд [10]:

$$\int_a^b f(x)dx \approx \frac{h}{3}[f(x_0) + f(x_n) + 2(f(x_2) + f(x_4) + \cdots + f(x_{n-2})) +$$
$$+ 4(f(x_1) + f(x_3) + \cdots + f(x_{n-1}))]$$

або коротше

$$S_n = \frac{h}{3}\left[f(x_0) + f(x_n) + 2\sum_{k=1}^{\frac{n}{2}-1} f(x_{2k}) + 4\sum_{k=0}^{\frac{n}{2}-1} f(x_{2k+1}) \right]. \qquad (9.4)$$

Як видно з формули (9.4), n має бути обов'язково парним.

Оцінка погрішності формули Сімпсона [10]:

$$|I - S_n| \le \frac{b-a}{180} M_4 h^4, \qquad (9.5)$$

де $M_4 = \max\limits_{\xi \in (a,b)} \left| f^{(4)}(\xi) \right|$.

З оцінки (9.5) виходить що, якщо функція $f(x)$ є багаточленом 3-ї степені, то формула (9.4) для обчислення інтеграла (9.1) є теоретично точною.

Приклад 9.2. Обчислити значення визначеного інтеграла $\int\limits_3^{15} \left(5x - \frac{4}{(2x-4)^3} \right) dx$ за формулою Сімпсона для числа розбиття інтервала інтегрування $n = 50$.

Розв'язання в математичному пакеті R

Відправними даними задачі є підінтегральна функція $F(x)$, межі інтегрування a і b, число розбиття n інтервала $[a,b]$:

```
F = function(x)
{
  return (5*x-4/((2*x-4)^3))
}
a = 3
b = 15
n = 50
```

Процедура обчислення визначеного інтеграла за формулою Сімпсона може бути записана так:

```
Simpson = function(f, a, b, n)
{
     h = (b-a)/n
     s1 = 0
     s2 = 0
     x = c(1:(n-1))
     for (k in 1:(n/2-1))
     {
          x[2*k] = a + (2*k)*h
          s1 = s1 + f(x[2*k])
     }
     for (k in 1:(n/2))
     {
          x[2*k-1] = a + (2*k-1)*h
          s2 = s2 + f(x[2*k-1])
     }
     return (h/3*(f(a)+f(b)+2*s1+4*s2))
}
```

Результат обчислення заданого визначеного інтеграла з використанням записаної процедури:

```
> S = Simpson(F, a, b, n) # виклик функції користувача
> S
[1] 539.751
```

Як видно з отриманих результатів у прикладах 9.1 і 9.2, формула Сімпсона є точнішою, тому на практиці краще застосовувати саме її.

Слід зазначити, що якщо необхідно чисельно обчислити значення інтеграла (9.1) із заданою точністю, то для цього потрібно якимось чином визначити відповідне значення n. Можна було б скористатися оцінками (9.3) або (9.5), але для цього потрібно оцінити максимальне значення модуля 2-ї (для формули трапецій) або 4-ї (для формули Сімпсона) похідної на відрізку $[a,b]$, що може виявитися достатньо важким або зовсім неможливим. Тому можна скористатися таким алгоритмом:

1. Задається початкове значення n і обчислюється значення інтеграла $S1$ для заданого n.

2. Збільшується значення n вдвічі й обчислюється значення інтеграла $S2$ для цього n.

3. Знаходиться значення $|S2 - S1|$ і порівнюється його з заданою точністю обчислення ε.

4. Кроки 2 – 3 повторюються доти, доки не виконається умова $|S2 - S1| \le \varepsilon$.

Для обчислення значення визначеного інтеграла з заданою точністю можна скористатися такою процедурою:

```
IntegralTochnist = function(f, a, b, n0, eps)
{
     S1 = Simpson (f, a, b, n)
     S2 = Simpson (f, a, b, 2*n)
     while (abs(S2-S1)>eps)
     {
          n = 2*n
          S1 = S2
          S2 = Simpson (f, a, b, 2*n)
     }
     return (S2)
}
```

Результат обчислення визначеного інтеграла з використанням наведеної процедури:

```
> n0=10
> eps = 10^(-5)
> SToch = IntegralTochnist(F, a, b, n0, eps)
> SToch
[1] 539.7515
```

9.4. Висновки

1. Точність обчислення визначеного інтеграла залежить від числа розбиття відрізка інтегрування.

2. Метод Сімпсона є більш ефективним для розрахунків визначеного інтеграла.

9.5. Контрольні запитання та завдання

1. Сформулюйте постановку задачі чисельного інтегрування функції.

2. У чому полягає ідея методу трапецій?

3. У чому полягає ідея методу Сімпсона?

4. Який з відомих вам методів чисельного інтегрування має більшу точність?

5. Вкажіть алгоритм, за яким можна розрахувати значення визначеного інтеграла з заданою точністю.

6. Обчисліть визначений інтеграл функції функції $f(x)=\dfrac{5-x}{x^2+2}$ на відрізку $[-2,\ 3]$ за формулами трапецій і Сімпсона. Порівняйте результати.

10. Розв'язання задачі Коші для звичайних диференціальних рівнянь

10.1. Постановка задачі Коші

Задача Коші для звичайних диференціальних рівнянь використовується як математична модель при розв'язанні багатьох задач природознавства. Наприклад, задачі динаміки системи взаємодіючих тіл (у моделі руху матеріальних точок), задачі хімічної кінетики, електричних ланцюгів. Ряд важливих рівнянь у частинних похідних у випадках, що допускають розділення змінних, приводить до задач для звичайних диференціальних рівнянь. Це, як правило, крайові задачі (задачі про власні коливання пружних балок і пластин, визначення спектра власних значень енергії частинки у сферично-симетричних полях і багато інших).

Задача Коші для диференціального рівняння n-го порядку

$$y^{(n)}=f(x,y,y',y'',\ldots,y^{(n-1)}) \tag{10.1}$$

полягає у відшуканні функції $y=y(x)$, що задовольняє це рівняння і початкові умови:

$$y(x_0)=y_0,\ y'(x_0)=y'_0,\ \ldots,\ y^{(n-1)}(x_0)=y_0^{(n-1)}, \tag{10.2}$$

де $x_0, y_0, y'_0, y'', \ldots, y_0^{(n-1)}$ – задані числа.

Наприклад, можна розглянути 2-й закон Ньютона: в інерціальній системі відліку прискорення, яке отримує матеріальна точка з постійною масою, прямо пропорційно рівнодіючій всіх доданих до неї сил і обернено пропорційно її масі. Цей закон може бути записаний у вигляді формули:

$$a = \frac{F}{m},$$

де a – прискорення матеріальної точки;

F – рівнодіюча всіх сил;

m – маса матеріальної точки.

Якщо припустити, що напрям сили не змінюється і позначити через y координату точки на прямій вздовж руху, то $a = y''$ і формула 2-го закону Ньютона приймає вигляд $a'' = \frac{F}{m}$, тобто є окремим випадком формули (10.1). Якщо відомі положення точки та її швидкість у початковий момент часу (10.2) і рівнодіюча сила F, то розв'язавши задачу Коші можна визначти її положення і швидкість у будь-який наступний момент часу.

Задача Коші для системи диференціальних рівнянь

$$\begin{cases} \frac{dy_1}{dx} = f_1(x, y_1, y_2, ..., y_m), \\ \frac{dy_2}{dx} = f_2(x, y_1, y_2, ..., y_m), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \frac{dy_m}{dx} = f_m(x, y_1, y_2, ..., y_m) \end{cases} \qquad (10.3)$$

полягає у відшуканні функцій $y_1 = y_1(x)$, $y_2 = y_2(x)$, .., $y_m = y_m(x)$, що задовольняють цю систему і початкові умови:

$$y_j(x_0) = y_{j0},\ j = \overline{1,m}, \qquad (10.4)$$

де x_0, y_{j0}, $j = \overline{1,m}$, – задані числа.

Систему, що містить похідні вищих порядків і розв'язану відносно старших похідних шуканих функцій, шляхом введення нових невідомих функцій можна привести до вигляду (10.3). Зокрема, диференціальне рівняння n-го порядку (10.1) приводиться до вигляду (10.3) за допомогою такої заміни:

$$y_1 = y,\ y_2 = y',\ y_3 = y'',\ \ldots,\ y_n = y^{(n-1)},$$

що дає систему

$$\begin{cases} \dfrac{dy_1}{dx} = y_2, \\ \dfrac{dy_2}{dx} = y_3, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ \dfrac{dy_n}{dx} = f(x, y_1, y_2, \ldots, y_n). \end{cases} . \qquad (10.5)$$

Таким чином, задача (10.1), (10.2) є окремим випадком задачі (10.3), (10.4), тому чисельні методи розв'язання задачі Коші розроблені для більш загальної задачі (10.3), (10.4).

Варто зазначити, що в більшості практичних задач змінна x – це час, тобто в (10.3), (10.4) замість x можна застосовувати і позначення t.

Для задачі (10.3), (10.4) вводяться позначення:

$$Y = \begin{pmatrix} y_1(x) \\ y_2(x) \\ \ldots \\ y_m(x) \end{pmatrix},\ Y_0 = \begin{pmatrix} y_{10} \\ y_{20} \\ \ldots \\ y_{m0} \end{pmatrix},\ Y' = \begin{pmatrix} y_1'(x) \\ y_2'(x) \\ \ldots \\ y_m'(x) \end{pmatrix},\ F(x,Y) = \begin{pmatrix} f_1(x, y_1, y_2, \ldots, y_m) \\ f_2(x, y_1, y_2, \ldots, y_m) \\ \ldots \\ f_m(x, y_1, y_2, \ldots, y_m) \end{pmatrix},$$

де Y – шуканий розв'язок;

Y_0 – вектор початкових умов;

$F(x,Y)$ – вектор правих частин системи (10.3).

Тоді задача Коші для системи диференціальних рівнянь у векторній формі має вигляд:

$$Y' = F(x,Y),\ Y(x_0) = Y_0. \qquad (10.6)$$

Надалі для простоти викладення буде розглянуто задачу Коші для одного звичайного диференціального рівняння виду:

$$y' = f(x, y), \tag{10.7}$$

де y – скалярна змінна.

При цьому **задача Коші** полягає в такому: знайти функцію $y = y(x)$ на заданому відрізку $[a,b]$, що задовольняє рівнянню (10.7) і початковій умові:

$$y(a) = y_0, \tag{10.8}$$

де y_0 задане.

Чисельні методи, що будуть розглянуті далі для задачі (10.7), (10.8), мають місце і для загальної задачі (10.6).

Чисельні методи розв'язання задачі (10.7), (10.8) знаходять розв'язок (тобто функцію $y(x)$ на відрізку $[a,b]$) у табличному вигляді, а саме у вигляді набору точок (x_i, y_i), $i = \overline{0,n}$, де $x_0 = a$, $x_i = x_0 + i\,h$, $i = \overline{1,n}$, $h = \dfrac{b-a}{n}$, n – задане число розбиття відрізка $[a,b]$, y_i, $i = \overline{1,n}$ – знайдені наближені значення функції $y(x)$ в точках x_i (слід нагадати, що y_0 задане спочатку).

Варто розглянути декілька методів розв'язання задачі Коші для диференціального рівняння (10.7) на відрізку $[a,b]$.

10.2. Метод Ейлера та його модифікації

Ідея методу Ейлера заснована на тому, що шукана інтегральна крива $y = y(x)$, яка проходить через точку $M_0(x_0, y_0)$, відновлюється у вигляді кусочно-лінійної ламаної $M_0M_1M_2...M_n$ з вершинами $M_i(x_i, y_i)$ ($i = \overline{0,n}$) (рис. 10.1). Кожен відрізок M_iM_{i+1} цієї ламаної має напрям, що співпадає з напрямом тієї інтегральної кривої рівняння (10.7), яка проходить через точку M_i.

Тоді $y_{i+1} - y_i = h \times tg(\alpha)$. Тангенс кута нахилу дотичної до $y(x)$ в точці x_i дорівнює $y'(x_i)$, а значить, згідно з (10.7), і дорівнює $f(x_i, y_i)$, звідки і випливає формула методу Ейлера.

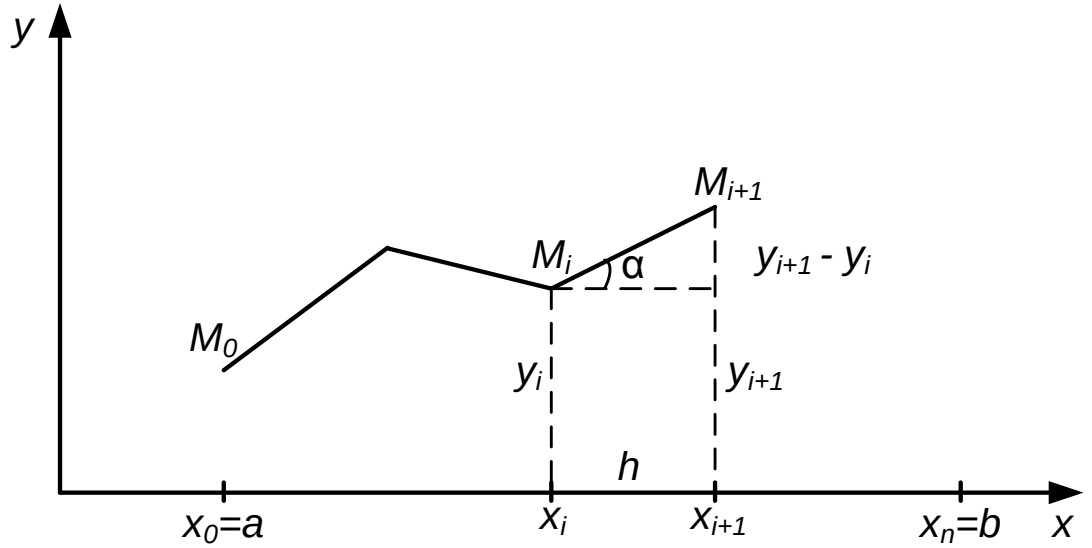


Рис. 10.1. **Графічна інтерпретація методу Ейлера**

Таким чином, у методі Ейлера значення y_i обчислюються рекурентно за формулою:

$$y_{i+1} = y_i + h \times f(x_i, y_i),\ i = \overline{0, n-1}. \qquad (10.9)$$

Погрішність методу Ейлера [3; 5; 6; 19], що оцінюється для величини $|y_n - y(x_n)|$, має порядок $O(h)$, тобто існує деяка константа $M > 0$ така, що:

$$|y_n - y(x_n)| \le M h. \qquad (10.10)$$

Тут $y(x)$ – точний розв'язок задачі Коші (10.7), (10.8), а порівняння з наближеним розв'язком (x_i, y_i), $i = \overline{0, n}$, йде в крайній (правій) точці $x_n = b$ відрізка $[a, b]$, оскільки саме в ній погрішність теоретично буде максимальною, відносно проміжних точок x_i.

Варто зазначити, що оцінка (10.10) носить лише теоретичний характер, на практиці ж для оцінки отриманого розв'язку можна скористатися подвійним прорахунком. Для цього проводять повторні

обчислення, але вже з кроком $\frac{h}{2}$, отримують вже точніший наближений розв'язок $\left(x_i^*, y_i^*\right)$, $i = \overline{0, n^*}$, $(n^* = 2n)$ і погрішність наближення оцінюють як:

$$\left|y_{n^*}^* - y(x_{n^*})\right| \approx \left|y_{n^*}^* - y_n\right|.$$

Таким чином, для того щоб отримати розв'язок задачі Коші (10.7), (10.8) із заданою точністю $\varepsilon > 0$ (застосовуючи метод Ейлера) можна, починаючи з деякого початкового значення кроку h, проводити подвійний прорахунок (тобто зменшуючи вдвічі h) допоки не виконається нерівність $\left|y_{n^*}^* - y_n\right| \le \varepsilon$.

Приклад 10.1. Розв'язати методом Ейлера задачу Коші для диференціального рівняння $y' = x + y$ на відрізку $[0, 5]$, якщо $y(0) = 1$.

Розв'язання в математичному пакеті R

Відправними даними задачі є диференціальне рівняння $F(x, y)$, межі відрізка A і B, початкове значення $Y0$, число розбиття n відрізка $[a,b]$:

```
# задаємо рівняння (функцію правої частини)
FunPr = function(x,y)
{
  return (x + y)
}
a = 0
b = 5
y0 = 1
n = 20
```

Процедура обчислення для розв'язання задачі Коші для диференціального рівняння методом Ейлера може бути записана так:

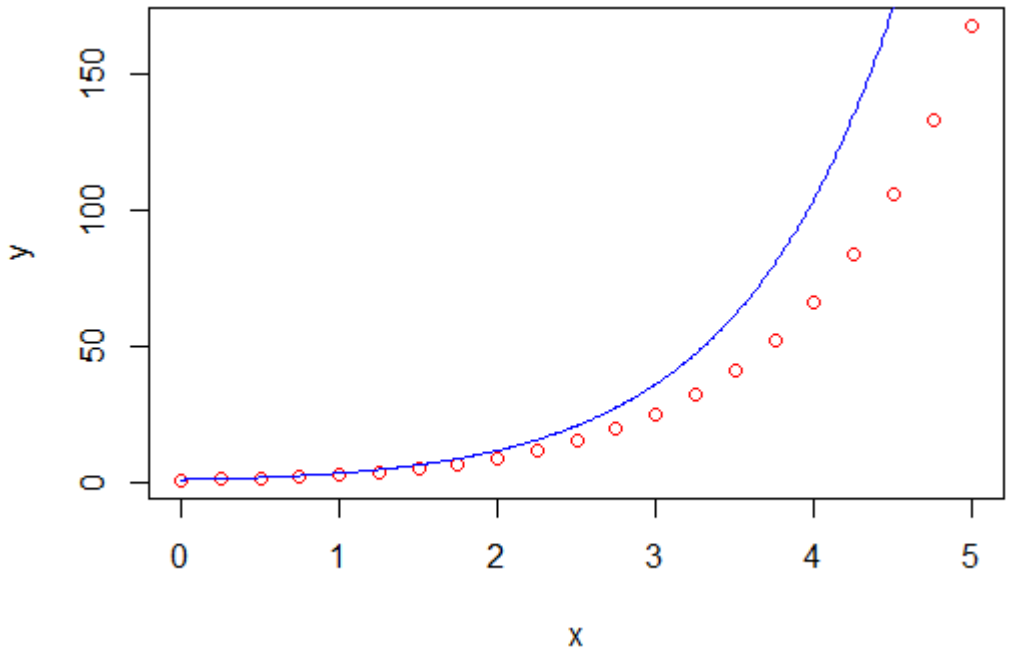
```
MetEiler = function(f, a, b, y0, n)
{
  h = (b-a)/n
  x = c(1:(n+1))
  y = c(1:(n+1))
  x[1] = a
  y[1] = y0
  for (i in 1:n)
  {
    x[i+1] = x[i] + h
    y[i+1] = y[i] + h*f(x[i],y[i])
  }
  return(cbind(x,y))
}
```

Результат обчислення з використанням записаної процедури:

```
> xy = MetEiler(FunPr, a, b, y0, n)
> xy
          x           y
 [1,] 0.00    1.000000
 [2,] 0.25    1.250000
 [3,] 0.50    1.625000
 [4,] 0.75    2.156250
 [5,] 1.00    2.882812
 [6,] 1.25    3.853516
 [7,] 1.50    5.129395
 [8,] 1.75    6.786743
 [9,] 2.00    8.920929
[10,] 2.25   11.651161

[11,] 2.50   15.126451
[12,] 2.75   19.533064
[13,] 3.00   25.103830
[14,] 3.25   32.129788
[15,] 3.50   40.974735
[16,] 3.75   52.093419
[17,] 4.00   66.054274
[18,] 4.25   83.567842
[19,] 4.50  105.522302
[20,] 4.75  133.027878
[21,] 5.00  167.472348
```

Аналітичним розв'язком даного диференціального рівняння є функція $YT(x) = 2e^x - x - 1$. Порівняти результати розв'язку, отримані чисельно методом Ейлера й аналітично, можна побудувавши графік:

```
x = xy[, 1]
y = xy[, 2]

# Будуємо графік чисельного розв'язку
plot(x, y, type = "p", col="red")
# Аналітичний розв'язок задачі
FunAn = function(x)
{
  return (2*exp(x) - x - 1)
}
x1 = seq(x[1], x[n+1], by=0.01)
y1 = FunAn(x1)
# додаємо в 1-у графічну підобласть ще лінію
lines(x1, y1, col="blue")
```



Як видно з наведеної оцінки погрішності (10.10), метод Ейлера дає невисоку точність при розв'язанні задачі Коші (10.7), (10.8), тому його рідко застосовують на практиці. Слід розглянути дві його модифікації.

1-а модифікація методу Ейлера. У 1-й модифікації методу Ейлера значення y_i обчислюються рекурентно за формулами ($i = \overline{0,n-1}$):

$$y_{i+\frac{1}{2}} = y_i + \frac{h}{2} f(x_i, y_i); \tag{10.11}$$

$$y_{i+1} = y_i + h \times f(x_i + \frac{h}{2}, y_{i+\frac{1}{2}}). \tag{10.12}$$

Ідея цієї модифікації методу Ейлера полягає в тому, що спочатку обчислюється значення $y_{i+\frac{1}{2}}$ за формулою (10.11) у проміжній точці $x_{i+\frac{1}{2}} = x_i + \frac{h}{2}$ (середині) інтервала (x_i, x_{i+1}), а потім вже отримують y_{i+1} в точці x_{i+1} за формулою (10.12).

Погрішність 1-ї модифікації методу Ейлера [3; 5; 6; 19], що оцінюється для величини $|y_n - y(x_n)|$, має порядок $O(h^3)$, тобто існує деяка константа $M > 0$ така, що $|y_n - y(x_n)| \le Mh^3$.

Як видно з наведеної оцінки погрішність 1-ї модифікації метода Ейлера на два порядки менша, ніж у звичайного методу Ейлера, хоча за це доводиться платити додатковими обчисленнями. А саме: функція $f(x,y)$ з правої частини диференціального рівняння (10.7) обчислюється на кожному кроці 1-ї модифікації (10.11), (10.12) двічі. Варто зазначити, що при розв'язанні практичних задач основні обчислювальні витрати припадатимуть саме на обчислення значень функції $f(x,y)$.

Приклад 10.2. Розв'язати задачу з прикладу 10.1, застосувавши 1-у модифікацію методу Ейлера.

Розв'язання в математичному пакеті R

Відправні дані – ті ж самі, що й у прикладі 10.1.

Процедура обчислення для розв'язання задачі Коші для диференціального рівняння з застосуванням 1-ї модифікації методу Ейлера може бути записана так:

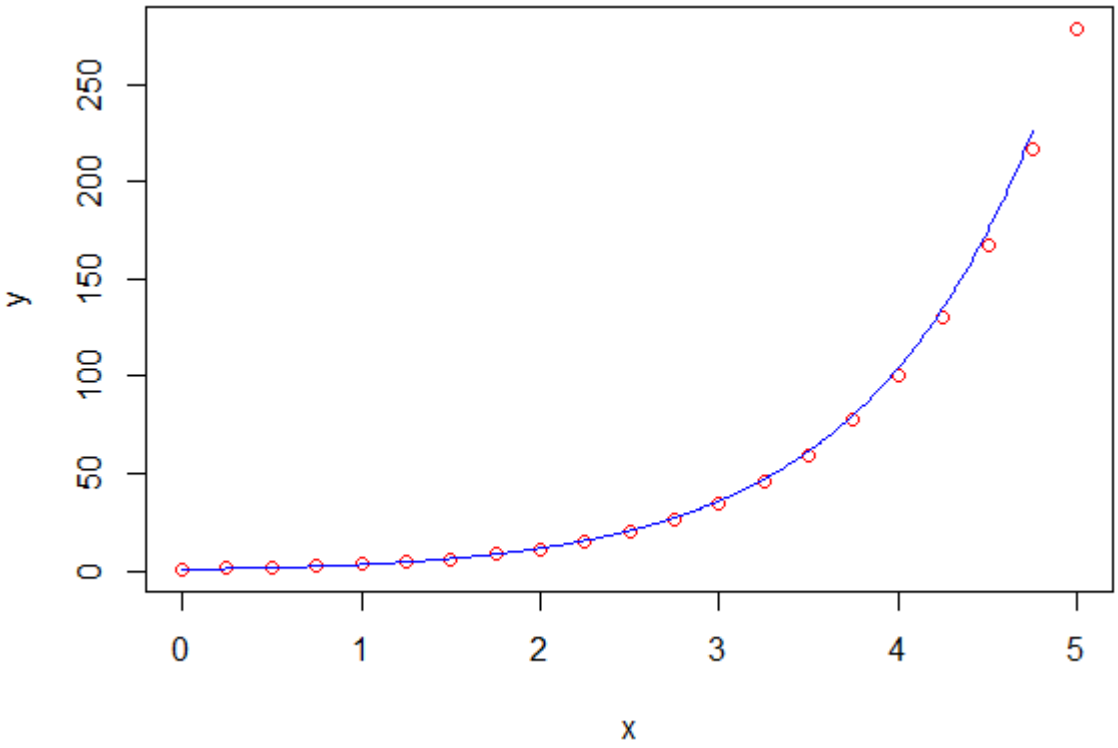
```
MetEiler1 = function(f, a, b, y0, n)
{
  h = (b-a)/n
  x = c(1:(n+1))
  y = c(1:(n+1))
  x[1] = a
  y[1] = y0
  for (i in 1:n)
  {
    x[i+1] = x[i] + h
    ypr = y[i] + h/2*f(x[i],y[i])
    y[i+1] = y[i] + h*f(x[i]+h/2,ypr)
  }
  return(cbind(x,y))
}
```

Результат обчислення з використанням записаної процедури:

```
> xy = MetEiler1(FunPr, a, b, y0, n)
> xy
          x           y
 [1,] 0.00    1.000000
 [2,] 0.25    1.312500
 [3,] 0.50    1.783203
 [4,] 0.75    2.456604
 [5,] 1.00    3.389711
 [6,] 1.25    4.655568
 [7,] 1.50    6.347759
 [8,] 1.75    8.586191
 [9,] 2.00   11.524494
[10,] 2.25   15.359508
[11,] 2.50   20.343433
[12,] 2.75   26.799398
[13,] 3.00   35.141416
[14,] 3.25   45.899940
[15,] 3.50   59.754610
[16,] 3.75   77.576219
[17,] 4.00  100.480468
[18,] 4.25  129.896850
[19,] 4.50  167.656902
[20,] 4.75  216.107281
[21,] 5.00  278.254641
```

Порівняти результати розв'язку, отримані чисельно з застосуванням 1-ї модифікації методу Ейлера й аналітично, можна побудувавши графік:

```
x = xy[, 1]
y = xy[, 2]
# Будуємо графік чисельного розв'язоку
plot(x, y, type = "p", col="red")
# Аналітичний розв'язок задачі
FunAn = function(x)
{
  return (2*exp(x) - x - 1)
}
x1 = seq(x[1], x[n], by=0.01)
y1 = FunAn(x1)
# додаємо в 1-у графічну підобласть ще лінію
lines(x1, y1, col="blue")
```



Як видно з цього графіка, 1-а модифікація методу Ейлера дає більш точний роз'язок задачі, ніж класичний метод Ейлера.

2-а модифікація методу Ейлера. У 2-й модифікації методу Ейлера значення y_i обчислюються рекурентно за формулами ($i = \overline{0,n-1}$):

$$y^*_{i+1} = y_i + h \times f(x_i, y_i); \tag{10.13}$$

$$y_{i+1} = y_i + \frac{h}{2}(f(x_i, y_i) + f(x_{i+1}, y^*_{i+1})). \tag{10.14}$$

Ідея цієї модифікації методу Ейлера полягає в тому, що спочатку обчислюється "грубе наближення" y^*_{i+1} за формулою (10.13) і за ним обчислюють значення $f(x_{i+1}, y^*_{i+1})$, а потім вже "остаточне наближення" y_{i+1} отримують за формулою (10.14), в якому фігурує середнє значення $\frac{1}{2}(f(x_i, y_i) + f(x_{i+1}, y^*_{i+1}))$.

Погрішність 2-ї модифікації методу Ейлера, що оцінюється для величини $|y_n - y(x_n)|$, також має порядок $O(h^3)$, тобто існує деяка

константа $M > 0$ така, що $|y_n - y(x_n)| \le Mh^3$. Значення функції $f(x,y)$ з правої частини диференціального рівняння (10.7) також обчислюються двічі на кожному кроці цієї модифікації (10.13), (10.14).

10.3. Метод Рунге – Кутта четвертого порядку

Ідея методу Рунге – Кутта заснована на застосуванні, на відміну від методу Ейлера (кусочно-лінійна ламана), кривих вищого порядку для відновлення значень шуканої функції $y(x)$ на відрізку $[a,b]$.

Найчастіше при розв'язанні практичних задач (10.7), (10.8) використовується **метод Рунге – Кутта четвертого порядку**. Згідно з цим методом значення y_i обчислюються рекурентно за формулою

$$y_{i+1} = y_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),\ i = \overline{0,n-1}, \qquad (10.15)$$

де $k_1 = hf(x_i, y_i)$, $k_2 = hf(x_i + \frac{h}{2}, y_i + \frac{k_1}{2})$, $k_3 = hf(x_i + \frac{h}{2}, y_i + \frac{k_2}{2})$, $k_4 = hf(x_i + h, y_i + k_3)$.

Погрішність методу Рунге – Кутта 4-го порядку, що оцінюється для величини $|y_n - y(x_n)|$, має порядок $O(h^4)$. Як видно, погрішність при використанні цього методу нижча, ніж при використанні методу Ейлера і його модифікацій, але за це доводиться платити додатковими обчисленнями. А саме: функція $f(x,y)$ з правої частини диференціального рівняння (10.7) обчислюється на кожному кроці методу Рунге – Кутта 4-го порядку (10.15) чотири рази, що у випадках, коли обчислення значення функції $f(x,y)$ є дуже трудомістким, може істотно уповільнити пошук розв'язку.

Приклад 10.3. Розв'язати задачу з прикладу 10.1 методом Рунге – Кутта 4-го порядку.

Розв'язання в математичному пакеті R

Відправні дані – ті ж самі, що й у прикладі 10.1.

Процедура обчислення для розв'язання задачі Коші для диференціального рівняння методом Рунге – Кутта 4-го порядку може бути записана так:

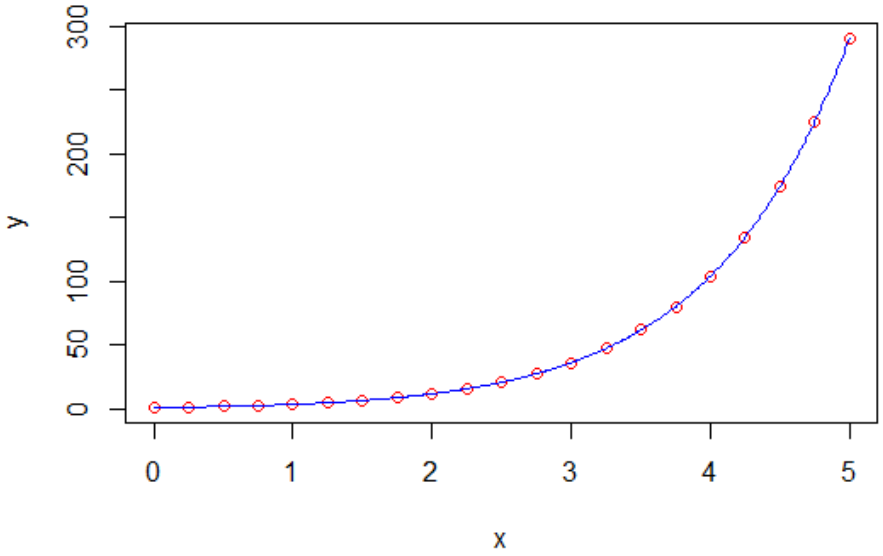
```
MetRungKut = function(f, a, b, y0, n)
{
  h = (b-a)/n
  x = c(1:(n+1))
  y = c(1:(n+1))
  x[1] = a
  y[1] = y0
  for (i in 1:n)
  {
    x[i+1] = x[i] + h
    k1 =  h*f(x[i],y[i])
    k2 =  h*f(x[i]+h/2,y[i]+k1/2)
    k3 =  h*f(x[i]+h/2,y[i]+k2/2)
    k4 =  h*f(x[i]+h,y[i]+k3)
    y[i+1] = y[i] + 1/6*(k1 + 2*k2 + 2*k3 + k4)
  }
  return(cbind(x,y))
}
```

Результат обчислення з використанням записаної процедури:

```
> xy = MetRungKut(FunPr, a, b, y0, n)
> xy
          x           y
 [1,] 0.00    1.000000
 [2,] 0.25    1.318034
 [3,] 0.50    1.797399
 [4,] 0.75    2.483916
 [5,] 1.00    3.436420

 [6,] 1.25    4.730455
 [7,] 1.50    6.463023
 [8,] 1.75    8.758673
 [9,] 2.00   11.777331
[10,] 2.25   15.724343
[11,] 2.50   20.863377
[12,] 2.75   27.532989
[13,] 3.00   36.167887
[14,] 3.25   47.326247
[15,] 3.50   61.724774
[16,] 3.75   80.283730
[17,] 4.00  104.184749
[18,] 4.25  134.945066
[19,] 4.50  174.512838
[20,] 4.75  225.389531
[21,] 5.00  290.787070
```

Порівняти результати розв'язку, отримані чисельно методом Рунге – Кутта 4-го порядку й аналітично, можна побудувавши графік:

```
> x = xy[, 1]
> y = xy[, 2]
> # Будуємо графік чисельного розв'язку
> plot(x, y, type = "p", col="red")
> # Аналітичний розв'язок задачі
> FunAn = function(x)
+ {
+   return (2*exp(x) - x - 1)
+ }
> x1 = seq(x[1], x[n+1], by=0.01)
> y1 = FunAn(x1)
> # додаємо в 1-у графічну підобласть ще лінію
> lines(x1, y1, col="blue")
```



Порівняння графіків розв'язків, отриманих чисельно в прикладах 10.1 – 10.3, з аналітичним розв'язком ще раз доводить, що з розглянутих методів розв'язання задачі Коші для диференціального рівняння найточнішим є метод методом Рунге – Кутта 4-го порядку (чисельно отриманий розв'язок збігається з аналітичним), а найгіршим – метод Ейлера.

10.4. Висновки

1. Математичні моделі процесів та явищ, зокрема моделі динамічних систем, у більшості випадків записуються у вигляді диференціальних рівнянь.

2. Задача Коші для звичайних диференціальних рівнянь має велике практичне значення.

10.5. Контрольні запитання та завдання

1. Що називається звичайним диференціальним рівнянням?
2. Сформулюйте постановку задачі Коші для звичайного диференціального рівняння. Що є її розв'язком? У якому вигляді подається розв'язок чисельним методом?
3. Який з відомих вам методів розв'язання задачі Коші для звичайного диференціального рівняння треба застосовувати в тих чи інших випадках?
4. Розв'яжіть задачу Коші для диференціального рівняння $y' = \frac{x + y - 3}{x - y - 1}$ на відрізку $[2, 25]$ (початкова умова $y(2) = 1$) чисельно методами Ейлера і його модифікаціями й методом Рунге – Кутта із числом розбиття відрізка $n = 20$. Побудуйте графіки отриманих розв'язків.

11. Багатокрокові методи розв'язання звичайних диференціальних рівнянь

11.1. Поняття багатокрокового методу

Розглянуті у розділі 10 чисельні методи розв'язання задачі Коші для звичайних диференціальних рівнянь відносяться до **однокрокових методів** тому, що при розрахунку поточного значення y_{i+1} на i-му кроці використовується тільки інформація на останньому відрізку $[x_i, x_{i+1}]$. Все, що робилось на попередніх кроках методу, явно не використовується. Навпаки, є **багатокрокові методи** розв'язання задачі Коші (10.7), які використовують те, що було отримано на попередніх кроках методу явно. Такими є, наприклад, методи: Адамса – Бошфорда, Адамса – Мултона, Гіра – Брайтона [21].

11.2. Метод Адамса – Бошфорда

В основі методу Адамса – Бошфорда лежить формула Ньютона – Лейбніца:

$$\int_a^b f(x)dx = \Phi(b) - \Phi(a),$$

де $\Phi(x)$ – будь-яка первісна для підінтегральної функції $f(x)$ на відрізку $[a, b]$.

Застосування формули Ньютона – Лейбніца до рівняння (10.7) на будь-якому відрізку $[x_i, x_{i+1}]$ приводить до рівняння:

$$y(x_{i+1}) - y(x_i) = \int_{x_i}^{x_{i+1}} f(x, y(x))dx .$$

Тоді, якщо вже були отримані значення $y_{i-m+1}, y_{i-m+2}, \ldots, y_{i-1}, y_i$ в m попередніх точках (вузлах) $x_{i-m+1}, x_{i-m+2}, \ldots, x_{i-1}, x_i$, то підінтегральну функцію $f(x, y(x))$ можна замінити інтерполяційним поліномом Ньютона $H_{m-1}(x)$ (див. розділ 7.5), побудованим за цими m вузлами. Таким чином, отримуємо загальну рекурентну формулу методу Адамса – Бошфорда m-го порядку:

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} H_{m-1}(x)dx . \qquad (11.1)$$

Оскільки $H_{m-1}(x)$ – поліном, то інтеграл від нього береться аналітично і таким чином отримують різницеву схему розв'язання задачі Коші для будь-якого m. Наприклад, якщо $m = 1$, то $H_{m-1}(x) = f(x_i, y_i)$ і з (11.1) витікає:

$$y_{i+1} = y_i + h\, f(x_i, y_i),$$

де $h = x_{i+1} - x_i$.

Таким чином, при $m = 1$ метод Адамса – Бошфорда співпадає з методом Ейлера (див. розділ 10.2).

Найчастіше застосовують метод Адамса – Бошфорда 4-го порядку, який має таку різницеву схему [21]:

$$y_{i+1} = y_i + \frac{h}{24}(55f_i - 59f_{i-1} + 37f_{i-2} - 9f_{i-3}), \qquad (11.2)$$

де $f_i = f(x_i, y_i)$.

Очевидно, що за формулою (11.2) можна проводити обчислення тільки при $i \geq 3$, тому що на i-му кроці методу треба знати значення $f_{i-3}, f_{i-2}, f_{i-1}, f_i$. Тому перш ніж застосовувати метод Адамса – Бошфорда 4-го порядку, необхідно спочатку обчислити значення f_0, f_1, f_2

для перших 3-х кроків. Зазвичай для цього застосовують метод Рунге – Кутта, бо він має достатньо велику точність.

Погрішність методу Адамса – Бошфорда 4-го порядку, що оцінюється для величини $|y_n - y(x_n)|$, має порядок $O(h^4)$ [21].

Приклад 11.1. Розв'язати задачу з прикладу 10.1 методом Адамса – Бошфорда 4-го порядку.

Розв'язання в математичному пакеті R

Відправні дані – ті ж самі, що й у прикладі 10.1.

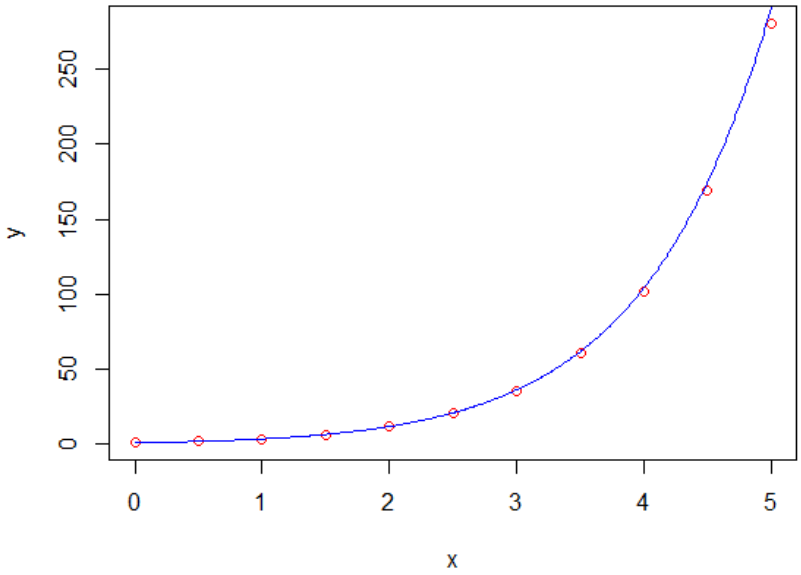
```
# задаємо рівняння (функцію правої частини)
FunPr = function(x,y)
{
  return (x + y)
}
a = 0
b = 5
y0 = 1
```

Процедура обчислення для розв'язання задачі Коші для диференціального рівняння методом Адамса – Бошфорда 4-го порядку може бути записана так:

```
MetAdamBosh4 = function(fun, a, b, y0, n)
{
  h = (b-a)/n
  x = c(1:(n+1))
  y = c(1:(n+1))
  f = c(1:(n+1))
  x[1] = a
  y[1] = y0
  f[1] =  fun(x[1], y[1])
  for (i in 1:3)
  {
    x[i+1] = x[i] + h
    k1 =  h*f[i]
    k2 =  h*fun(x[i]+h/2, y[i]+k1/2)
    k3 =  h*fun(x[i]+h/2, y[i]+k2/2)
    k4 =  h*fun(x[i]+h, y[i]+k3)
    y[i+1] = y[i] + 1/6*(k1 + 2*k2 + 2*k3 + k4)
    f[i+1] =  fun(x[i+1], y[i+1])
  }
  for (i in 4:n)
  {
    x[i+1] = x[i] + h
    y[i+1] = y[i] + h/24*(55*f[i] - 59*f[i-1] +
               + 37*f[i-2] - 9*f[i-3])
    f[i+1] =  fun(x[i+1], y[i+1])
  }
  return(cbind(x,y))
}
```

Результат обчислення з використанням записаної процедури та порівняння отриманого й аналітичного розв'язку:

```
> # Розв'язуємо систему диференційних рівнянь
> n = 10
> xy = MetAdamBosh4(FunPr, a, b, y0, n)
> x = xy[, 1]
> y = xy[, 2]
> # Будуємо графік чисельного розв'язоку
> plot(x, y, type = "p", col="red")
> # Аналітичний розв'язок задачі
> FunAn = function(x)
+ {
+   return (2*exp(x) - x - 1)
+ }
> x1 = seq(x[1], x[n+1], by=0.01)
> y1 = FunAn(x1)
> # додаємо в 1-у графічну підобласть ще лінію
> lines(x1, y1, col="blue")
```



Порівняння графіків розв'язків, отриманих чисельно в прикладах 10.3 і 11.1, з аналітичним розв'язком ще раз доводить, що з розглянутих методів розв'язання задачі Коші для диференціального рівняння метод Рунге – Кутта 4-го порядку і метод Адамса – Бошфорда 4-го порядку дають майже однакові результати.

11.3. Метод Адамса – Мултона

Слід зазначити, що в методі Адамса – Бошфорда інтерполяційний поліномом Ньютона $H_{m-1}(x)$ застосовується для екстраполяції функції $f(x, y(x))$ на відрізку $[x_i, x_{i+1}]$, оскільки він будується за вузлами $x_{i-m+1}, x_{i-m+2}, \ldots, x_{i-1}, x_i$. У методі Адамса – Мултона також застосо-

вується інтерполяційний поліномом Ньютона $H_{m-1}(x)$, але не для екстраполяції, а для інтерполяції функції $f(x,y(x))$ на відрізку $[x_i, x_{i+1}]$, оскільки він будується за вузлами $x_{i-m+2}, x_{i-m+3}, \ldots, x_i, x_{i+1}$.

Тому для методу Адамса – Мултона 4-го порядку отримано таку різницеву схему [21]:

$$y_{i+1} = y_i + \frac{h}{24}(9f_{i+1} + 19f_i - 5f_{i-1} + f_{i-2}), \qquad (11.3)$$

де $f_{i+1} = f(x_{i+1}, y_{i+1})$.

На відміну від схеми (11.2), в схемі (11.3) у правій частині фігурує ще не відоме значення y_{i+1}. Такі методи носять назву **неявних методів**. Тому для пошуку y_{i+1} треба розв'язати нелінійне рівняння (11.3) відносно y_{i+1}. Слід зазначити, що в загальній постановці задачі Коші виду (10.3), (10.4) при застосуванні методу Адамса – Мултона доведеться на кожному кроці розв'язувати систему нелінійних рівнянь.

Погрішність методу Адамса – Мултона 4-го порядку, що оцінюється для величини $|y_n - y(x_n)|$, має порядок $O(h^5)$ [21].

11.4. Метод прогнозу та корекції

Різницеву схему (11.3) застосовують і для уточнення значення y_{i+1}, що було розраховане за різницевою схемою (11.2). Така комбінація методів Адамса – Бошфорда і Адамса – Мултона 4-го порядку має назву **методу прогнозу та корекції** [21].

У багатокроковому **методі прогнозу та корекції** 4-го порядку у ході розв'язання задачі Коші (10.7), (10.8) значення y_i у вузлах сітки $x_i, i = \overline{1,n}$, обчислюються рекурентно за формулами:

$$y_{i+1}^{(pred)} = y_i + \frac{h}{24}(55f_i - 59f_{i-1} + 37f_{i-2} - 9f_{i-3}),$$

$$f_{i+1}^{(pred)} = f(x_{i+1}, y_{i+1}^{(pred)}),$$

$$y_{i+1} = y_i + \frac{h}{24}(9f_{i+1}^{(pred)} + 19f_i - 5f_{i-1} + f_{i-2}),$$

де $f_i = f(x_i, y_i)$.

Похибка методу прогнозу та корекції 4-го порядку, оцінювана для величини $|y_n - y(x_n)|$, має порядок $O(h^5)$ [6; 21].

Якщо порівнювати однокрокові та багатокрокові методи розв'язання задачі Коші для звичайних диференціальних рівнянь, треба підкреслити, що в багатокрокових методах крок h можна обирати більшим, ніж у однокрокових методах. Це дає можливість значно зменшити кількість кроків, а значить і трудомісткість розв'язання задачі в цілому.

Приклад 11.2. Розв'язати задачу з прикладу 10.1 методом прогнозу та корекції 4-го порядку.

Розв'язання в математичному пакеті R

Відправні дані – ті ж самі, що й у прикладі 10.1.

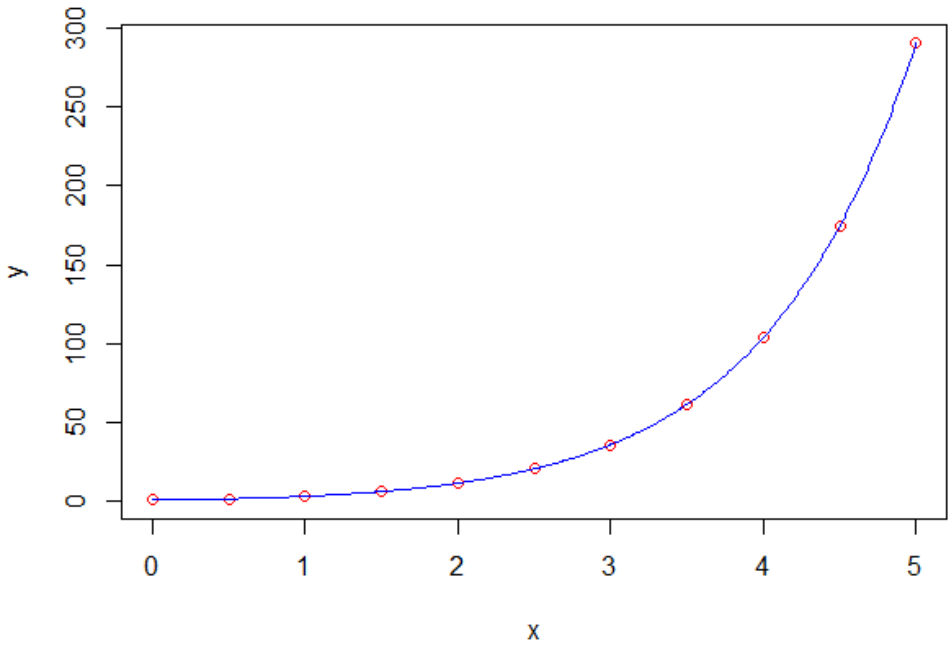
```
# задаємо рівняння (функцію правої частини)
FunPr = function(x,y)
{
  return (x + y)
}
a = 0
b = 5
y0 = 1
```

Процедура обчислення для розв'язання задачі Коші для диференціального рівняння методом прогнозу та корекції 4-го порядку може бути записана так:

```
MetPredCor4 = function(fun, a, b, y0, n)
{
     h = (b-a)/n
     x = c(1:(n+1))
     y = c(1:(n+1))
     f = c(1:(n+1))
     x[1] = a
     y[1] = y0
     f[1] = fun(x[1], y[1])
     for(i in 1:3)
     {
          x[i+1] = x[i] + h
          k1 = h*f[i]
          k2 = h*fun(x[i]+h/2, y[i]+k1/2)
          k3 = h*fun(x[i]+h/2, y[i]+k2/2)
          k4 = h*fun(x[i]+h, y[i]+k3)
          y[i+1] = y[i]+(1/6)*(k1+2*k2+2*k3+k4)
          f[i+1] = fun(x[i+1], y[i+1])
     }
     for(i in 4:n)
     {
          x[i+1] = x[i] + h
          ypred = y[i] + h/24*(55*f[i] - 59*f[i-1] + 37*f[i-2] - 9*f[i-3])
          fpred = fun(x[i+1], ypred)
          y[i+1] = y[i] + h/24*(9*fpred + 19*f[i]- 5*f[i-1] + f[i-2])
          f[i+1] = fun(x[i+1], y[i+1])
     }
     return(cbind(x, y))
}
```

Результат обчислення з використанням записаної процедури та порівняння отриманого й аналітичного розв'язку:

```
> n = 10
>
> # Розв'язуємо задачу Коші для диференціального рівняння
> xy = MetPredCor4(FunPr, a, b, y0, n)
> xy
         x           y
 [1,] 0.0    1.000000
 [2,] 0.5    1.796875
 [3,] 1.0    3.434692
 [4,] 1.5    6.458751
 [5,] 2.0   11.765654
 [6,] 2.5   20.836144
 [7,] 3.0   36.109983
 [8,] 3.5   61.607877
 [9,] 4.0  103.956680
[10,] 4.5  174.078571
[11,] 5.0  289.975092
> # Будуємо графік чисельного розв'язку
> x = xy[,1]
> y = xy[,2]
> plot(x, y, type = "p", col= "red")
>
> # Аналітичний розв'язок задачі
> FunAn = function(x)
+ {
+ return(2*exp(x) - x - 1)
+ }
> # Будуємо графік аналітичного розв'язку
> x1 = seq(a, b, by=0.01)
> y1 = FunAn(x1)
> # додаємо в 1-у графічну підобласть ще лінію
> lines(x1, y1, col="blue")
```



11.5. Висновки

1. При застосуванні багатокрокового методу необхідно спочатку декілька кроків виконати однокроковим методом.

2. У багатокрокових методах крок h можна обирати більшим, ніж у однокрокових методах.

11.6. Контрольні запитання та завдання

1. Які чисельні методи розв'язання задачі Коші для звичайних диференціальних рівнянь називаються багатокроковими методами?

2. Які особливості мають багатокрокові методи відносно одно-крокових методів?

3. Наведіть загальну схему всіх методів розв'язання задачі Коші для системи звичайних диференціальних рівнянь.

12. Неявні методи розв'язання жорстких задач Коші

12.1. Поняття жорсткої системи диференціальних рівнянь

Поняття жорсткої задачі Коші вводиться для задачі виду (10.3), (10.4) (в матричній формі (10.6)), тобто для системи диференціальних рівнянь.

Для підвищення точності та адекватності математичних моделей складних об'єктів і процесів при побудові моделей доводиться враховувати велику кількість факторів та параметрів. При цьому в математичній моделі, що описується системою диференціальних рівнянь, опиняються складові з великими і малими значеннями похідних від шуканих функцій $y_j(x)$. Це і призводить до так званої **жорсткої системи диференціальних рівнянь.** Тут треба зазначити, що жорсткість є властивістю самої математичної задачі, а не чисельного методу її розв'язання. Застосування описаних (розділи 10, 11) явних чисельних методів для жорстких систем диференціальних рівнянь дає велику похибку у розв'язку, тому для них розроблені так звані **неявні методи** [21].

Слід розглянути спочатку (замість (10.6)) лінійну систему диференціальних рівнянь з незалежною від x матрицею $A \in R^{m \times m}$:

$$Y' = A \times Y(x). \tag{12.1}$$

Нехай $\lambda_i, i = \overline{1,m}$ – множина власних чисел матриці A, $K = \dfrac{\max\limits_i \left|Re(\lambda_i)\right|}{\min\limits_i \left|Re(\lambda_i)\right|}$, де $Re(\lambda_i)$ – дійсна частина власного числа.

Визначення. Система диференціальних рівнянь (12.1) називається **жорсткою**, якщо [21]:

$$Re(\lambda_i) < 0, i = \overline{1,m} \text{ і } K >> 1.$$

При цьому K називається **числом жорсткості системи** (12.1).

Це поняття жорсткої системи узагальнюється і на систему (10.6). Роль матриці A при цьому відіграє матриця Якобі

$$A(x) = F'_Y(x,Y) = \begin{pmatrix} \dfrac{\partial f_1(x,Y)}{\partial y_1} & \cdots & \dfrac{\partial f_1(x,Y)}{\partial y_m} \\ \cdots & \cdots & \cdots \\ \dfrac{\partial f_n(x,Y)}{\partial y_1} & \cdots & \dfrac{\partial f_n(x,Y)}{\partial y_m} \end{pmatrix},$$

яка в загальному випадку вже буде залежати від x, тобто змінюватись у часі буде і число жорсткості системи (10.6).

Як вже згадувалося раніше, застосування явних чисельних методів для жорстких систем диференціальних рівнянь дає велику похибку у розв'язку, якщо крок обчислень не є досить малим, а саме він повинен задовольняти обмеженню [21]:

$$h < \frac{c}{|\lambda_{max}|},$$

де c – константа, яка залежить від умов задачі (10.6),

λ_{max} – максимальне за модулем власне число матриці Якобі.

12.2. Неявні методи Ейлера і Рунге – Кутта

Поняття неявного методу було введено в розділі 10.3. Це методи, що застосовують схему (наприклад, як в схемі (11.3) методу Адамса – Мултона 4-го порядку), в якій в лівій і правій частині фігурує ще не відоме значення Y_{i+1}.

Неявний метод Ейлера. Явна схема методу Ейлера має вид (10.9).

У розділі 10.2 було зазначено, що цей метод Ейлера співпадає з методом Адамса – Бошфорда 1-го порядку (див. розділ 10.2). В той же час, метод Адамса – Мултона дає алгоритм переходу від явного методу до неявного, шляхом заміни виду інтерполяційного поліному Ньютона. Так, при $m=1$, якщо побудувати інтерполяційний поліном Ньютона за вузлом x_{i+1}, то він буде мати вигляд: $H_{m-1}(x)=F(x_{i+1},Y_{i+1})$. Таким чином, з (11.1) отримується схема неявного методу Ейлера:

$$Y_{i+1}=Y_i+h\,F(x_{i+1},Y_{i+1}). \qquad (12.2)$$

Слід зазначити, що для визначення Y_{i+1} треба розв'язати систему нелінійних рівнянь (12.2). Система (12.2) нелінійна оскільки в загальному випадку векторна функція $F(x,Y)$ нелінійна відносно Y Розв'язати систему нелінійних рівнянь (12.2) можна, наприклад, методом Ньютона (див. розділ 5.2).

Неявний метод Рунге – Кутта. У неявному методі Рунге – Кутта 4-го порядку для розв'язання жорстких систем звичайних диференціальних рівнянь (10.6) значення Y_i у вузлах сітки x_i, $i=\overline{1,n}$, рекурентно обчислюються, як розв'язання системи рівнянь [21]:

$$Y_{i+1}=Y_i+\frac{1}{6}(K_1^*+2K_2^*+2K_3^*+K_4^*),\ i=\overline{0,n-1}, \qquad (12.3)$$

де $K_1^*=hF(x_{i+1},Y_{i+1})$, $K_2^*=hF(x_{i+1}-\frac{h}{2},Y_{i+1}-\frac{h}{2}K_1^*)$,

$K_3^*=hF(x_{i+1}-\frac{h}{2},Y_{i+1}-\frac{h}{2}K_2^*)$, $K_4^*=hF(x_{i+1},Y_{i+1}-hK_3^*)$.

Глобальна похибка неявного методу Рунге – Кутта 4-го порядку, оцінювана для величини $\|Y_n - Y(x_n)\|$, має порядок $O(h^4)$, а локальна – $O(h^5)$ [21].

Приклад 12.1. Процес руху автомобіля на площині в найпростішому випадку може бути описаний системою диференціальних рівнянь [27]:

$$\begin{cases} \dfrac{\partial x}{\partial t} = V\cos\theta, \\ \dfrac{\partial y}{\partial t} = V\sin\theta, \\ \dfrac{\partial \theta}{\partial t} = -\dfrac{V}{W\,Ctg\phi + \dfrac{w}{2}}. \end{cases} , \qquad (12.4)$$

де (x, y) – координати точки M на площині xOy;

θ – кут між повздовжньою віссю автомобіля й віссю Ox;

V – швидкість;

ϕ.– кут повороту передніх коліс відносно повздовжньої осі автомобіля;

W – відстань між передньою і задньою осями (колісна база);

w – відстань між колесами автомобіля на задній осі (колія задніх коліс) (рис. 12.1).

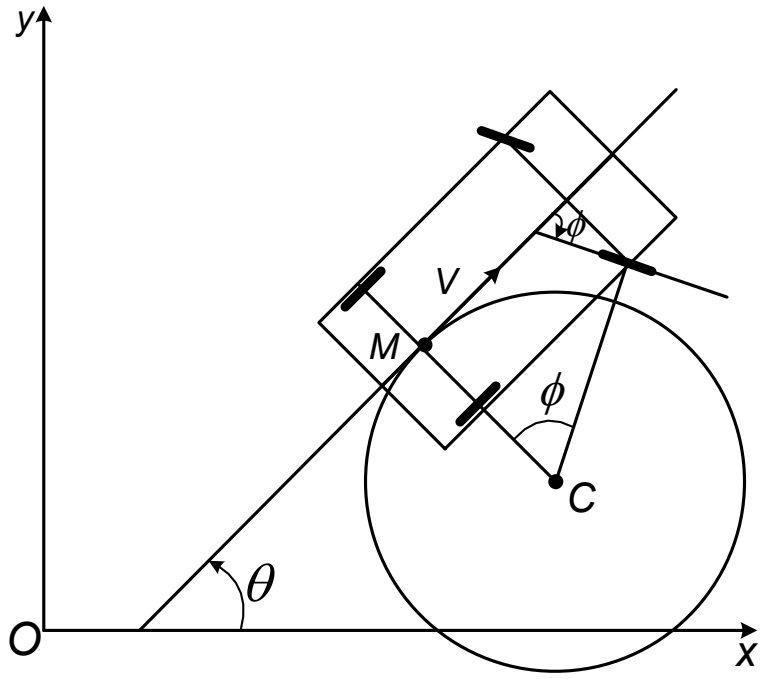


Рис. 12.1. **Модель руху автомобіля на площині**

У моделі (12.3) всі кути вимірюються в радіанах.

Треба визначити траєкторію руху автомобіля (тобто точки M) протягом 5 секунд, якщо його швидкість V була постійна і дорівнювала -3 км/год (задній хід), кут ϕ змінювався відповідно функції $\phi(t) = \frac{-1.606}{\pi} arctg(a_0 t + a_1)$, $a_0 = 11771.1$, $a_1 = -13.9164$, $W = 2.47$ м, $w = 1.456$ м, на початку руху точка M мала координати $(0, 2)$, а кут $\theta = 0$.

Цим прикладом імітується паралельна парковка автомобіля заднім ходом.

Розв'язання.

Оскільки в даному прикладі є три координати (x, y, θ), які змінюються у часі, то процедуру, що реалізує неявний метод Рунге – Кутта 4-го порядку, можна записати в такому вигляді:

```
# Задаємо функцію системи нелінійних рівнянь (12.3)
# для неявного методу Рунке-Кутта
SysFun = function(Y)
{
     K1 = h*DEfunPr(ti1, Y)
     K2 = h*DEfunPr(ti1-h/2, Y-h/2*K1)
     K3 = h*DEfunPr(ti1-h/2, Y-h/2*K2)
     K4 = h*DEfunPr(ti1, Y-h/2*K3)
     F = Yi + 1/6*(K1 + 2*K2 + 2*K3 + K4) - Y
     return(F)
}

MetRKipml = function(a, b, Y0, n)
{
     t = numeric(n+1)
     Y = matrix(0, nrow=(n+1), ncol=3)
     h = (b-a)/n
     assign("h", h, envir = .GlobalEnv)
     t[1] = a
     Y[1,] = Y0
     for( i in 1:n)
     {
          t[i+1] = t[i] + h
          assign("ti1", t[i+1], envir = .GlobalEnv)
          assign("Yi", Y[i,], envir = .GlobalEnv)
# Розвязуємо систему нелінійних рівнянь за допомогою
# функції пакета "nleqslv"
          Z = nleqslv(Y[i,], SysFun, control=list(btol=.01))
          Y[i+1,] = Z$x
     }
     return(cbind(t, Y))
}
```

Вводяться дані для розв'язання задачі:

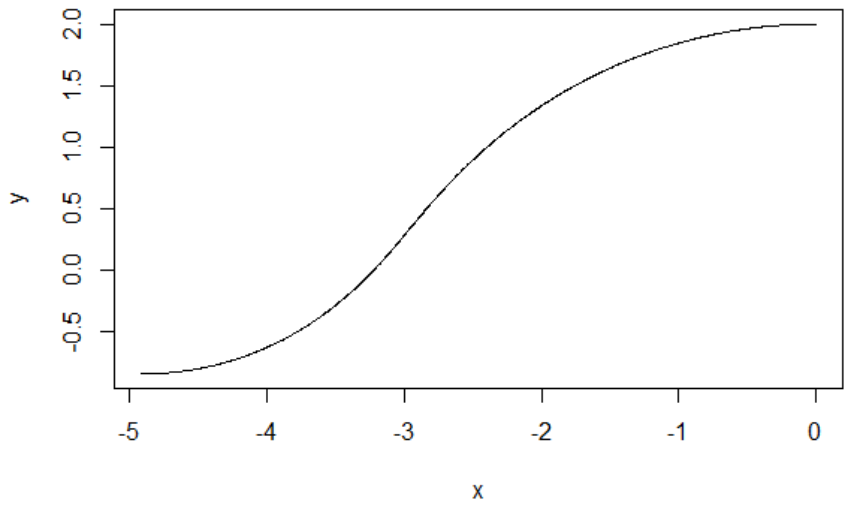
```
# Вводимо дані для розв’язання задачі
W = 2.47                  # колісна база
w = 1.456                 # колія задніх коліс
V = function(t){return(-3000)}# постійна швидкість заднім ходом
Fi = function(t){             # змінний кут повороту коліс
   a1 = -13.9164
   a0 = 11771.1
   return(-1.606/pi*atan(a0*t+a1))
}
a = 0                         # початковий момент часу
b = 0.00197         # кінцевий момент часу в годинах
# положення автомобіля у початковий момент часу
Y0 = c(0, 2, 0)
```

```
# Задаємо праву частину системи диференціальних рівнянь (12.4)
DEfunPr = function(t, Y)
{
     Ft = numeric(3)
     Ft[1] = V(t)*cos(Y[3])
     Ft[2] = V(t)*sin(Y[3])
     Ft[3] = -V(t)/(W/tan(Fi(t)) + w/2)
     return(Ft)
}
```

Викликається записана процедура й отримується розв'язок. Будується графік отриманого розв'язку:

```
> # Підключаємо бібліотеку для розв'язання
> # систем нелінійних рівнянь виду F(x)=0
> library("nleqslv")
```

```
> # Розв'язуємо систему диференціальних рівнянь
> n = 1000
> tY = MetRKipml(a, b, Y0, n)
> x = tY[, 2]
> y = tY[, 3]
> # Будуємо графік траєкторії руху автомобіля
> plot(x, y, type = "l")
```



Таким чином, виконана імітація руху автомобіля при паралельній парковці автомобіля заднім ходом.

12.3. Висновки

1. На практиці задача Коші як математична модель деяких динамічних процесів є найчастіше жорсткою.

2. У неявних методах крок h можна обирати більшим, ніж в явних методах.

12.4. Контрольні запитання та завдання

1. У чому полягає суть жорсткої задачі Коші для системи звичайних диференціальних рівнянь?

2. Які методи розв'язання задачі Коші для звичайних диференціальних рівнянь називаються неявними методами?

3. Які особливості мають неявні методи відносно явних методів?

4. Сформулюйте постановку задачі Коші для системи звичайних диференціальних рівнянь. Що є її розв'язком? У якому вигляді подається розв'язок чисельним методом?

5. Який з відомих вам методів розв'язання задачі Коші для системи звичайних диференціальних рівнянь треба застосовувати в тих чи інших випадках?

13. Крайові задачі для звичайних диференціальних рівнянь

13.1. Постановка крайової задачі

Крайова задача для звичайного диференціального рівняння 2-го порядку виду

$$F(x, y, y', y'') = 0 \tag{13.1}$$

полягає в такому: знайти функцію $y = y(x)$ на заданому відрізку $[a, b]$, що задовольняє рівняння (13.1) і крайові умови:

$$\begin{cases} \phi_1(y(a), y'(a)) = 0 \\ \phi_2(y(b), y'(b)) = 0 \end{cases}, \tag{13.2}$$

де F, ϕ_1, ϕ_2 – задані неперервні функції відповідної кількості аргументів.

Крайова задача (13.1), (13.2) називається **лінійною**, якщо всі функції F, ϕ_1, ϕ_2 лінійні відносно y, y', y''. Таким чином, лінійна крайова задача для звичайного диференціального рівняння 2-го порядку полягає в такому: знайти функцію $y = y(x)$ на заданому відрізку $[a,b]$, що задовольняє лінійне рівняння виду

$$y'' + p(x)y' + q(x)y = f(x) \tag{13.3}$$

і лінійні крайові умови

$$\begin{cases} \alpha_0 y(a) + \alpha_1 y'(a) = A \\ \beta_0 y(b) + \beta_1 y'(b) = B \end{cases}, \tag{13.4}$$

де $p(x)$, $q(x)$, $f(x)$ – задані неперервні функції від x;

α_0, α_1, β_0, β_1, A, B – задані константи, причому $|\alpha_0| + |\alpha_1| \neq 0$, $|\beta_0| + |\beta_1| \neq 0$.

Чисельні методи розв'язання задачі (13.1), (13.2), а зокрема і задачі (13.3), (13.4), знаходять розв'язок (тобто функцію $y(x)$ на відрізку $[a,b]$) у табличному вигляді, а саме у вигляді набору точок (x_i, y_i), $i = \overline{0,n}$, де $x_0 = a$, $x_i = x_0 + i\,h$, $i = \overline{1,n}$, $h = \dfrac{b-a}{n}$, n – задане число розбиття відрізка $[a,b]$, y_i, $i = \overline{0,n}$ – знайдені наближені значення функції $y(x)$ в точках x_i.

13.2. Метод кінцевих різниць для лінійних диференціальних рівнянь другого порядку

Варто розглянути **метод кінцевих різниць** розв'язання лінійної крайової задачі (13.3), (13.4).

Слід зазначити, що для пошуку чисельного розв'язку задачі (13.3), (13.4) необхідно знайти всі значення y_i, $i = \overline{0,n}$, тому вони розглядаються як невідомі.

Вводять позначення:

$$p_i = p(x_i),\ q_i = q(x_i),\ f_i = f(x_i),\ i = \overline{1,n}.$$

Замінюють наближено в кожному внутрішньому вузлі x_i похідні $y'(x_i)$, $y''(x_i)$ кінцево-різницевими формулами (див. розділ 8.2):

$$y'(x_i) \approx \frac{y_{i+1} - y_{i-1}}{2h},\ y''(x_i) \approx \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2},$$

а на кінцях відрізка $[a,b]$ покладають:

$$y'(x_0) \approx \frac{y_1 - y_0}{h},\ y'(x_n) \approx \frac{y_n - y_{n-1}}{h}.$$

Використовуючи ці формули, наближено замінюють рівняння (13.3) (в точках x_i, $i = \overline{1,n}$) і крайові умови (13.4) системою рівнянь:

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} + p_i \frac{y_{i+1} - y_{i-1}}{2h} + q_i y_i = f_i,\ (i = \overline{1,n-1}); \qquad (13.5)$$

$$\alpha_0 y_0 + \alpha_1 \frac{y_1 - y_0}{h} = A; \qquad (13.6)$$

$$\beta_0 y_n + \beta_1 \frac{y_n - y_{n-1}}{h} = B. \qquad (13.7)$$

Отримана система (13.5) – (13.7) є лінійною алгебраїчною системою $(n+1)$ рівнянь відносно $(n+1)$ невідомих y_i, $i = \overline{0,n}$. Розв'язавши її, якщо це можливо, і буде отримана таблиця наближених значень шуканої функції $y(x)$ на відрізку $[a,b]$.

Погрішність методу кінцевих різниць, що оцінюється для величин $|y_i - y(x_i)|$, $i = \overline{0,n}$, має порядок $O(h^2)$ [3; 5; 6; 19].

Варто додати, що систему (13.5) – (13.7) при розв'язанні краще записати у вигляді:

$$\begin{cases} (\alpha_0 h - \alpha_1) y_0 + \alpha_1 y_1 = Ah \\ (1 - \frac{p_i h}{2}) y_{i-1} + (q_i h^2 - 2) y_i + (1 + \frac{p_i h}{2}) y_{i+1} = f_i h^2\ (i = \overline{1,n-1}). \\ -\beta_1 y_{n-1} + (\beta_0 h + \beta_1) y_n + = Bh \end{cases} \qquad (13.8)$$

Слід звернути увагу на те, що система лінійних алгебраїчних рівнянь (13.5) має трьохдіагональний вигляд, а саме: в кожне i-те внутрішнє рівняння входять лише 3 невідомі y_{i-1}, y_i, y_{i+1}. Для розв'язання такого виду систем на практиці використовують **метод прогонки** [3; 5; 6; 19]. Цей чисельний метод, по суті, є модифікацією методу Гауса, пристосованою для прискореного пошуку розв'язку систем трьохдіагонального виду (див. розділ 3.3).

Приклад 13.1. Розв'язати методом кінцевих різниць крайову задачу $x^2y'' + xy' = 1$, $y(1) = 0$, $y(1.4) = 0{,}05661$ з числом розбиття відрізка $n = 10$.

Дана крайова задача є лінійною задачею, тому насамперед необхідно привести її до вигляду (13.3), (13.4). Тоді саме диференціальне рівняння запишеться так: $y'' + \frac{1}{x}y' = \frac{1}{x^2}$, тобто $p(x) = \frac{1}{x}$, $q(x) = 0$, $f(x) = \frac{1}{x^2}$.

Розв'язання в математичному пакеті R

Відправними даними задачі є:

1) задані неперервні функції від x: $p(x)$, $q(x)$, $f(x)$;

2) задані константи α_0, α_1, β_0, β_1, A, B;

3) відрізок $[a,b]$, на якому шукається розв'язок задачі;

4) число розбиття n відрізка $[a,b]$.

```
p1 = function(x) { return(1/x) }
q1 = function(x) { return(0) }
f1 = function(x) { return(1/(x^2)) }
a1 = 1
b1 = 1.4
alpha1 = c(1,0)
beta1 = c(1,0)
A1 = 0
B1 = 0.05661
n = 10
```

Оскільки розв'язання крайової задачі зводиться до розв'язання системи лінійних алгебраїчних рівнянь, то спочатку необхідно обчислити матрицю коефіцієнтів і вектор-стовпець вільних членів системи рівнянь (13.5).

Процедура обчислення матриці коефіцієнтів і вектора-стовпця вільних членів системи (13.8) і реалізація методу кінцевих різниць для розв'язання лінійної крайової задачі для диференціальних рівнянь другого порядку може бути записана так:

```
LinDEbp = function(p, q, f, a, b, alpha, beta, A, B, n)
{
  h = (b-a)/n
  x = c(1:(n+1))
  for (i in 1:(n+1))
  {
    x[i] = a + (i-1)*h
  }
  C = matrix(0, nrow=(n+1), ncol=(n+1))
  d = c(1:(n+1))

  C[1,1] = (alpha[1])*h - alpha[2]
  C[1,2] = alpha[2]
  C[(n+1),n] = -beta[2]
  C[(n+1),(n+1)] = (beta[1])*h+beta[2]
  for (i in 2:n)
  {
    C[i,(i-1)] = 1-h*p(x[i])/2
    C[i,i] = (h^2)*q(x[i])-2
    C[i,(i+1)] = 1+h*p(x[i])/2
  }
  d[1] = A*h
  d[n+1] = B*h
  for (i in 2:n)
  {
    d[i] = f(x[i])*h^2
  }
  y = solve(C, d)
  return(cbind(x, y))
}
```

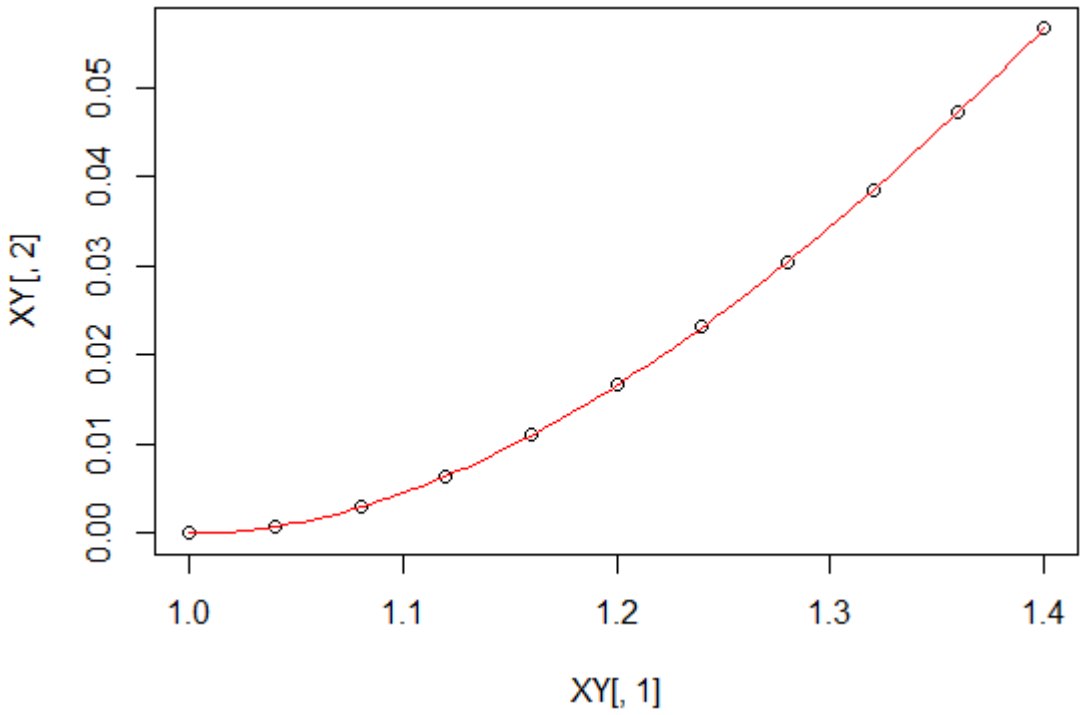
Для розв'язання системи була використана процедура *solve* пакета R, хоча краще було б реалізувати метод прогонки.

Розв'язком крайової задачі є таблично подана функція $y = y(x)$. Результат обчислення з використанням записаних процедур:

```
> n = 10
> XY = LinDEbp(p1, q1, f1, a1, b1, alpha1, beta1, A1, B1, n)
> XY
          x             y
 [1,] 1.00 -6.743312e-17
 [2,] 1.04  7.724224e-04
 [3,] 1.08  2.967076e-03
 [4,] 1.12  6.428724e-03
 [5,] 1.16  1.102204e-02
 [6,] 1.20  1.662857e-02
 [7,] 1.24  2.314416e-02
 [8,] 1.28  3.047698e-02
 [9,] 1.32  3.854571e-02
[10,] 1.36  4.727816e-02
[11,] 1.40  5.661000e-02
```

Аналітичним розв'язком заданого диференціального рівняння другого порядку є функція $YT(x) = 0.5(ln(x))^2$. Аналітичний і чисельний розв'язки можна подати графічно:

```
> # Будуємо графік чисельного розв'язку
> plot(XY[,1], XY[,2], type="p")
> # Порівнюємо отриманий розв'язок з точним (аналітичним)
> # розв'язком
> n1 = 100
> x = seq(a1,  b1, len=n1)
> y = c(1:n1)
> for(i in 1:n1){ y[i] = 0.5*(log(x[i]))^2 }
> lines(x, y, type="l", col="red")
```



З графіка видно, що чисельний та аналітичний розв'язки збігаються.

13.3. Метод кінцевих різниць для нелінійних диференціальних рівнянь другого порядку

Варто розглянути тепер крайову задачу для нелінійного диференціального рівняння:

$$y'' = f(x, y, y'); \tag{13.9}$$

при лінійних обмеженнях

$$\begin{cases} \alpha_0 y(a) + \alpha_1 y'(a) = A \\ \beta_0 y(b) + \beta_1 y'(b) = B \end{cases}, \tag{13.10}$$

де $f(x, y, y')$ – задана неперервна функція,

α_0, α_1, β_0, β_1, A, B – задані константи.

Метод кінцевих різниць для задачі (13.9), (13.10) записується таким чином.

Знову необхідно знайти всі наближені значення y_i, $i = \overline{0,n}$, функції $y = y(x)$ у рівновіддалених вузлах $x_0 = a$, $x_i = x_0 + i\,h$, $i = \overline{1,n}$ з кроком $h = \dfrac{b-a}{n}$, де n – задане число розбиття відрізка $[a,b]$.

Треба замінити наближено в кожному внутрішньому вузлі x_i похідні $y'(x_i)$, $y''(x_i)$ симетричними кінцево-різницевими формулами (див. розділ 8.2):

$$y'(x_i) \approx \frac{y_{i+1} - y_{i-1}}{2h},\ y''(x_i) \approx \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2},$$

а на кінцях відрізка $[a,b]$ покласти:

$$y'(x_0) \approx \frac{y_1 - y_0}{h},\ y'(x_n) \approx \frac{y_n - y_{n-1}}{h}.$$

Використовуючи ці формули, наближено замінюють рівняння (13.9) (в точках x_i, $i = \overline{1,n}$) і крайові умови (13.10) системою рівнянь:

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} = f\left(x_i, y_i, \frac{y_{i+1} - y_{i-1}}{2h}\right),\ (i = \overline{1,n-1}); \tag{13.11}$$

$$\alpha_0 y_0 + \alpha_1 \frac{y_1 - y_0}{h} = A; \tag{13.12}$$

$$\beta_0 y_n + \beta_1 \frac{y_n - y_{n-1}}{h} = B. \tag{13.13}$$

Таким чином, отримуємо систему (13.11) – (13.13) $(n+1)$ нелінійних алгебраїчних рівнянь відносно $(n+1)$ невідомих y_i, $i=\overline{0,n}$. Розв'язавши її, і буде отримана таблиця наближених значень шуканої функції $y(x)$ на відрізку $[a,b]$

Слід зазначити, що для розв'язання системи (13.11) – (13.13) можна скористатися методом Ньютона (див. розділ 5.2) або методом ітерацій (див. розділ 5.3) [3; 5; 6; 10].

Приклад 13.2. Розв'язати методом кінцевих різниць крайову задачу $y''=\frac{1-xy}{x^2}$, $y(1)=0$, $y(1.4)=0{,}05661$, з числом розбиття відрізка $n=10$.

Розв'язання в математичному пакеті R

Дана крайова задача може підпадає під вид (13.9), (13.10).

Відправними даними задачі є:

1) задана неперервна функція $f(x,y,y'')$;

2) задані константи α_0, α_1, β_0, β_1, A, B;

3) відрізок $[a,b]$, на якому шукається розв'язок задачі;

4) число розбиття n відрізка $[a,b]$.

```
f2 = function(x, y, yd) { return((1-x*yd)/(x^2)) }
a1 = 1
b1 = 1.4
alpha1 = c(1,0)
beta1 = c(1,0)
A1 = 0
B1 = 0.05661
eps = 0.00001
kmax = 100
n = 10
```

Систему (13.11) – (13.13) треба розв'язати методом ітерацій, тому для розв'язання допоміжної лінійної системи рівнянь необхідно запрограмувати обчислення елементів цієї системи. Тоді процедура розв'язання крайової задачі для нелінійних диференціальних рівнянь другого порядку методом кінцевих різниць може бути записана так:

```
# Програмуємо функцію обчислення норми вектора
Norma = function(x){
  return( sqrt(crossprod(x, x)) )
}
```

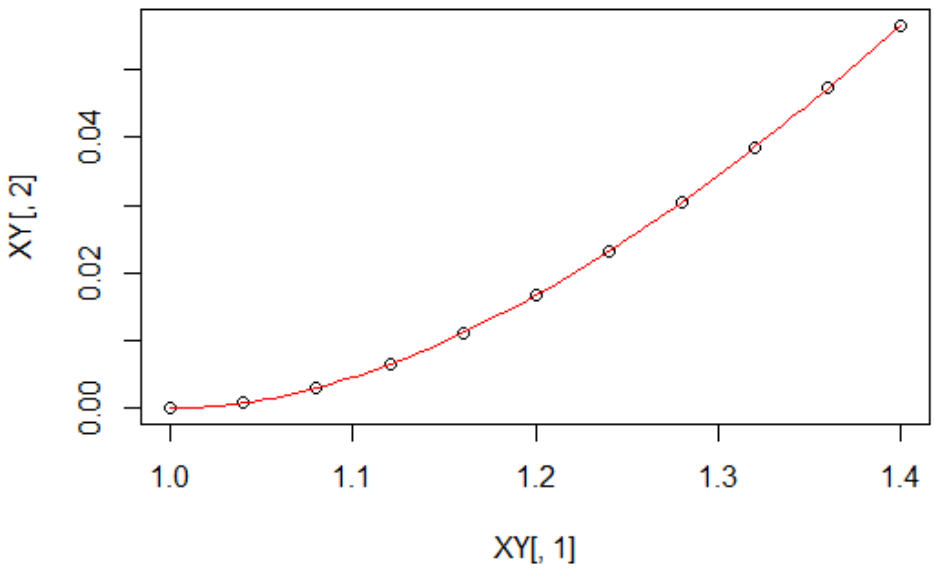
```
# процедура розв'язку лінійного диф. рівняння
# методом кінцевих різниць
nLinDEbp = function(f, a, b, alpha, beta, A, B, n, eps, kmax)
{
  h = (b-a)/n
  x = c(1:(n+1))
  y = c(1:(n+1))

  for (i in 1:(n+1))
    x[i] = a + (i-1)*h
  y[1] = A
  y[n+1] = B
  # Задаємо початкове значення для шуканих y
  for (i in 2:n)
    y[i] = (y[1] + y[n+1])/2

  C = matrix(0, nrow=(n+1), ncol=(n+1))
  d = c(1:(n+1))

  C[1,1] = (alpha[1])*h - alpha[2]
  C[1,2] = alpha[2]
  C[(n+1),n] = -beta[2]
  C[(n+1),(n+1)] = (beta[1])*h+beta[2]
  d[1] = A*h
  d[n+1] = B*h
  for (i in 2:n)
  {
    C[i,(i-1)] = 1
    C[i,i] = -2
    C[i,(i+1)] = 1
  }
  for (k in 0:kmax)
  {
     for (i in 2:n)
     {
         ydi = (y[i+1] - y[i-1])/(2*h)
          d[i] = f(x[i], y[i], ydi)*h^2
     }
     y1 = solve(C, d)
     if(Norma(y1-y) <= eps)
          break
      y = y1
  }
  return(cbind(x, y1))
}
```

Розв'язком крайової задачі є таблично подана функція $y = y(x)$. Результат обчислення з використанням записаних процедур:

```
> n = 10
> XY = nLinDEbp(f2, a1, b1, alpha1, beta1, A1, B1, n, eps, kmax)
> XY
          x             y1
 [1,] 1.00 -2.493665e-17
 [2,] 1.04  7.725171e-04
 [3,] 1.08  2.967249e-03
 [4,] 1.12  6.428928e-03
 [5,] 1.16  1.102222e-02
 [6,] 1.20  1.662868e-02
 [7,] 1.24  2.314420e-02
 [8,] 1.28  3.047695e-02
 [9,] 1.32  3.854565e-02
[10,] 1.36  4.727811e-02
[11,] 1.40  5.661000e-02
```

Аналітичним розв'язком заданого диференціального рівняння другого порядку є функція $YT(x) = 0.5(ln(x))^2$. Аналітичний і чисельний розв'язки можна подати графічно:

```
> # Будуємо графік чисельного розв'язку
> plot(XY[,1], XY[,2], type="p")
> # Порівнюємо отриманий розв'язок
> # з точним (аналітичним) розв'язком
> n1 =100
> x = seq(a1,  b1, len=n1)
> y = c(1:n1)
> for(i in 1:n1){ y[i] = 0.5*(log(x[i]))^2 }
> lines(x, y, type="l", col="red")
```



З графіка видно, що чисельний та аналітичний розв'язки збігаються.

У прикладах 13.1 і 13.2 була розв'язана одна й та ж крайова задача, але різними методами. Отримані результати співпадають.

13.4. Висновки

1. Математичні моделі процесів та явищ, зокрема моделі динамічних систем, у більшості випадків записуються у вигляді диференціальних рівнянь.

2. Задача Коші і крайова задача для звичайних диференціальних рівнянь має велике практичне значення.

13.5. Контрольні запитання та завдання

1. Що називається звичайним диференціальним рівнянням?

2. Сформулюйте постановку крайової задачі для звичайного диференціального рівняння. Що є її розв'язком? У якому вигляді подається розв'язок чисельним методом?

3. В чому суть методу кінцевих різниць для лінійної крайової задачі? Які він має оцінки погрішності?

4. Розв'яжіть методом кінцевих різниць крайову задачу $y'' + x^2 y' + (1-x)y = \dfrac{x}{x^2+3}$, $y(0)=0$, $y(1)=0$, з числом розбиття відрізка $n=25$. Побудуйте графік отриманого розв'язку.

5. Розв'яжіть методом кінцевих різниць нелінійне диференціальне рівняння другого порядку $y'' - 2xy' - 2y^2 = -4x$, $y(0)=0$, $y(1)=3.7$ з числом розбиття відрізка $n=20$. Побудуйте графік отриманого розв'язку.

14. Чисельні методи розв'язання інтегральних рівнянь

14.1. Поняття та класифікація інтегральних рівнянь

Визначення. Інтегральними рівняннями 2-го роду називаються рівняння виду:

$$\phi(x) - \lambda \int_a^b K(x,t)\phi(t)\,dt = f(x), \qquad (14.1)$$

де $\phi(x)$ – шукана (невідома) функція;

λ – заданий параметр рівняння;

$K(x,t)$ – ядро рівняння;

$f(x)$ – вільний член (права частина) рівняння, причому, $K(x,t)$ та $f(x)$ – задані (відомі) функції.

Необхідність у чисельному розв'язанні інтегральних рівнянь виникає при дослідженні об'єктів різноманітних типів у різних галузях науки та техніки. Наприклад, у механіці (аналіз жорсткості конструкцій), в фізиці (кристалографія, теорія плазми, спектроскопія), в радіотехніці (оптимальна лінійна фільтрація) та інших.

Визначення. Інтегральними рівняннями 1-го роду називаються рівняння виду:

$$\int_a^b K(x,t)\phi(t)dt = f(x). \qquad (14.2)$$

Якщо ядро $K(x,t)$ неперервне для $\forall\, x \in [a,b]$ та $\forall\, t \in [a,b]$, або має такі розриви, що інтеграл $\int_a^b\int_a^b \left|K^2(x,t)\right| dx\,dt$ є обмеженим, то рівняння (14.1) та (14.2) називаються **інтегральними рівняннями Фредгольма**, відповідно 2-го та 1-го роду.

Якщо в рівнянні Фредгольма для ядра $K(x,t)$ виконується умова: $K(x,t) = 0$, $\forall\, t > x$, то рівняння (14.1) або (14.2) називається **інтегральним рівнянням Вольтера.** При цьому для рівняння 2-го роду (14.1) воно буде мати вигляд:

$$\phi(x) - \lambda\int_a^x K(x,t)\phi(t)dt = f(x),$$

а для рівняння 1-го роду (14.2) – вигляд:

$$\int_a^x K(x,t)\phi(t)dt = f(x).$$

Інтегральні рівняння взаємопов'язані з задачею Коші та крайовою задачею для звичайних диференціальних рівнянь, а саме ці задачі можна записати через інтегральні рівняння [21]. Задача Коші відповідає рівнянню Вольтера, а крайова задача – рівнянню Фредгольма.

14.2. Чисельні методи розв'язання інтегральних рівнянь

14.2.1. Рівняння з виродженим ядром

Розглядається інтегральне рівняння (14.1) з **виродженим ядром** [21], тобто:

$$K(x,t) = \sum_{i=1}^{n} g_i(x) h_i(t),$$

де $g_i(x)$, $h_i(t)$, $i = \overline{1,n}$ – деякі відомі функції.

Тоді рівняння (14.1) можна переписати таким чином:

$$\phi(x) - \lambda \sum_{i=0}^{n} g_i(x) \int_a^b h_i(t)\phi(t)dt = f(x).$$

Якщо ввести позначення $c_i = \int_a^b h_i(t)\phi(t)dt$, тоді шукану функцію $\phi(x)$ можна подати у вигляді:

$$\phi(x) = \lambda \sum_{i=0}^{n} g_i(x) c_i + f(x).$$

Таким чином, для знаходження функції $\phi(x)$ достатньо знайти коефіцієнти c_i, $i = \overline{1,n}$. Їх можна зайти розв'язавши систему лінійних алгебраїчних рівнянь:

$$c_k = \lambda \sum_{i=0}^{n} a_{ki} c_i + f_k,\ k = \overline{1,n},$$

де $f_k = \int_a^b h_k(t)f(t)dt$, $a_{ki} = \int_a^b g_k(t)h_i(t)dt$.

Як видно, цей метод є прямим чисельним методом та зводиться до розв'язання системи лінійних алгебраїчних рівнянь.

14.2.2. Метод квадратурних сум

Суть **методу квадратурних сум** полягає в заміні інтеграла, що фігурує в рівнянні (14.1) однією з квадратурних сум, тобто

$$\int_a^b y(t)dt = \sum_{i=0}^{n} A_i y(t_i),$$

де t_i, $i = \overline{0,n}$ – вузли сітки на відрізку $[a,b]$;

A_i, $i = \overline{1,n}$, – коефіцієнти виду квадратурної суми, наприклад, формули трапецій, формули Сімпсона (див. розділ 9) або формули прямокутників [21].

Розв'язок $\phi(x)$ рівняння (14.1) на відрізку $[a,b]$ шукається в табличному вигляді, тобто треба знайти значення ϕ_i цієї функції у вузлах сітки x_i, $i=\overline{0,n}$.

З рівняння (14.1) витікає, що наближений розв'язок буде мати вигляд:

$$\phi(x)=\lambda\sum_{i=0}^{n}A_iK(x_i,x_i)\phi(x_i)+f(x).$$

Тут $x_i=t_i=a+i\times h$, $i=\overline{0,n}$, $h=\dfrac{b-a}{n}$, n – число розбиття відрізка $[a,b]$. Тоді для вузлів сітки x_k, $k=\overline{0,n}$, на відрізку $[a,b]$ повинні виконуватись рівняння:

$$\phi(x_k)=\lambda\sum_{i=0}^{n}A_iK(x_i,x_i)\phi(x_i)+f(x_k),\ k=\overline{0,n}.$$

Таким чином, для знаходження значень $\phi_i\approx\phi(x_i)$, $i=\overline{0,n}$, треба розв'язати систему лінійних алгебраїчних рівнянь:

$$\phi(x_k)=\lambda\sum_{i=0}^{n}A_iK(x_i,x_i)\phi(x_i)+f(x_k),\ k=\overline{0,n}.$$

Як видно, метод квадратурних сум є прямим чисельним методом та зводиться до розв'язання системи лінійних алгебраїчних рівнянь.

Для равновідстоящих вузлів сітки нерідко застосовують формулу трапецій (див. розділ 9.2), з якої виходить рекурентна формула для знаходження $\phi_i\approx\phi(x_i)$, $i=\overline{0,n}$ [21]:

$$\phi(x_0)=f(a)$$

$$\phi(x_i)=\frac{f(x_i)+h\sum_{j=1}^{i-1}A_i\,K(x_i,x_j)\phi(x_j)}{1-\dfrac{h}{2}K(x_i,x_i)},\ i=\overline{1,n},$$

де $A_1=\dfrac{1}{2}$, $A_j=1$ при $j>1$.

14.2.3. Метод послідовних наближень

Цей метод по суті є методом ітерацій, застосований для розв'язання рівняння (14.1).

У методі послідовних наближень [3; 6; 21] при розв'язанні рівняння (14.1) будується ітераційна послідовність функцій $\{\phi_k(x)\}$, $k = 0, 1, 2,...$ за рекурентною формулою:

$$\phi_{k+1}(x) = f(x) + \lambda \int_a^b K(x,t)\phi_k(t)dt.$$

Початкове наближення, наприклад, можна узяти таким: $\phi_0(x) = f(x)$.

Умова збіжності. Послідовність функцій $\{\phi_k(x)\}$, $k = 0, 1, 2,...$, збігається до розв'язку $\phi(x)$ рівняння (14.1), якщо виконується умова [21]: $|\lambda| < \frac{1}{B}$, де $B = \sqrt{\int_a^b \int_a^b K^2(x,t)dx\,dt}$.

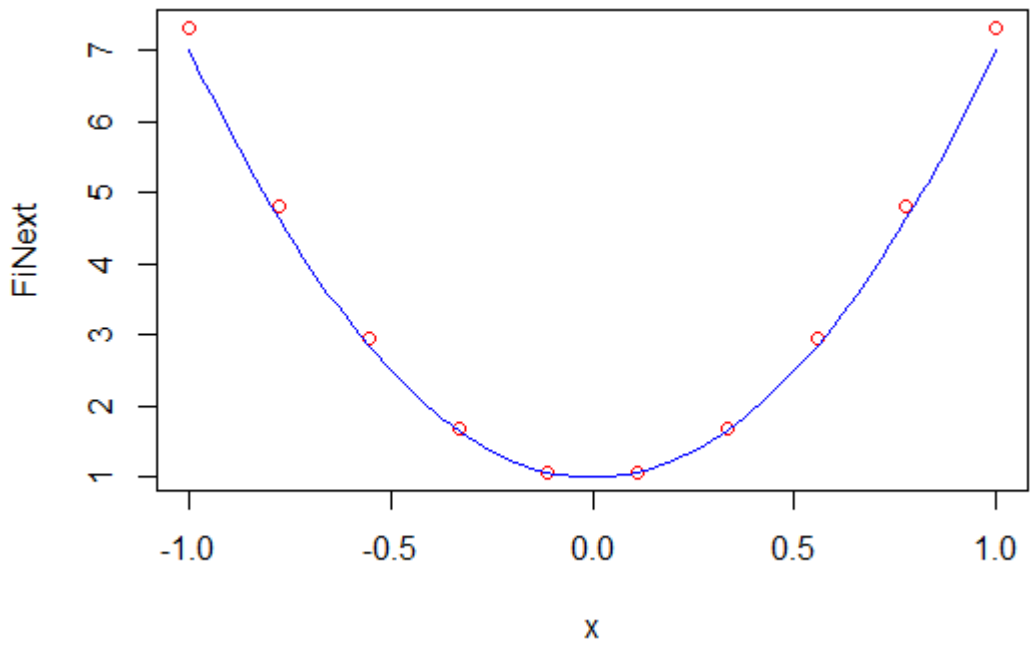
Швидкість збіжності при цьому буде лінійна, а саме: $|\phi_k(x) - \phi(x)| \le M|\lambda B|^k$, де $M > 0$.

Приклад 14.1. Розв'язати методом послідовних наближень інтегральне рівняння [21] $\phi(x) = 1 + \int_{-1}^{1}(xt + x^2)\phi(t)dt$ (аналітичний розв'язок $\phi(x) = 1 + 6x^2$).

Розв'язання в математичному пакеті R

```
# Задаємо границі інтегрування і параметр lambda
a=-1; b=1; lambda=1
# Задаємо функцію ядра
FunCore = function(x, t){ return( x*t+x*x ) }
# Задаємо функцію правої частини рівняння
FunF = function(x){ return( 1 ) }
# Програмуємо функцію обчислення інтеграла за формулою трапецій
# для таблично заданої функції
MetTrap = function(x, y, n){
  h = x[2] - x[1]
  S = (y[1] + y[n])/2
  for(i in 2:(n-1)){ S = S + y[i] }
  return( h*S )
}
```

```
# Програмуємо функцію обчислення норми вектора
Norma = function(x){
  return( sqrt(crossprod(x, x)) )
}
# Задаємо число точок сітки на відрізку [a,b] і точність
n = 10; eps = 1E-5
# Визначаємо точки сітки
h = (b-a)/(n-1)
x = c(1:n)
for(i in 1:n){ x[i] = a + h*(i-1) }
t = c(1:n)
t = x
# Обчислюємо значення функції ядра в точках сітки
K = matrix(0, ncol=n, nrow=n)
for(i in 1:n){
  for(j in 1:n){ K[i,j] = FunCore(x[i], t[j]) }
}
Fi = c(1:n); FiNext = c(1:n)
# Задаємо початкове наближення розв'язку
for(i in 1:n){
  Fi[i] = FunF(x[i]) }
> # Запускаємо ітераційний процес
> y = c(1:n)
> for(ki in 1:100)
+ {
+   # Обчислюємо наступне наближення розв'язку
+   for(i in 1:n)
+   {
+     for(j in 1:n){ y[j] = K[i,j]*Fi[j] }
+     FiNext[i] = FunF(x[i]) + lambda*MetTrap(t, y, n)
+   }
+   # Порівнюємо два наближення розв'язку
+   if( Norma(FiNext-Fi) <= eps)
+     break
+   Fi = FiNext
+ }
> ki
[1] 35
> plot(x, FiNext, type="p", col="red")
> # Порівнюємо отриманий розв'язок з
> # точним (аналітичним) розв'язком
> t = seq(a,  b, len=100)
> for(i in 1:100){ y[i] = 1+6*t[i]*t[i] }
> lines(t, y, type="l", col="blue")
```



Як видно з графіка, чисельний та аналітичний розв'язки не зовсім співпадають, але це залежить від обраного числа точок (10) сітки на відрізку $[a,b]$. При цьому для досягнення заданої точності знадобилось 35 ітерацій методу послідовних наближень.

14.2.4. Методи апроксимуючих функцій

У методах апроксимуючих функцій [3; 6; 21], наближений розв'язок $\tilde{\phi}(x)$ інтегрального рівняння (14.1) знаходиться у вигляді:

$$\tilde{\phi}(x) = \sum_{i=1}^{n} c_i U_i(x), \qquad (14.3)$$

де $U_i(x)$, $i = \overline{1,n}$, – система базисних лінійно незалежних функцій (тобто $\int_a^b U_i(x)U_j(x)dx = 0 \ \forall\, i \neq j$);

c_i, $i = \overline{1,n}$, – шукані коефіцієнти.

У якості функцій $U_i(x)$, $i = \overline{1,n}$, можна, наприклад, узяти ортогональні поліноми Лежандра [21]:

$$U_i(x) = P_i(z),$$

де $P_1(z) = 1$, $P_2(z) = z$, $P_3(z) = \frac{3z^2 - 1}{2}$, $P_4(z) = \frac{5z^3 - 3z}{2}$; $z = \frac{2x - b - a}{b - a}$.

Після підстановки виразу (14.3) у рівняння (14.1), буде отримана нев'язка:

$$R(x;c_1,c_2,\ldots,c_n)=\sum_{i=1}^{n}c_i\left[U_i(x)-\lambda\int_a^b K(x,t)U_i(t)dt\right]-f(x), \qquad (14.4)$$

де шукані коефіцієнти c_i, $i=\overline{1,n}$, у (14.3) визначаються після введення додаткових умов.

Варто розглянути два методи з групи методів апроксимуючих функцій.

У **методі колокацій** вимагається, щоб нев'язка (14.4) у вузлах колокацій $a\le x_1\le\ldots\le x_k\le\ldots\le x_n\le b$ – дорівнювала нулю:

$$R(x_k;c_1,c_2,\ldots,c_n)=0,\ k=\overline{1,n}. \qquad (14.5)$$

Після знаходження коефіцієнтів c_i, $i=\overline{1,n}$, з системи лінійних алгебраїчних рівнянь (14.5) наближений розв'язок інтегрального рівняння (14.1) записується у вигляді (14.3).

У **методі найменших квадратів** розглядається функція

$$\Phi(c)\equiv\sum_{k=1}^{m}[R(x_k;c_1,c_2,\ldots,c_n)]^2$$

у m вузлах сітки $a\le x_1\le\ldots\le x_k\le\ldots\le x_m\le b$, де $m\ge n$. Тоді шукані коефіцієнти c_i, $i=\overline{1,n}$ знаходяться як точка мінімуму функції $\Phi(c)$ (тут $c=(c_1,c_2,\ldots,c_n)^T\in R^n$), після чого наближений розв'язок інтегрального рівняння (14.1) записується у вигляді (14.3).

Слід зазначити, що метод колокацій і метод найменших квадратів є прямі чисельні методи на відміну від методу послідовних наближень.

14.2.5. Метод моментів

Метод моментів ще носить назву **методу Гальоркіна**.

У методі моментів [3; 6; 21], наближений розв'язок $\tilde{\phi}(x)$ інтегрального рівняння (14.1) знаходиться у вигляді:

$$\tilde{\phi}(x)=f(x)+\sum_{i=1}^{n}c_iU_i(x), \qquad (14.6)$$

де $U_i(x)$, $i=\overline{1,n}$, – система базисних лінійно незалежних функцій (також як і у методах апроксимуючих функцій);

c_i, $i=\overline{1,n}$, – шукані коефіцієнти.

Після підстановки $\tilde{\phi}(x)$ в інтегральне рівняння (14.1) (подібно тому, як і у методах апроксимуючих функцій), буде отримана функція нев'язки:

$$R(x,c)=\sum_{i=1}^{n} c_i\left(U_i(x)-\int_a^b K(x,t)U_i(t)\,dt\right).$$

Далі вимагається, щоб функція $R(x,c)$ була ортогональною усім функціям $U_i(x)$, $i=\overline{1,n}$, тобто буде система лінійних алгебраїчних рівнянь

$$\int_a^b R(x,c)U_k(t)\,dx=0,\ k=\overline{1,n},$$

для пошуку коефіцієнтів c_i, $i=\overline{1,n}$.

Слід зазначити, що метод моментів також є прямим чисельним методом розв'язання інтегрального рівняння (14.1).

14.3. Висновки

1. Задача розв'язання інтегрального рівняння пов'язана з задачею Коші та з крайовою задачею для звичайних диференціальних рівнянь.

2. Існує багато методів розв'язання інтегральних рівнянь – як прямих, так і ітераційних.

3. У прямих методах пошук розв'язку інтегрального рівняння зводиться до розв'язання системи лінійних алгебраїчних рівнянь.

14.4. Контрольні запитання та завдання

1. Які рівняння називаються інтегральними?
2. Наведіть класифікацію інтегральних рівнянь.
3. Які є методи розв'язання інтегральних рівнянь? У чому полягає їх ідея?

Розв'язати методом послідовних наближень інтегральне рівняння $\phi(x)=1+\frac{1}{2}\int_0^7 \phi(t)\,dt$ та порівняти його з точним розв'язком $\phi(x)=e^x$ [21].

15. Методи математичної фізики

15.1. Розв'язання диференціальних рівнянь з частинними похідними

Інженеру дуже часто доводиться стикатися з задачами, в яких шукана величина залежить від декількох змінних, наприклад, $u = u(t, x, y, z)$. У цьому випадку розв'язувані диференціальні рівняння містять частинні похідні і називаються **диференціальними рівняннями з частинними похідними** [3; 10; 21; 23; 24].

Рівняннями з частинними похідними описується багато фізичних процесів у таких областях, як механіка суцільних середовищ, термодинаміка, квантова механіка, електродинаміка, теорія пружності і багато ін. Тому розділ математики, що вивчає можливість розв'язання диференціальних рівнянь із частинними похідними називається математичною фізикою, а рівняння – **рівняннями математичної фізики**.

Нажаль, дуже багато з таких рівнянь не мають аналітичного розв'язку, і щоб їх розв'язати, доводиться вдаватися до чисельних методів. Якщо для розв'язання звичайних диференціальних рівнянь існує багато різних методів (див. розділи 10 – 13), то для розв'язання диференціальних рівнянь із частинними похідними доводиться вибирати в основному між методом кінцевих різниць і методом кінцевих елементів.

У цьому розділі питання про чисельне інтегрування диференціальних рівнянь із частинними похідними розглядається з точки зору застосування цих методів для вирішення різних технічних завдань. Дається також класифікація диференціальних рівнянь з частинними похідними, що часто зустрічаються, і вказуються раціональні шляхи їх чисельного розв'язання.

15.1.1. Класифікація диференціальних рівнянь з частинними похідними

Диференціальні рівняння з частинними похідними класифікують або залежно від їх математичної природи (еліптичні, параболічні тощо), або залежно від фізичного змісту розв'язуваних за їх допомогою задач (рівняння дифузії, хвильове рівняння тощо). Для того, щоб користуватися

математичною літературою та літературою з прикладних дисциплін, фахівець повинен бути знайомий з обома цими класифікаціями [24].

Для спрощення викладання слід розглянути випадок, коли шукана величина залежить від двох змінних, тобто $u = u(x, y)$.

З математичної точки зору диференціальні рівняння другого порядку з частинними похідними з двома незалежними змінними:

$$a(x,y)\frac{\partial^2 u}{\partial x^2} + 2b\frac{\partial^2 u}{\partial x \partial y} + c(x,y)\frac{\partial^2 u}{\partial y^2} + d(x,y,u,\frac{\partial u}{\partial x},\frac{\partial u}{\partial y}) = 0, \qquad (15.1)$$

класифікуються залежно від співвідношення функцій a, b, c. Ці функції залежать від змінних x і y. Якщо $b^2 - ac < 0$, рівняння називається **еліптичним**, якщо $b^2 - ac = 0$ – **параболічним**, а якщо $b^2 - ac > 0$ – **гіперболічним**. Залежність функцій a, b, c від x і y ускладнює ситуацію, оскільки робить можливим зміну типу рівняння при переході з однієї частини розглянутої області в іншу.

Додатковими умовами для диференціальних рівнянь другого порядку з частинними похідними можуть слугувати граничні або початкові умови, а також комбінація тих і інших. Еліптичні рівняння описують усталені (стаціонарні) процеси; задача ставиться в замкнутій області, і в кожній точці межі цієї області задаються граничні умови. Параболічними і гіперболічними рівняннями описуються еволюційні (нестаціонарні) процеси (процеси "поширення"). У таких задачах на одній частині границі ставляться граничні умови, на іншій – початкові; можливі також відкриті області, в які "поширюється розв'язок".

В інженерній практиці найчастіше зустрічаються рівняння з частинними похідними, що наведені в табл. 15.1 [24], де Δ – **оператор Лапласа**. У випадку однієї незалежної змінної x він має вигляд $\Delta u = \frac{\partial^2 u}{\partial x^2}$, а у випадку двох незалежних змінних x і y – $\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$.

Оператор Δ^2 називається **бігармонічним оператором** і у випадку двох незалежних змінних він має вигляд $\Delta^2 u = \frac{\partial^4 u}{\partial x^4} + 2\frac{\partial^4 u}{\partial x^2\ \partial y^2} + \frac{\partial^4 u}{\partial y^2}$.

Оператори Δ та Δ^2 застосовуються для скорочення запису диференціальних рівнянь.

Таблиця 15.1

Диференціальні рівняння з частинними похідними, що найчастіше зустрічаються в інженерній практиці

Назва рівняння	Математична форма	Приклади практичних задач, де вони зустрічаються
Лапласа (еліптичне)	$\Delta u = 0$	Усталена течія рідини стаціонарних потоках. Стаціонарні теплові поля
Пуасона (еліптичне)	$\Delta u = -k$	Теплопередача з внутрішніми джерела тепла
Теплопровідності або дифузії (параболічне)	$\Delta u = \frac{1}{a^2}\frac{\partial u}{\partial t}$	Нестаціонарна теплопровідність
Хвильове (гіперболічне)	$\Delta u = \frac{1}{c^2}\frac{\partial^2 u}{\partial t^2}$	Розповсюдження звукових хвиль
Бігармонічне	$\Delta^2 u = F(x, y)$	Деформація пластин

15.1.2. Метод кінцевих різниць для розв'язання диференціальних рівнянь із частинними похідними

В основі розв'язання рівнянь з частинними похідними методом кінцевих різниць лежить, природно, – різницева апроксимація похідних, яка багато в чому нагадує апроксимацію похідних, що була описана в розділі 13, присвяченому розв'язанню крайової задачі для звичайних диференціальних рівнянь. Розв'язання здійснюється в три етапи (рис. 15.1). Спочатку в області розв'язку вводять рівномірну сітку "вузлових точок", що відповідає характеру задачі і граничним умовам.

Потім розв'язуване рівняння з частинними похідними записують у найбільш зручній системі координат і, представляючи похідні в кінцево-різницевій формі, приводять його до виду різницевого рівняння. Отримане різницеве рівняння використовують надалі для опису функціонального зв'язку між сусідніми вузлами сітки. Різницеве рівняння записують для всіх вузлів сітки і отримують в результаті систему n алгебраїчних рівнянь з n невідомими. На останньому етапі отриману систему алгебраїчних рівнянь розв'язують одним із чисельних методів. На перший погляд, ця процедура, яка складається з трьох етапів, може здатися простою і такою, що прямо проводить до розв'язку, однак насправді це не так – широке розмаїття типів і розмірів сіток, видів рівнянь із частинними похідними, можливих кінцево-різницевих апроксимацій цих рівнянь і методів розв'язання отриманих систем алгебраїчних рівнянь роблять задачу чисельного розв'язання рівнянь із частинними похідними виключно багатогранним і цікавим дослідженням.

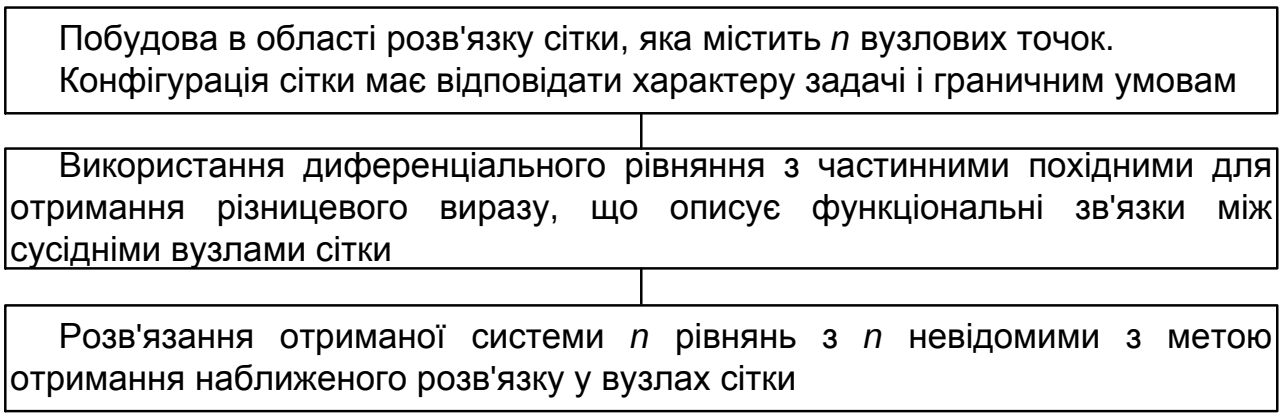


Рис. 15.1. **Етапи чисельного розв'язання диференціальних рівнянь із частинними похідними методом кінцевих різниць**

Варто розглянути тепер всі три етапи рішення детальніше.

Етап 1. Сітки, що застосовуються при поданні диференціальних рівнянь **з частинними похідними в** кінцево-різницевій формі.

Усі раніше наведені рівняння з частинними похідними були записані в декартовій системі координат, однак іноді буває зручніше користуватися іншими системами координат, що володіють спеціальними геометричними властивостями і враховують форму фізичного тіла [24].

Найчастіше в інженерній практиці застосовується декартова, циліндрична та сферична системи координат. На рис. 15.2 показані сітки,

які найчастіше застосовують при розв'язанні рівнянь з частинними похідними – прямокутна, полярна, трикутна та скошена. Нерідко доводиться мати справу з областями неправильної форми, наприклад, такими, як зображено на рис. 15.3.

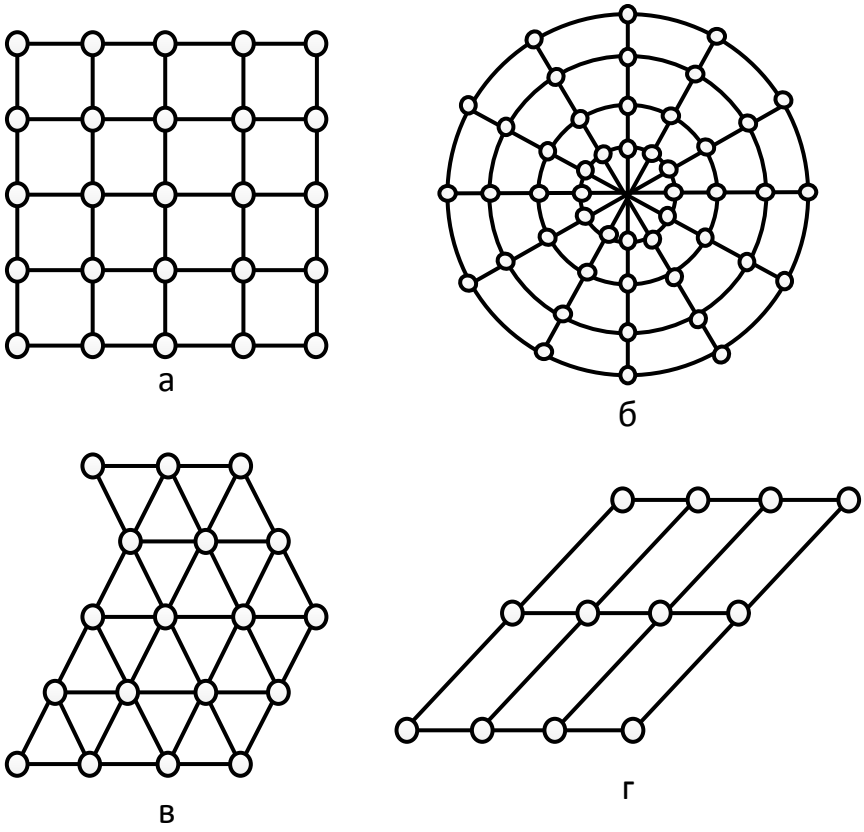


Рис. 15.2. **Види сіток, які найчастіше використовуються при чисельному розв'язанні диференціальних рівнянь в частинних похідних:** а – прямокутна, б – полярна, в – трикутна, г – скошена

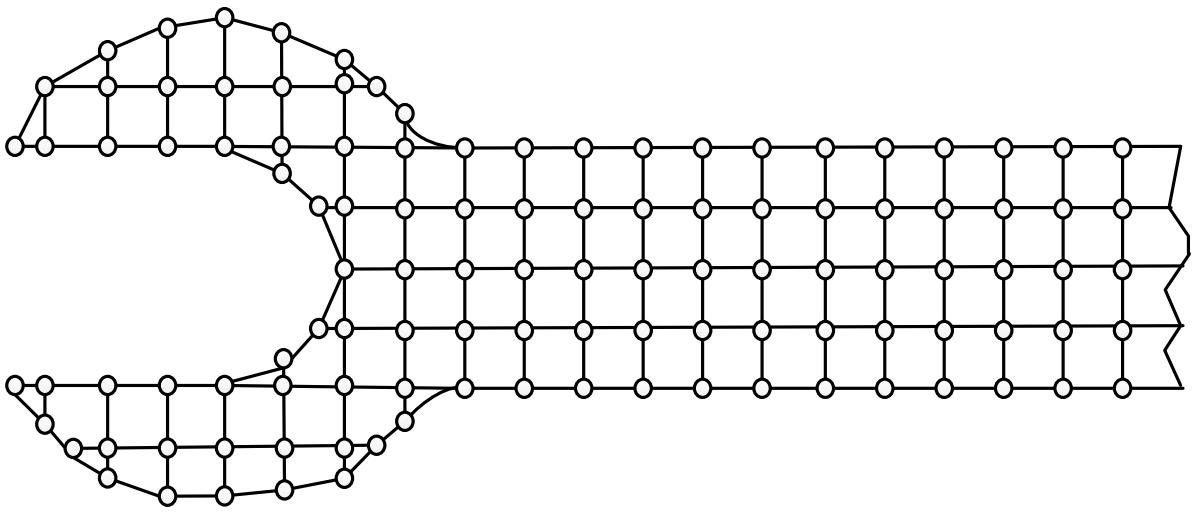


Рис. 15.3. **Приклад задачі з границею складної форми конфігурації**

Етап 2. Подання частинних похідних у кінцево-різницевому вигляді.

Для подання частинних похідних у кінцево-різницевому вигляді застосовують формули, аналогічні формулам чисельного диференціювання, що були розглянуті в розділі 8.2. Вони тільки переписуються для випадку декількох змінних. Для двох змінних на практиці найчастіше застосовують симетричні формули:

$$\frac{\partial u}{\partial x} \approx \frac{u_{i+1,j} - u_{i-1,j}}{2h}, \tag{15.2}$$

$$\frac{\partial u}{\partial y} \approx \frac{u_{i,j+1} - u_{i,j-1}}{2h}, \tag{15.3}$$

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}, \tag{15.4}$$

$$\frac{\partial^2 u}{\partial y^2} \approx \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2}, \tag{15.5}$$

$$\frac{\partial^2 u}{\partial x \partial y} \approx \frac{u_{i+1,j+1} - u_{i-1,j+1} - u_{i+1,j-1} + u_{i-1,j-1}}{4h^2}. \tag{15.6}$$

Тут $u_{k,m}$ – значення функції $u = u(x, y)$ у вузлах, розташованих в околі центральної точки (x_i, y_j), якій відповідає значення $u_{i,j}$.

Етап 3. Чисельні методи розв'язання систем лінійних алгебраїчних рівнянь.

Для розв'язання систем лінійних алгебраїчних рівнянь на 3-му етапі методу кінцевих різниць застосовують чисельні методи, що були описані в розділах 2 та 3. Зазвичай матриці такої системи виявляються розрідженними, оскільки в більшій частині розрахункових схем застосовуються лише сусідні вузли, а не всі вузли сітки.

15.1.3. Метод кінцевих елементів для розв'язання диференціальних рівнянь із частинними похідними

Метод кінцевих елементів для опису суцільних середовищ уперше було застосовано в середені 50-х років XX століття. Він широко застосовується в гідродинаміці, теорії поля, при розрахунку складних напружених станів та в інших областях науки і техніки. Хоча метод кінцевих елементів застосовується для розв'язання тих же самих задач, що і метод кінцевих різниць, але базуються вони на різних ідеях. В методі кінцевих різниць виконується різницева апроксімація похідних, що входять в диференціальне рівняння. В той час, як у методі кінцевих елементів фізична задача замінюється кусочно-гладкою моделлю, приблизно так, як це робиться при інтерполяції функції сплайнами (див. розділ 7.6) тільки в об'ємному вигляді.

Для повного розуміння методу кінцевих елементів необхідні достатньо глибокі знання вищої математики (зокрема, знання варіаційного счислення), тому обмежимось тільки коротким описом його основ. Щоб було більш зрозуміло, слід розглянути метод кінцевих елементів на прикладі дослідження напруженого стану тіла [24].

Основні етапи застосування методу кінцевих елементів наведені на рис. 15.4.

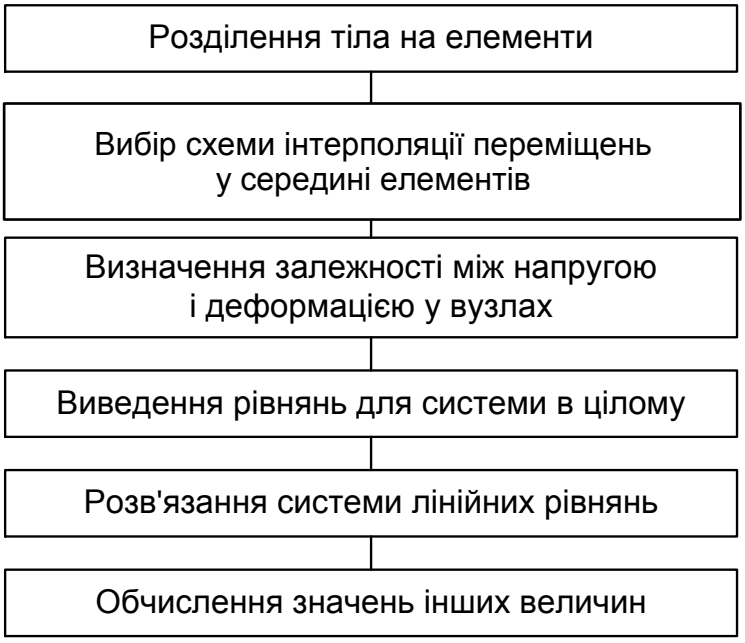


Рис. 15.4. **Етапи чисельного розв'язання задач методом кінцевих елементів**

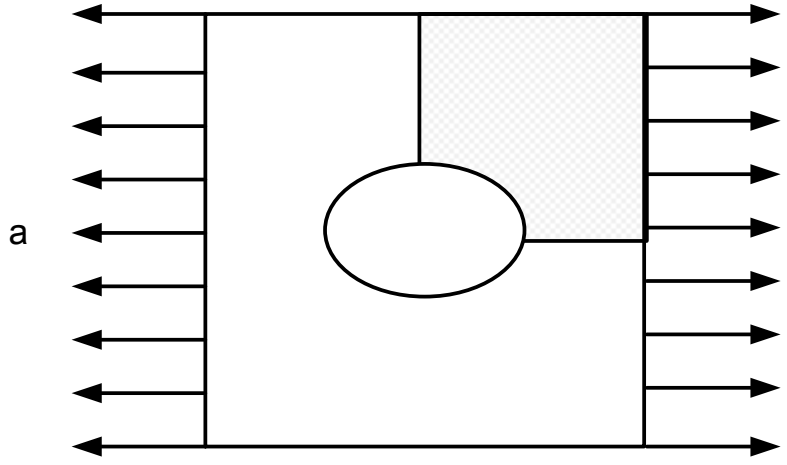
Перший етап полягає в поділі тіла на малі елементи простої форми, що стикаються в точках, які називаються **вузлами**. Поділ на елементи

можна виконати багатьма різними способами, оскільки вибір розмірів, форми й орієнтації елементів цілком визначається уявленнями інженера про те, як простіше розв'язати дану задачу. Елементи плоского тіла мають зазвичай трикутну (в цьому разі цей процес називається **тріангуляцією** – розбиття геометричного об'єкта на симплекси) або чотирикутну форму, а елементи тривимірних тіл – форму тетраедрів або гексаедрів. Ті ділянки тіла, для яких із фізичних міркувань потрібно отримати більш детальну інформацію, розбиваються на більшу кількість дрібних елементів. Якщо фізичні властивості тіла змінюються в точці або вздовж лінії, то можна змінювати форму, розміри або орієнтацію елементів на цій ділянці тіла.

На рис.15.5 показано розбиття рівномірно навантаженої квадратної пластинки з елліптичним отвором у центрі на 26 трикутних кінцевих елементів. Оскільки пластинка має дві осі симетрії, то розглядається тільки одна її чверть. Слід звернути увагу на зменшення розмірів елементів поблизу еліптичного отвору. Це дозволяє отримати більш детальну інформацію про ті ділянки пластинки, на яких великі градієнти напружень.

Як видно з рис. 15.5, зазвичай нумерують і елементи, і вузли, оскільки це полегшує введення відправних даних у програму розрахунку методом кінцевих елементів. Рівномірне навантаження, в даному випадку, моделюється прикладанням зосереджених сил у вузлах 21, 20 і 19.

Другий етап застосування методу скінченних елементів полягає у виборі будь-якої простої схеми інтерполяції, що дозволяє виразити переміщення в будь-якій точці всередині елемента через його значення у вузлах. Зазвичай переміщення задається яким-небудь простим поліномом. У межах кожного елемента для інтерполяції значень переміщення використовуються поліноми з коефіцієнтами, обумовленими в процесі розв'язання.



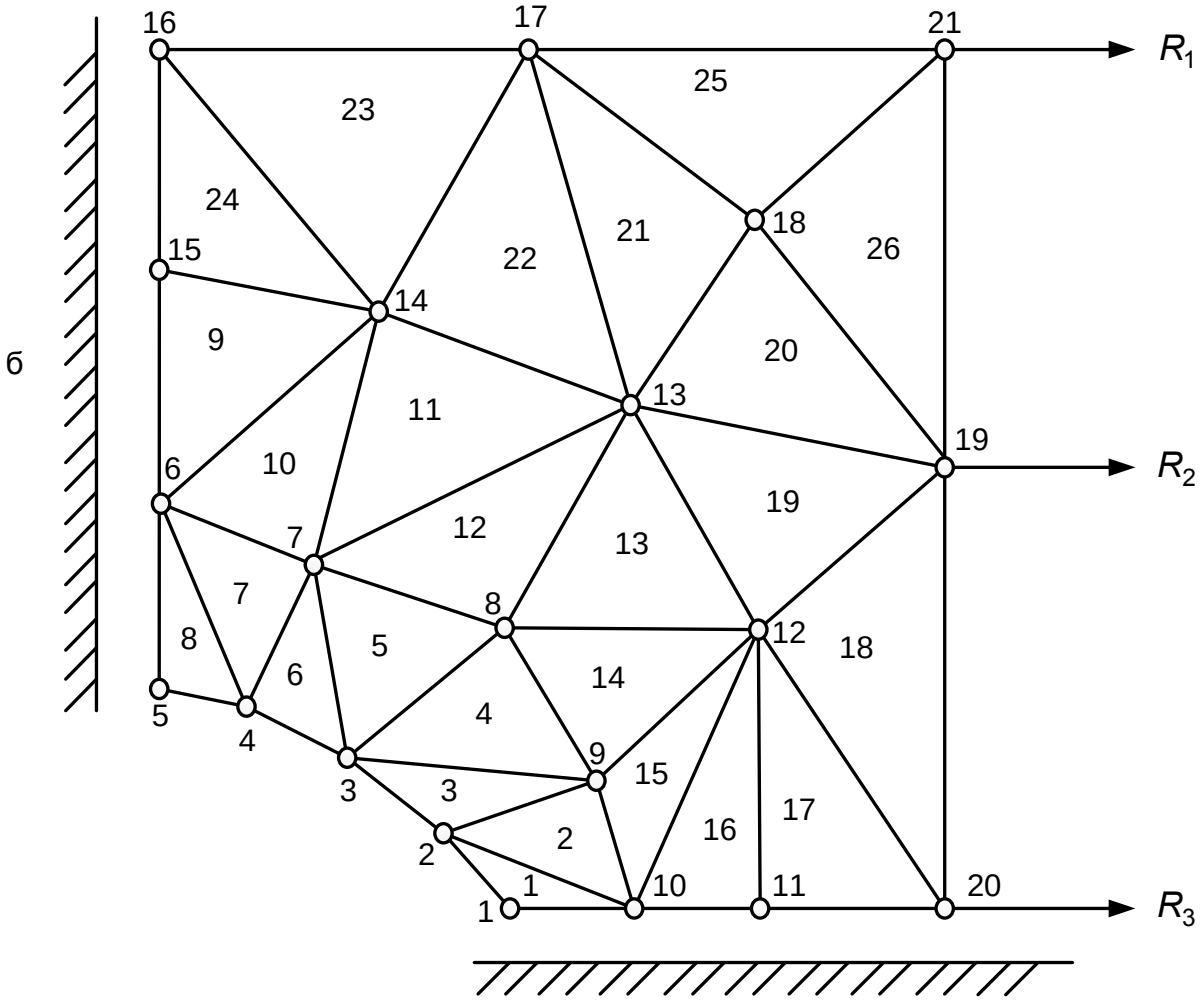


Рис. 15.5. **Розбиття рівномірно навантаженої квадратної пластинки**

На третьому етапі виписуються залежності між напруженнями і деформаціями в вузлах всіх елементів. На цій стадії з великою користю може бути використана концепція матриць і коефіцієнтів впливу. Знаючи співвідношення між напругою і деформаціями елементів, можна побудувати такі ж співвідношення для системи в цілому. При цьому деформації дотичних елементів повинні бути рівні, а сили, що діють у вузлах, повинні складати в сумі зовнішню силу, прикладену в тій же точці. У результаті отримується система лінійних алгебраїчних рівнянь виду:

$$A\,d = b,$$

де A – відома матриця жорсткості системи;

d – шуканий вектор переміщень системи;

b – відомий вектор навантаження.

Отримана система рівнянь містить досить багато нульових елементів, оскільки не кожен вузол належить кожному елементу. У разі довільної деформації кожен з m вузлів може мати n незалежних переміщень (наприклад, по x і y в плоскому випадку). Тому матриця жорсткості A буде мати розмірність $(nm)\times(nm)$, а вектори деформації і сили – розмірність (nm). Деякі значення переміщень безпосередньо визначаються граничними умовами. Відомі значення переміщень можна виключити із системи рівнянь і тим самим знизити її порядок. Оскільки в даному випадку виходить розріджена система лінійних алгебраїчних рівнянь великої розмірності, для її розв'язання зручно користуватися методом Гауса – Зейделя (див. розділ 2.6). У результаті отримують значення переміщень для всіх вузлів. Отримавши розподіл переміщень, можна за допомогою звичайних рівнянь теорії пружності знайти розподіл напружень і деформацій.

15.2. Розв'язання параболічних рівнянь

Розглядається **рівняння теплопровідності** (дифузії):

$$\frac{\partial u}{\partial t} = a^2 \frac{\partial^2 u}{\partial x^2} + f(x,t),\ 0 \le x \le L,\ t \ge 0, \qquad (15.7)$$

де $u(x,t)$ – шукана функція двох змінних (x – координата, t – час);

a – коефіцієнт теплопровідності (якщо u – температура) чи коефіцієнт масоперенесення (якщо u – концентрація речовини);

$f(x,t)$ – функція, яка задає внутрішні джерела тепла (викиду речовини).

Рівняння теплопровідності описує: процеси поширення тепла, процеси дифузії.

Для рівняння (15.7) задаються:
початкова умова (при $t = 0$)

$$u(x,0) = \varphi(x),\ 0 \le x \le L, \qquad (15.8)$$

і граничні умови (при $x = 0$ і $x = L$)

$$u(0,t) = \psi(t),\ u(L,t) = \xi(t),\ t \ge 0. \qquad (15.9)$$

Граничні умови (15.9) також можуть бути задані у вигляді:

$$\frac{\partial u(0,t)}{\partial x} = \psi(x),\ \frac{\partial u(L,t)}{\partial x} = \xi(x), \qquad (15.10)$$

чи комбінації (15.9) і (15.10).

В умовах (15.8) – (15.10) $\varphi(x)$, $\psi(t)$, $\xi(t)$ – задані функції.

Область пошуку розв'язку цієї задачі зображена на рис. 15.6.

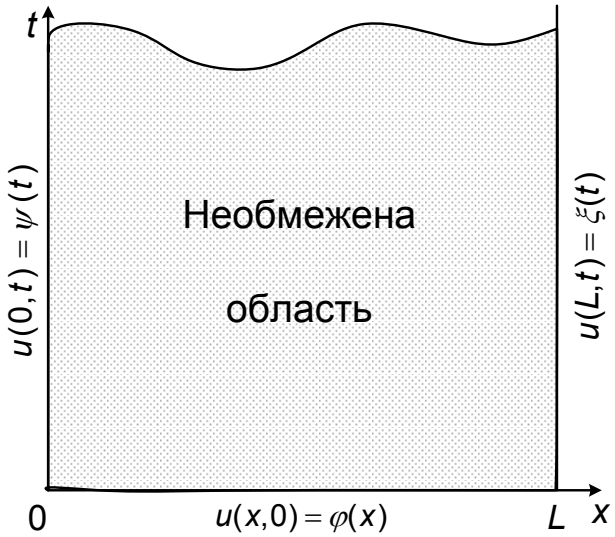


Рис. 15.6. **Область пошуку розв'язку рівняння теплопровідності**

Приклад 15.1. [24]**.** Стержень довжиною L з постійним по довжині перетином занурений в ізолюючий матеріал так, що з навколишнім середовищем взаємодіє тільки його лівий торець (рис. 15.7). У початковий момент часу весь стержень має рівноважну температуру $T = 0$, а його лівий торець стрибком набуває температури $T = 100^{o}C$. Потрібно визначити, як буде змінюватися за часом температура в точках стержня, розташованих на різних відстанях від його лівого торця.

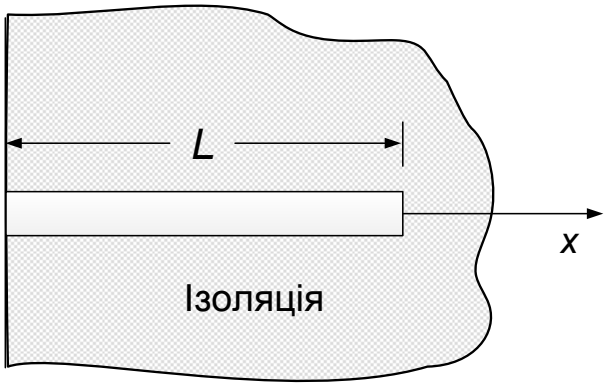


Рис. 15.7. **Стержень з постійним по довжині перетином довжиною L, що занурений в ізолюючий матеріал**

Розв'язання.

Залежність температури $T(x,t)$ від часу і положення по довжині стержня описується рівнянням з частинними похідними (15.7), а з урахуванням відсутності внутрішніх джерел тепла – рівнянням:

$$\frac{\partial T}{\partial t} = a^2 \frac{\partial^2 T}{\partial x^2},$$

де $0 \le x \le L$, $t > 0$,

a – коефіцієнт теплопровідності стержня, що залежить від теплопровідності, питомої теплоємністі та щільності матеріалу, з якого він зроблений.

Граничні умови (при $x = 0$ і $x = L$):

$$T(0,t) = 100 = T_{0+},$$

$$\frac{\partial T(L,t)}{\partial x} = 0$$ (температура не змінюється).

Початкові умови (при $t = 0$):

$$T(x,0) = 0 = T_0.$$

Представивши частинні похідні в різницевому вигляді (15.3), (15.4), маємо:

$$\frac{\partial T}{\partial t} - a^2 \frac{\partial^2 T}{\partial x^2} = \frac{T_{i,j+1} - T_{i,j}}{q} - a^2 \frac{T_{i+1,j} - 2T_{i,j} + T_{i-1,j}}{h^2} = 0. \quad (15.11)$$

Тут розглянута сітка не квадратна, а прямокута: h – крок по змінній x і q – крок по змінній t.

Прийнявши $r = \frac{q a^2}{h^2}$, отримуємо рекурентну (по часу) формулу:

$$T_{i,j+1} = r T_{i+1,j} + (1 - 2r) T_{i,j} + r T_{i-1,j}. \quad (15.12)$$

Цей кінцево-різницевий вираз справедливий для всіх внутрішніх вузлів і дозволяє явним чином виразити температуру в момент часу $t + q$ через температуру в момент часу t. Таким чином, немає необхідності застосовувати ітераційні методи.

Обирається величина $r = \frac{1}{2}$, рівняння (15.12) прийме простіший вигляд:

$$T_{i,j+1} = 0.5(T_{i+1,j} + T_{i-1,j}).$$

Нехай, наприклад, $a = 1$, $L = 1$ і обраний $h = 0.2$, тоді $q = \frac{r\,h^2}{a^2} = 0.02$. На рис. 15.8 показана сітка, що відповідає цим параметрам.

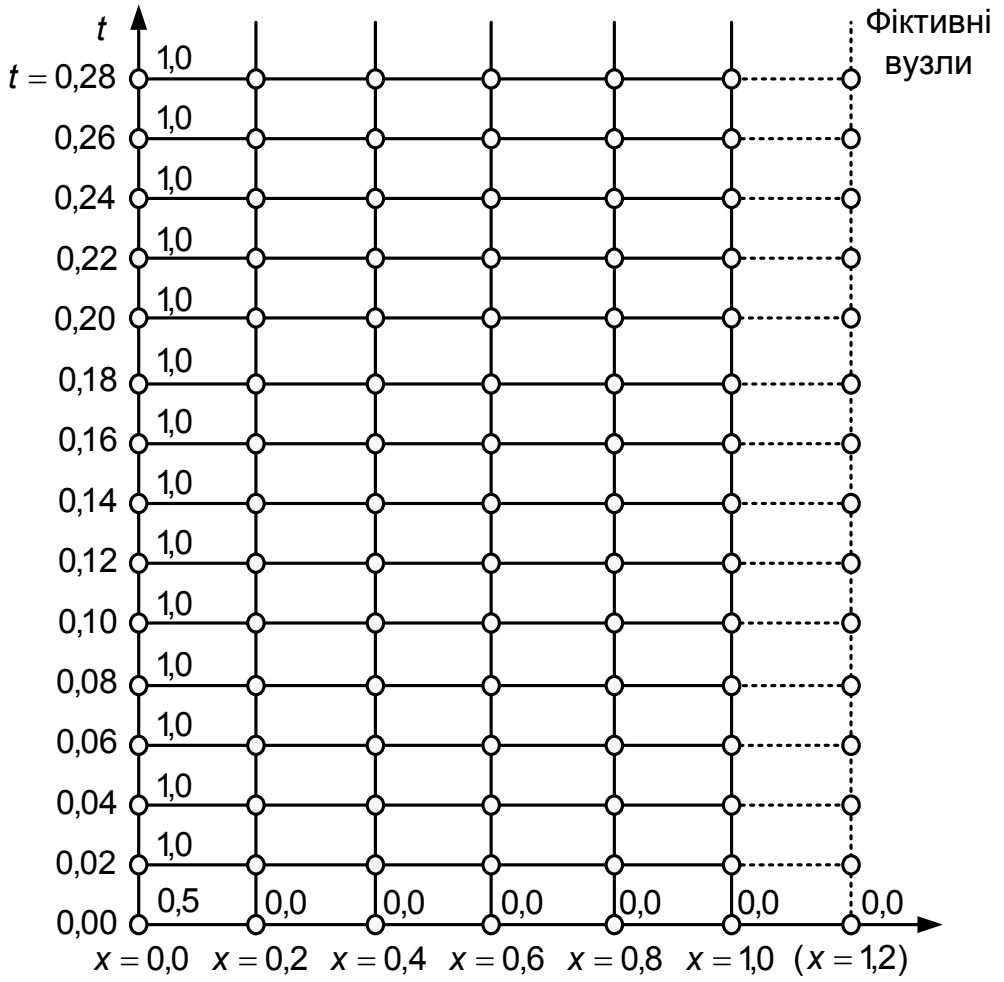


Рис. 15.8. **Сітка для стержня з постійним по довжині перетином довжиною L, що занурений в ізолюючий матеріал**

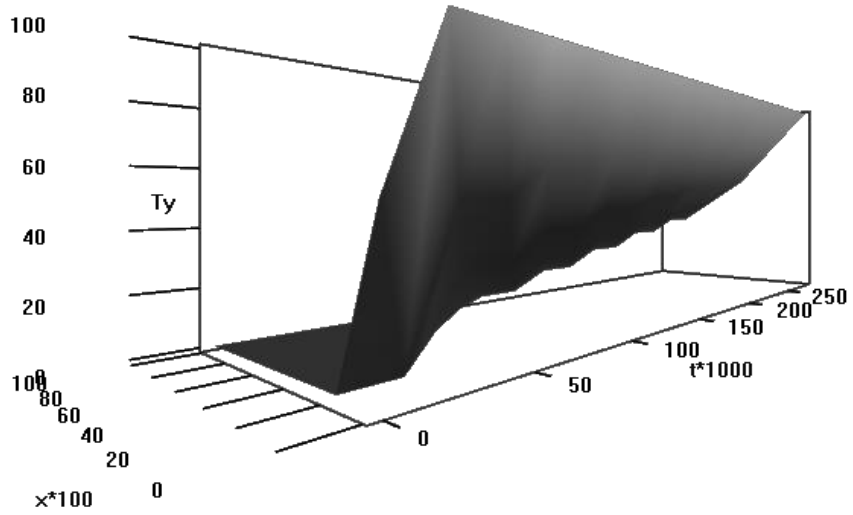
Оскільки гранична умова і початкова умова в точці початку координат терплять розрив, то температурі тут приписується значення $T(0,0) = 50$ – середнє між двома можливими значеннями. Зазвичай саме так чинять при вирішенні подібних завдань. Нульовий нахил дотичної в правій частині стержня моделюється введенням при $x = 1.2$

лінії фіктивних вузлів, температура в яких дорівнює температурі у вузлах при $x = 0.8$. За допомогою цієї сітки і рекурентної формули можна отримати розв'язок для будь-якого моменту часу.

Розв'язання.

```
# процедура розв'язку параболічного диф. рівняння
# методом кінцевих різниць
# Відправні данні задачі
a = 1
L = 1
T0p = 100
T0 = 0

xL = 0
xR = L
tL = 0
tR = 0.28
h = 0.2
q = h*h/(2*a*a)
nx = (xR - xL)/h
nt = (tR - tL)/q
> T = matrix(0, nrow=(nx+2), ncol=(nt+2))
>
> T[1,1] = (T0p + T0)/2
> for (i in 2:(nx+2))
+    T[i,1] = T0
> for (j in 2:(nt+1))
+    T[1,j] = T0p
>
> for (j in 2:nt)
+ {
+    for (i in 2:(nx+1))
+       T[i,j+1] = 0.5*(T[i+1,j] + T[i-1,j])
+    T[nx+2,j+1] = T[nx,j+1]
+ }
> T
     [,1] [,2] [,3] [,4]  [,5]   [,6]    [,7]     [,8]      [,9]     [,10]
[1,]   50  100  100  100 100.0 100.00 100.000 100.0000 100.00000 100.00000
[2,]    0    0   50   50  62.5  62.50  68.750  68.7500  72.65625  72.65625
[3,]    0    0    0   25  25.0  37.50  37.500  45.3125  45.31250  51.17188
[4,]    0    0    0    0  12.5  12.50  21.875  21.8750  29.68750  29.68750
[5,]    0    0    0    0   0.0   6.25   6.250  14.0625  14.06250  21.87500
[6,]    0    0    0    0   0.0   0.00   6.250   6.2500  14.06250  14.06250
[7,]    0    0    0    0   0.0   6.25   6.250  14.0625  14.06250  21.87500
         [,11]     [,12]     [,13]     [,14]     [,15]
[1,] 100.00000 100.00000 100.00000 100.00000 100.00000
[2,]  75.58594  75.58594  78.02734  78.02734  80.16357
[3,]  51.17188  56.05469  56.05469  60.32715  60.32715
[4,]  36.52344  36.52344  42.62695  42.62695  48.12012
[5,]  21.87500  29.19922  29.19922  35.91309  35.91309
[6,]  21.87500  21.87500  29.19922  29.19922  35.91309
[7,]  21.87500  29.19922  29.19922  35.91309  35.91309
```

```
> x1 = seq(xL, xR, by=h)
> t1 = seq(tL, tR, by=q)
> x1 = 100*x1
> t1 = 1000*t1
> # Відрізаються мнимі точки
> T1 = matrix(0, nrow=(nx+1), ncol=(nt+2))
> for (j in 1:(nt+1))
+ {
+    for (i in 1:(nx+1))
+       T1[i,j] = T[i,j]
+ }
> # побудова графіка зміни температури стержня у часі
> library(rgl)
> ylim = range(T1)
> ylen = ylim[2] - ylim[1] + 1
> # height color lookup table
> colorlut = terrain.colors(ylen)
> # assign colors to heights for each point
> col = colorlut[ T1-ylim[1]+1 ]
>
> rgl.surface(x1, t1, T1, coords=1:3, color=col, back="fill")
> axes3d()                    # виведення осей на графік
> title3d('','','x*100','Ty','t*1000') # виведення назв осей
```



15.3. Розв'язання гіперболічних рівнянь

Розглядається **хвилеве рівняння:**

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2} = f(x,t),\ 0 \le x \le L,\ t \ge 0, \qquad (15.13)$$

де $u(x,t)$ – шукана функція двох змінних (x – координата, t – час);

a – швидкість розповсюдження збурювання (u – відхилення від рівноваги);

$f(x,t)$ – функція, яка задає внутрішні джерела енергії.

Хвилеве рівняння описує: малі подовжні коливання стержня, поперечні коливання струни, процеси поширення малих акустичних коливань тощо.

Для рівняння (15.13) задаються:

початкові умови (при $t = 0$)

$$u(x,0) = \varphi_1(x),\ \frac{\partial u(x,0)}{\partial t} = \varphi_2(x),\ 0 \le x \le L \qquad (15.14)$$

і граничні умови (при $x = 0$ і $x = L$)

$$u(0,t) = \psi(t),\ u(L,t) = \xi(t),\ t \ge 0. \qquad (15.15)$$

В умовах (15.14), (15.15) $\varphi_1(x)$, $\varphi_2(x)$, $\psi(t)$, $\xi(t)$ – задані функції.

Граничні умови (15.15) також можуть бути задані у вигляді

$$\frac{\partial u(0,t)}{\partial x} = \psi(x),\ \frac{\partial u(L,t)}{\partial x} = \xi(x), \qquad (15.16)$$

чи комбінації (15.15) і (15.16).

15.4. Розв'язання еліптичних рівнянь

Розглядається **рівняння Пуассона:**

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y),\ (x,y) \in G, \qquad (15.17)$$

де $u(x,y)$ – шукана функція двох змінних (x, y – координати);

G – задана область (тіло);

$f(x,y)$ – функція, яка задає внутрішні джерела енергії (чи відтоків).

Якщо $f(x,y) = 0 \quad \forall (x,y) \in G$, то рівняння (15.17) називається **рівнянням Лапласа**.

Рівняння Лапласа і Пуассона описують: потоки ідеальних рідин в стаціонарних потоках, стаціонарний розподіл температури або напруженості електричних і магнітних полів. Рівняння Лапласа описує ці процеси за відсутності внутрішніх джерел енергії (чи відтоків).

Для рівняння (15.10) задаються лише граничні умови (Γ – межа області G), (рис. 15.9):

1) 1-ша крайова задача (**задача Діріхле**):

$$u\big|_{\Gamma} = \phi(x,y),\ (x,y) \in \Gamma. \tag{15.18}$$

2) 2-га крайова задача (**задача Неймана**):

$$\left.\frac{\partial u}{\partial n}\right|_{\Gamma} = \phi(x,y),\ (x,y) \in \Gamma. \tag{15.19}$$

3) 3-тя крайова задача (комбінована):

$$\alpha u\big|_{\Gamma} + \beta \left.\frac{\partial u}{\partial n}\right|_{\Gamma} = \phi(x,y),\ (x,y) \in \Gamma. \tag{15.20}$$

В умовах (15.19) і (15.20) n – зовнішня нормаль відносно області G в точці $(x,y) \in \Gamma$, $\phi(x,y)$ – задана функція на межі Γ області G.

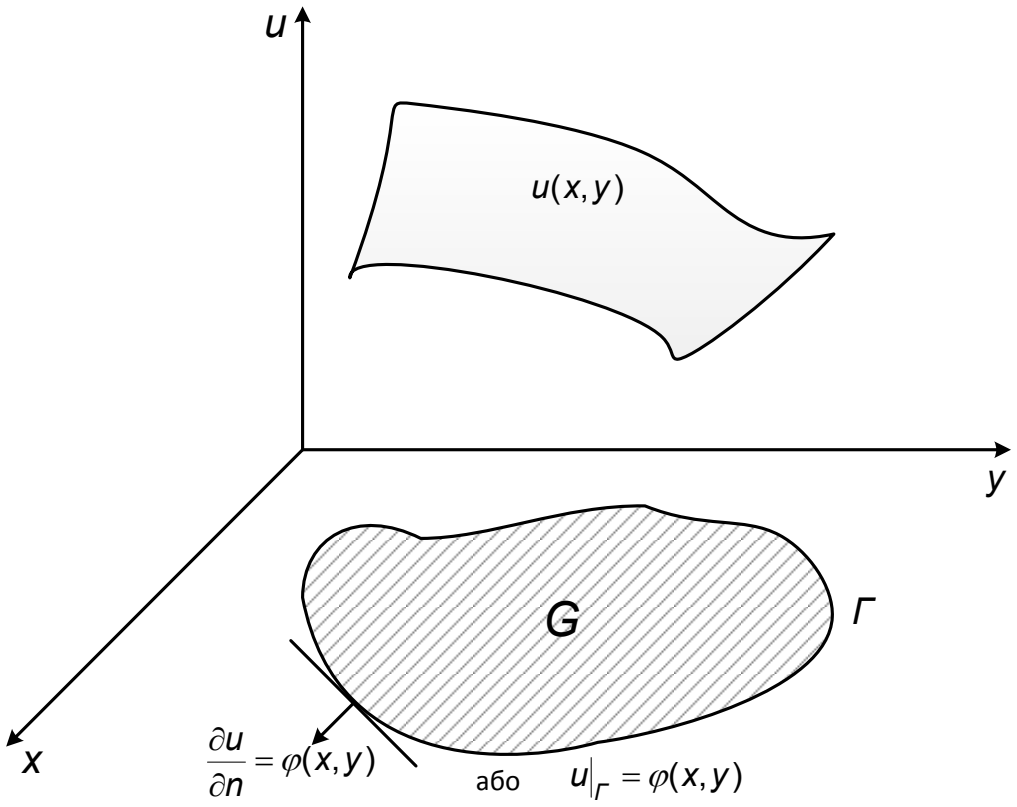


Рис. 15.9. **Розв'язання еліптичного рівняння**

Приклад 15.2 [24]. Вимагається знайти стаціонарний розподіл температури $u(x,y)$ у квадратній однорідній пластині розміру 1×1, для якої задані такі граничні умови: $u(0,y)=0$, $u(1,y)=100$, $u(x,0)=100x$, $u(x,1)=100x^2$. Температура вимірюється у градусах Цельсія.

Розв'язання.

Оскільки процес стаціонарний і немає внутрішніх джерел тепла, то розподіл температури в пластині описується рівнянням Лапласа

$$\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}=0.$$

Для того, щоб сформулювати і розв'язати задачу методом кінцевих різниць введемо на пластині двовимірну сітку з відстанню між вузлами $h=0.25$ (рис. 15.10).

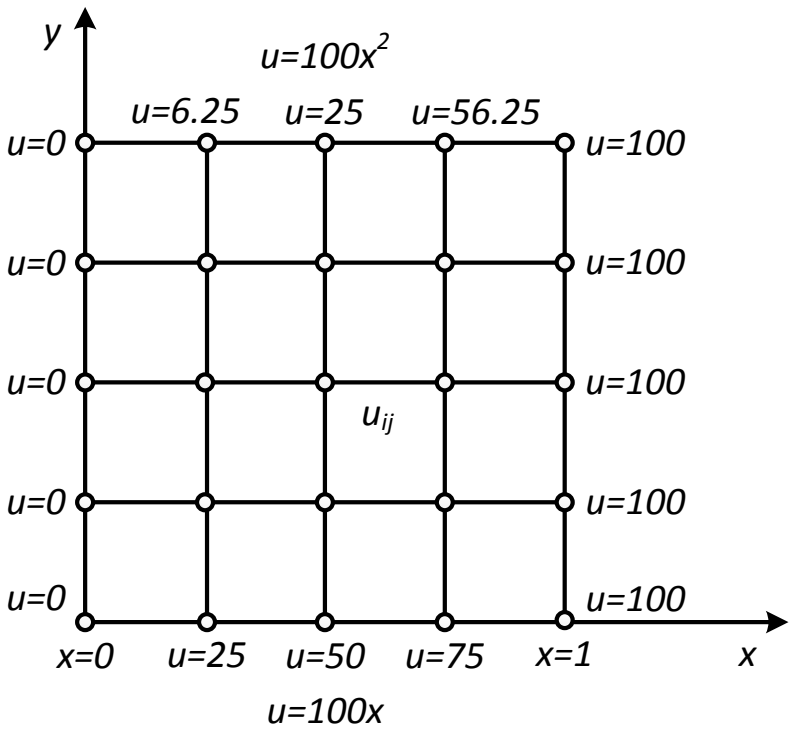


Рис. 15.10. **Двовимірна сітка на пластині**

Сітка складається з 25 вузлів, у 16 з яких температура відома з граничних умов. Таким чином, задача полягає у визначенні температури в 9 внутрішніх вузлах.

Замінюючи у вузлах (i, j) частинні похідні $\frac{\partial^2 u}{\partial x^2}$, $\frac{\partial^2 u}{\partial y^2}$ на симетричні кінцево-різницеві формули:

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}, \frac{\partial^2 u}{\partial y^2} \approx \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2},$$

отримуємо співвідношення:

$$u_{i,j} = \frac{u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}}{4},$$

з якого й отримується формула для обчислень за методом ітерацій:

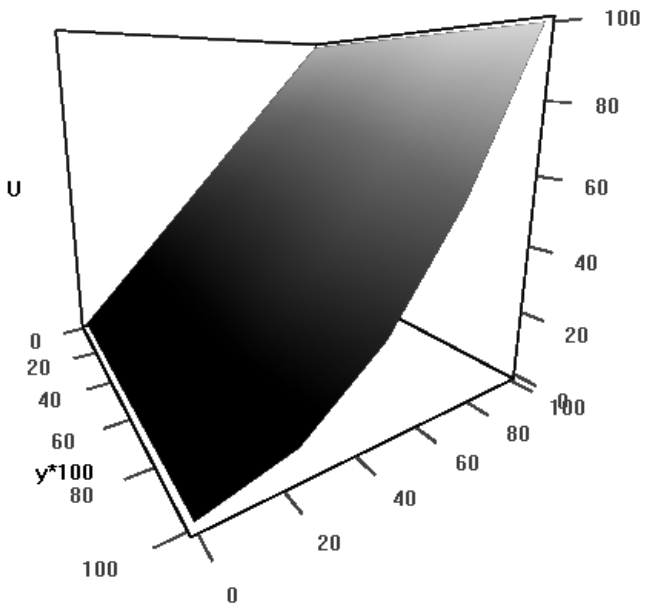
$$u_{i,j}^{(k+1)} = \frac{u_{i-1,j}^{(k)} + u_{i+1,j}^{(k)} + u_{i,j-1}^{(k)} + u_{i,j+1}^{(k)}}{4}.$$

Початкові значення температури $u_{i,j}^{(0)}$ у вузлах (i, j) можна задати за допомогою лінійної інтерполяції.

Розв'язання в математичному пакеті R

```
# процедура розв'язку еліптичного диф. рівняння
# методом кінцевих різниць

# Відправні данні задачі
xL = 0
xR = 1
yL = 0
yR = 1
Fi1 = function(x, y){ return( 0 ) }
Fi2 = function(x, y){ return( 100 ) }
Fi3 = function(x, y){ return( 100*x ) }
Fi4 = function(x, y){ return( 100*x*x ) }

h = 0.25
nx = (xR - xL)/h
ny = (yR - yL)/h
x = seq(xL, xR, by=h)
y = seq(yL, yR, by=h)
```

```
> U = matrix(0, nrow=(nx+1), ncol=(ny+1))
>
> for (i in 1:(nx+1))
+    U[i,1] = Fi3(x[i], y[1])
> for (i in 1:(nx+1))
+    U[i,ny+1] = Fi4(x[i], y[ny+1])
> for (j in 1:(ny+1))
+    U[1,j] = Fi1(x[1], y[j])
> for (j in 1:(ny+1))
+    U[nx+1,j] = Fi2(x[nx+1], y[j])
>
> for (j in 2:ny)
+    for (i in 2:nx)
+       U[i,j] = U[i,1]
> U
     [,1] [,2] [,3] [,4]   [,5]
[1,]    0    0    0    0   0.00
[2,]   25   25   25   25   6.25
[3,]   50   50   50   50  25.00
[4,]   75   75   75   75  56.25
[5,]  100  100  100  100 100.00
> kmax = 100
> for (k in 1:kmax)
+ {
+    for (j in 2:ny)
+     for (i in 2:nx)
+      U[i,j] = (U[i-1,j] + U[i+1,j] + U[i,j-1] + U[i,j+1])/4
+ }
> U
     [,1]       [,2]       [,3]       [,4]   [,5]
[1,]    0   0.00000   0.00000   0.00000   0.00
[2,]   25  23.49330  21.09375  16.35045   6.25
[3,]   50  47.87946  44.53125  38.05804  25.00
[4,]   75  73.49330  71.09375  66.35045  56.25
[5,]  100 100.00000 100.00000 100.00000 100.00
> x
[1] 0.00 0.25 0.50 0.75 1.00
> y
[1] 0.00 0.25 0.50 0.75 1.00
> x = x*100
> y = y*100
> # побудова графіка розподілу температури на пластині
> library(rgl)
> ylim = range(U)
> ylen = ylim[2] - ylim[1] + 1
> colorlut = terrain.colors(ylen) # height color lookup table
> col = colorlut[ U-ylim[1]+1 ]   # assign colors to heights for each point
>
> rgl.surface(x, y, U, coords=1:3, color=col, back="fill")
> axes3d()                  # виведення осей на графік
> title3d('','','x*100','U','y*100') # виведення назв осей
```



15.5. Висновки

1. Для розв'язання диференціальних рівнянь з частинними похідними доводиться вибирати в основному між методом **кінцевих різниць** і методом **кінцевих елементів**.

2. Розмаїття типів і розмірів сіток, видів рівнянь з частинними похідними, можливих кінцево-різницевих апроксимацій цих рівнянь і методів розв'язання отриманих систем алгебраїчних рівнянь роблять задачу чисельного розв'язання рівнянь з частинними похідними методом кінцевих різниць виключно багатогранним дослідженням.

15.6. Контрольні запитання та завдання

1. Запишіть загальний вигляд диференціального рівняння в частинних похідних другого порядку з двома змінними. Які типи класифікації використовуються для цих рівнянь?

2. Наведіть класифікацію диференціальних рівнянь в частинних похідних другого порядку з двома змінними залежно від фізичного сенсу розв'язуваних з їх допомогою задач.

3. Наведіть класифікацію диференціальних рівнянь в частинних похідних другого порядку з двома змінними залежно від їх математичної природи.

4. Сформулюйте постановку задачі Діріхле для рівняння Пуассона.

5. Сформулюйте постановку задачі розв'язання диференціального рівняння в частинних похідних другого порядку з двома змінними для рівняння параболічного типу.

Використана література

1. Амосов А. А. Вычислительные методы для инженеров : учебн. пособ. / А. А. Амосов, Ю. А. Дубинський, Н. В. Копченова. – М. : Высшая школа, 1994. – 544 с.

2. Барахнин В. Б. Введение в численный анализ / В. Б. Барахнин, В. П. Шапеев. – Новосибирск, 1997. – 112 с.

3. Бахвалов Н. С. Численные методы / Н. С. Бахвалов, Н. П. Жидков, Г. М. Кобельков. – М. : Бином, 2007. – 636 с.

4. Боглаев Ю. П. Вычислительная математика и программирование / Ю. П. Боглаев. – М. : Высшая школа, 1990. – 544 с.

5. Волков Е. А. Численные методы / Е. А. Волков. – М. : Высшая школа, 1987. – 312 с.

6. Демидович Б. П. Основы вычислительной математики / Б. П. Демидович. – М. : Наука, 1994. – 664 с.

7. Дэннис Дж. Численные методы безусловной оптимизации и решения нелинейных уравнений / Дж. Дэннис, Р. Шнабель – М. : Мир, 1988. – 40 с.

8. Заварыкин В. М. Численные методы / В. М. Заварыкин, В. Г. Житомирский, М. П. Лапчик. – М. : Просвещение, 1990. – 176 с.

9. Задачин В. М. Робоча програма навчальної дисципліни "Чисельні методи" для студентів напряму підготовки "Комп'ютерні науки" всіх форм навчання / В. М. Задачин, І. Г. Конюшенко. – Х. : Вид. ХНЕУ, 2012. – 40 с.

10. Калиткин Н. Н. Численные методы / Н. Н. Калиткин. – М. : Наука, 1978. – 512 с.

11. Копченова Н. В. Вычислительная математика в примерах и задачах / Н. В. Копченова, И. А. Марон. – М. : Наука, 1972. – 368 с.

12. Марчук Г. И. Методы вычислительной математики / Г. И. Марчук. – М. : Наука,1989. – 608 с.

13. Ортега Дж. Введение в численные методы решения дифференциальных уравнений / Дж. Ортега, У. Пул. – М. : Наука, 1986. – 56 с.

14. Ракитин В. И. Практическое руководство по методам вычислений с приложением программ для персональных компьютеров / В. И. Ракитин, В. Е. Первушин. – М. : Высшая школа, 1998. – 384 с.

15. Самарский А. А. Введение в численные методы / А. А. Самарский. – М. : Наука, 1997. – 240 с.

16. Самарский А. А. Численные методы / А. А. Самарский, А. В. Гулин. – М. : Наука, 1989. – 432 с.

17. Сборник задач по методам вычислений / под ред. П. И. Монастырного. – М. : Наука, 1994. – 320 с.

18. Тейлор Дж. Введение в теорию ошибок / Дж. Тейлор. – М. : Мир, 1985. – 16 с.

19. Турчак Л. И. Основы численных методов / Л. И. Турчак. – М. : Наука, 1997. – 320 с.

20. Федоренко Р. П. Введение в вычислительную физику / Р. П. Федоренко. – М. : Изд. МФТИ, 1994. – 528 с.

21. Фельдман Л. П. Чисельні методи в інформатиці / Л. П. Фельдман, А. І. Петренко, О. А. Дмитрієва – К. : Видавнича група BHV. – 2006. – 480 с.

22. Фурунжиев Р. И. Применение математических методов и ЭВМ. Практикум : учебн. пособ. для вузов / Р. И. Фурунжиев, Ф. М. Бабушкин, В. В. Варавко. – Мн. : Вышейшая школа, 1988. – 192 с.

23. Шикин Е. В. Кривые и поверхности на экране компьютера. Руководство по сплайнам для пользователей / Е. В. Шикин, А. И. Плис. – М. : ДИАЛОГ–МИФИ, 1996. – 50 с.

24. Шуп Т. Решение инженерных задач на ЭВМ: практическое руководство : учебн. изд. / Т. Шуп ; пер. с англ. – М. : Мир, 1982. – 237 с.

25. Everitt B. A handbook of statistical analyses using R / B. Everitt, T. Hothorn. – 2-nd ed. – Chapman and HALL/CRC, 2009. – 376 p.

26. Shumway R. H. Time series analyses and its applications: With R examples / R. H. Shumway, D. S. Stoffer. – 3-rd ed. – New York : Springer, 2011. – 596 p.

27. Zadachyn V. Calculation of optimal path for parallel car parking / V. Zadachyn, O. Dorokhov // Transport and Telecommunication. – Volume 13. – 2012. – pp. 303–309.

Ресурси мережі Інтернет

28. Сайт персональних навчальних систем ХНЕУ ім. С. Кузнеця. – http://www.ikt.hneu.edu.ua/.

29. Quick-R [Electronic resource]. – Access mode : http://www.statmethods.net/index.html.

30. R Site Search [Electronic resource]. – Access mode : http://finzi.psych.upenn.edu/nmz.html.

31. Rtips. Revival 2014! [Electronic resource]. – Access mode : http://pj.freefaculty.org/R/Rtips.html.

32. Statistics with R [Electronic resource]. – Access mode : http://zoonek2.free.fr/UNIX/48_R/all.html.

33. The Comprehensive R Archive Network [Electronic resource]. – Access mode : http://cran.r-project.org.

НАВЧАЛЬНЕ ВИДАННЯ

Задачин Віктор Михайлович
Конюшенко Ірина Григорівна

ЧИСЕЛЬНІ МЕТОДИ

Навчальний посібник

Відповідальний за випуск **Золотарьова І. О.**

Відповідальний редактор **Оленич М. М.**

Редактор **Бутенко В. О.**

Коректор **Маркова Т. А.**

План 2014 р. Поз. № 58-П.
Підп. до друку 23.12.2014 р. Формат 60 х 90 1/16. Папір MultiCopy. Друк Riso.
Ум.-друк. арк. 11,25. Обл.-вид. арк. 14,06. Тираж 400 прим. Зам. № 334.

Видавець і виготівник – видавництво ХНЕУ ім. С. Кузнеця, 61166, м. Харків, пр. Леніна, 9-А

Свідоцтво про внесення до Державного реєстру суб'єктів видавничої справи
Дк № 481 від 13.06.2001 р.