

Міністерство освіти і науки України  
Вінницький національний технічний університет

# **МЕТОДИ ТА АЛГОРИТМИ КОМП'ЮТЕРНИХ ОБЧИСЛЕНЬ**

**Теорія і практика**

Підручник

Вінниця  
ВНТУ  
2023

УДК 004.021+519.6

M54

Рекомендовано до друку Вченою радою Вінницького національного технічного університету Міністерства освіти і науки України (протокол № 6 від 26.01.2023 р.)

**Автори:**

**Р. Н. Кветний, Я. В. Іванчук,  
І. В. Богач, О. Ю. Софіна, М. В. Барабан**

*Рецензенти:*

**В. М. Дубовой**, доктор технічних наук, професор

**В. В. Романюк**, доктор технічних наук, професор

**А. Я. Кулик**, доктор технічних наук, професор

M54 **Методи** та алгоритми комп'ютерних обчислень. Теорія і практика: підручник / Р. Н. Кветний, Я. В. Іванчук, І. В. Богач, О. Ю. Софіна, М. В. Барабан. – Вінниця : ВНТУ, 2023. – 280 с.

ISBN 978-966-641-924-1

У підручнику розглянуто головні застосування задач обчислювальної математики в інженерній практиці з прикладами розв'язання та фрагментами програм на сучасних алгоритмічних мовах C++ та Python. У посібнику наведено обчислювальні задачі, які найчастіше зустрічаються у практиці проектування комп'ютерних систем та інформаційних технологій.

Підручник орієнтовано, в першу чергу, на спеціалістів у галузі технічних наук, тому увага приділялася доведенню всіх викладених методів обчислень до конкретних практичних алгоритмів. Він може привернути увагу студентів та аспірантів спеціальностей, що пов'язані з комп'ютерними науками, кібернетичними системами та інформаційними технологіями.

УДК 004.21+519.6

ISBN 978-966-641-924-1

© ВНТУ, 2023

## ЗМІСТ

ВСТУП.....	5
РОЗДІЛ 1 ЗАДАЧІ ЛІНІЙНОЇ АЛГЕБРИ .....	7
1.1 Розв'язання систем лінійних алгебраїчних рівнянь .....	7
1.2 Прямі методи обчислення .....	9
1.3 Ітераційні методи обчислення .....	29
Контрольні запитання та завдання.....	42
РОЗДІЛ 2 ЗАДАЧІ НЕЛІНІЙНОЇ МАТЕМАТИКИ .....	47
2.1 Узагальнена постановка задачі і процедура локалізації коренів рівняння .....	49
2.2 Чисельні методи розв'язання нелінійних алгебраїчних рівнянь .....	51
2.3 Метод визначення комплексних коренів .....	69
2.4 Чисельні методи розв'язання систем нелінійних алгебраїчних рівнянь .....	74
Контрольні запитання та завдання.....	81
РОЗДІЛ 3. ЗАДАЧІ ДИФЕРЕНЦІАЛЬНОГО ЧИСЛЕННЯ.....	86
3.1 Узагальнена постановка задачі для звичайних диференціальних рівнянь.....	90
3.2 Чисельні методи розв'язання звичайних диференціальних рівнянь для задач типу Коші .....	90
3.3 Чисельні методи розв'язання звичайних диференціальних рівнянь для крайових задач .....	132
3.4 Чисельні методи розв'язання звичайних диференціальних рівнянь для «жорстких» задач.....	147
Контрольні запитання та завдання.....	155
РОЗДІЛ 4. ДИФЕРЕНЦІАЛЬНІ РІВНЯННЯ МАТЕМАТИЧНОЇ ФІЗИКИ.....	162
4.1 Класифікація диференціальних рівнянь у частинних похідних.....	163
4.2 Метод кінцевих різниць .....	166
4.3 Розв'язання різних видів диференціальних рівнянь в частинних похідних.....	168
Контрольні запитання та завдання.....	194

РОЗДІЛ 5 ЗАДАЧІ ОБРОБКИ ДАНИХ.....	199
5.1 Інтерполяція .....	199
5.1.1 Різницеві методи.....	200
5.1.2 Інтерполяція за Лагранжем.....	210
5.1.3 Інтерполяція функцій двох змінних за методом Лагранжа.....	215
5.1.4 Сплайн-інтерполяція .....	221
5.2 Апроксимація даних .....	228
5.3 Статистична обробка даних.....	234
Контрольні запитання та завдання.....	240
 РОЗДІЛ 6 ЧИСЕЛЬНЕ ІНТЕГРУВАННЯ ТА ДИФЕРЕНЦІЮВАННЯ....	245
6.1 Метод прямокутників.....	245
6.2 Формули Ньютона – Котеса .....	248
6.3 Формула Чебишова.....	251
6.4 Формула Гаусса.....	255
6.5 Алгоритми застосування чисельних методів.....	260
6.6 Метод Монте-Карло .....	266
6.7 Оцінення похибки чисельного інтегрування .....	268
6.8 Чисельне диференціювання .....	269
6.8.1 Чисельне диференціювання аналітично заданих функцій.....	269
6.8.2 Чисельне диференціювання експериментальних даних .....	271
Контрольні запитання та завдання.....	274
 ЛІТЕРАТУРА.....	279

## ВСТУП

Цей підручник має за мету викладення найбільш поширених в інженерній практиці задач обчислювальної математики з прикладами розв'язання та фрагментами програм сучасними алгоритмічними мовами C++ та Python. Автори намагалися узагальнити досвід багаторічного викладання курсів з комп'ютерної математики для студентів та аспірантів спеціальностей, пов'язаних з комп'ютерними науками, кібернетичними системами та інформаційними технологіями. Існує велика кількість фундаментальних підручників та посібників з обчислювальної математики. Вони слугують методологічною основою цього підручника.

Підручник складається зі вступу та шести розділів:

Вступ (підготовлено Р. Кветним).

1. Задачі лінійної алгебри (підготовлено О. Софіною з використанням матеріалів Р. Кветного, частину прикладів надано І. Богач та М. Барабан).

2. Задачі нелінійної математики (підготовлено Я. Іванчуком з використанням матеріалів Р. Кветного).

3. Задачі диференціального числення (підготовлено Я. Іванчуком з використанням матеріалів Р. Кветного).

4. Диференціальні рівняння математичної фізики (підготовлено Я. Іванчуком з використанням матеріалів Р. Кветного).

5. Задачі обробки даних (підготовлено І. Богач з використанням матеріалів Р. Кветного, частину прикладів надано О. Софіною та М. Барабан).

6. Чисельне інтегрування та диференціювання (підготовлено І. Богач з використанням матеріалів Р. Кветного, частину прикладів надано О. Софіною та М. Барабан).

Ідея написання підручника належить Р. Кветному. Ним же здійснено загальну редакцію підручника.

Автори обмежилися обчислювальними задачами, які найчастіше зустрічаються у практиці проектування комп'ютерних систем та інформаційних технологій. Причому акцент ставився саме на алгоритмізації процесу розв'язання задач. Підручник орієнтовано, насамперед, на спеціалістів в галузі технічних наук, тому увага приділялася доведенню всіх викладених методів обчислень до конкретних практичних алгоритмів. Отже деякі методи та формули надано без виведення, яке можна знайти у спеціальній літературі. Наведені фрагменти програм можуть бути легко інтерпретовані за мінімальної підготовки в сфері, галузі програмування. Різні розділи склалися різними авторами, але їх стиль викладення було узагальнено під час редагування. Уподобання та завдання користувачів в галузі програмування та комп'ютерних обчислень бувають різними, тому автори не зупинилися на одній мові програмування і надали ілюстративні приклади традиційною мовою C++ та популярною мовою сьогодення Python. Підручник орієнтовано на науковців та інженерів у галузі

комп'ютерних наук, інформаційних технологій, робототехнічних комплексів та комп'ютерно-інтегрованих систем автоматизації та управління, також він буде корисним студентам та аспірантам усіх спеціальностей галузей «Інформаційні технології» та «Електроніка та телекомунікації».

У кінці підручника додаємо перелік навчальної літератури, якою можна скористатися під час вивчення методів комп'ютерних обчислень. Поряд із виданими авторами та вітчизняними колегами посібниками та підручниками наведено базові відомі світові англійськомовні джерела, які доступні сьогодні у повному обсязі в інформаційному просторі.

## РОЗДІЛ 1 ЗАДАЧІ ЛІНІЙНОЇ АЛГЕБРИ

У цьому розділі розглянуто розв'язання однієї з найбільш поширених обчислювальних задач лінійної математики – розв'язання **систем лінійних алгебраїчних рівнянь** (СЛАР). Причому вважається, що користувач вже знайомий з головними відомостями з теорії матриць.

Дослідження багатьох фізичних систем призводить до математичних моделей у формі СЛАР, які можуть з'являтися у процесі математичного моделювання як проміжний етап під час вирішення складнішого завдання. Існує значна кількість науково-технічних завдань, у яких математичні моделі складних нелінійних систем, у формі дискретизації чи лінеаризації, зводяться до розв'язання СЛАР.

Приклади завдань, які використовують математичні моделі у формі СЛАР:

- під час моделювання економічних задач, таких як задача управління та планування виробництва, визначення оптимального розміщення обладнання, оптимального плану виробництва, оптимального плану перевезень вантажів, розподілу кадрів тощо, може бути покладена гіпотеза лінійного подання реального світу. Математичні моделі таких задач описуються системою лінійних рівнянь;

- під час проектування та експлуатації електротехнічних пристроїв необхідне проведення розрахунку та аналізу їх роботи у стаціонарних режимах. Задача зводиться до розрахунку еквівалентних схем, в основі якої лежить формування та розв'язання СЛАР;

- під час побудови математичної моделі, яка зв'язує функціонально залежністю деякі параметри  $x_i$  та  $y_i$  досліджуваного об'єкта на підставі отриманих внаслідок експерименту даних, де  $i = 1, 2, \dots, n$  (задання апроксимації даних);

- для дослідження фізичних процесів у складних системах, математичні моделі яких будуються на основі диференціальних рівнянь у частинних похідних. Внаслідок різницевої апроксимації вихідної моделі за певних умов приходять до математичних співвідношень у формі СЛАР;

- суть багатьох фізичних процесів математично відображається за допомогою інтегральних рівнянь. Зважаючи на складність розв'язання багатьох з них, досліднику краще звести задачу до розв'язання математичної моделі у формі СЛАР, використовуючи відомі методи апроксимації;

- дослідження систем автоматичного регулювання в усталеному режимі призводить, у багатьох випадках, до статичних моделей у формі СЛАР.

### 1.1 Розв'язання систем лінійних алгебраїчних рівнянь

Постановкою задачі розв'язання СЛАР є визначення невідомих значень  $x_1, x_2, \dots, x_n$ , що задовольняють систему з  $m$  лінійних алгебраїчних рівнянь (*system of linear algebraic equations*):





паралельними або збігатися. Тому будь-яка СЛАР може мати: єдиний розв'язок; нескінченну множину розв'язків; не мати розв'язку взагалі.

Необхідною і достатньою умовою існування єдиного розв'язку квадратної СЛАР (1.1) є нерівність нулю визначника  $A$  (лінійна незалежність рівнянь):

$$\det A = \begin{vmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{vmatrix} \neq 0.$$

Методи розв'язання СЛАР можуть бути поділені на прямі та ітераційні. До прямих належать методи, які дозволяють одержати точний розв'язок (методи Крамера, Гаусса, прогонки). До ітераційних відносяться методи, що ґрунтуються на одержанні й уточненні послідовних наближень до точного розв'язку. Ітераційні методи ефективні в тому випадку, коли є багато нульових коефіцієнтів або високий порядок системи (метод Гаусса ефективний до порядку  $10^4$ , ітераційні – до  $10^6$ ).

## 1.2 Прямі методи обчислення

До найбільш популярних прямих методів відносяться метод Гаусса та його різновиди, метод Крамера (визначників), метод оберненої матриці, а також метод прогонки, що використовується в задачах із діагональними матрицями. Проте, метод Крамера (визначників), який детально розглянуто у стандартних курсах вищої математики, не може бути застосований у більшості практичних задач з причини великої складності розрахунку визначників навіть у разі невеликого зростання порядку системи. Тому в цьому розділі буде зосереджено увагу на розгляді методу Гаусса, який, якщо й поступається ітераційним методам у певних практичних областях, все ж таки є найбільш універсальним. Також буде розглянуто метод прогонки, що використовується в задачах із діагональними матрицями.

### Метод Гаусса

Цей метод є одним із найпоширеніших методів розв'язання СЛАР. У його основі лежить ідея послідовного виключення невідомих аргументів, що приводить вихідну систему до трикутного вигляду, у якому всі коефіцієнти нижче головної діагоналі дорівнюють нулю. Існують різні обчислювальні схеми, що реалізують цей метод. Найбільше поширення мають схеми з вибором головного елемента по рядку, по стовпцю або по всій матриці.

Класичний метод Гаусса (метод виключення) ґрунтується на зведенні матриці  $A$  коефіцієнтів системи (1.1) до трикутного вигляду:

$$\begin{pmatrix} * & * & * & \dots & * & * \\ 0 & * & * & \dots & * & * \\ 0 & 0 & * & \dots & * & * \\ \vdots & \vdots & \vdots & \dots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & * & * \\ 0 & 0 & 0 & \dots & 0 & * \end{pmatrix}$$

і складається з двох етапів: прямого ходу і оберненої підстановки. Етап прямого ходу закінчується, коли одне з рівнянь системи стає рівнянням з одним невідомим. Далі, здійснюючи обернену підстановку, визначають усі невідомі значення  $x_{ij}$ . З метою зменшення обчислювальної похибки використовують метод Гаусса із вибором головного елемента. Під головним елементом будемо розуміти максимальний за модулем елемент  $a_{ij}^{\max}$  матриці  $A$ , вибраний на заданій множині рядків та стовпців. Алгоритм методу такий. Спочатку знаходиться головний елемент  $a_{ij}^{\max}$  матриці  $A$  і шляхом перестановки відповідних рівнянь та стовпців він розміщується на місце елемента  $a_{11}$  (перестановку стовпців необхідно запам'ятовувати для правильного зіставлення змінних з отриманими значеннями). Потім перше рівняння нормується за допомогою його ділення на  $a_{11} = a_{ij}^{\max}$ :

$$x_1 + \frac{a_{12}}{a_{ij}^{\max}} x_2 + \dots + \frac{a_{1n}}{a_{ij}^{\max}} x_n = \frac{b_1}{a_{ij}^{\max}}. \quad (1.2)$$

Помноживши послідовно (1.2) на  $a_{21}$ ,  $a_{31}$ , ...,  $a_{n1}$  та віднявши відповідно з 2-го, ...,  $n$ -го рівняння системи (1.1), отримаємо СЛАР типу  $A^1 X = B^1$ :

$$\begin{cases} x_1 + a_{12}^1 x_2 + \dots + a_{1n}^1 x_n = b_1^1; \\ 0 + a_{22}^1 x_2 + \dots + a_{2n}^1 x_n = b_2^1; \\ \dots \dots \dots; \\ 0 + a_{n2}^1 x_2 + \dots + a_{nn}^1 x_n = b_n^1, \end{cases}$$

де  $a_{ij}^1 = a_{ij} - a_{1j}^1 \cdot a_{i1}$ ,  $b_i^1 = b_i - b_1^1 \cdot a_{i1} \quad \forall (i = \overline{2, n} \text{ і } j = \overline{2, n})$ ;  $a_{1j}^1 = a_{1n} / a_{ij}^{\max}$ ;  $b_1^1 = b_1 / a_{ij}^{\max}$ .

Нехай на деякому кроці буде отримано матриці  $A^k$  і  $B^k$  (початкові матриці відповідають  $A^0$  і  $B^0$ ), а діагональний елемент  $a_{k+1, k+1}^k$  – максимальний за

модулем елемент матриці  $A^k$  для рядків  $i > k$  і стовпців  $j > k$  (за необхідності переставляються відповідні рівняння і стовпці).

Наведемо формули для розрахунку матриць  $A^{k+1}$  і  $B^{k+1}$ :

$$a_{ij}^{k+1} = \begin{cases} a_{ij}^k & (i \leq k \text{ або } j \leq k); \\ m_{i,j}^k = a_{k+1,j}^k / a_{i,k+1}^k & (i = k+1, j > k); \\ a_{ij}^k - m_{i,j}^k \cdot a_{k+1,j}^{k+1} & (i > k+1, j > k), \end{cases} \quad (1.3)$$

$$b_i^{k+1} = \begin{cases} b_i^k & (i \leq k); \\ b_{k+1}^k / a_{k+1,k+1}^k & (i = k+1); \\ b_i^k - b_{k+1}^{k+1} \cdot a_{i,k+1}^k & (i > k+1). \end{cases} \quad (1.4)$$

Під час програмування методу для збереження інформації значень коефіцієнтів матриць достатньо використовувати один двовимірний масив з  $n$  рядками та  $n + 1$  стовпцем, в якому розміщується розширена матриця системи  $(A|B)$ .

### Умови завершення прямого ходу методу Гаусса

В процесі виконання розрахунків може виникнути проблема:

$$\text{елемент } a_{k+1,k+1}^k = 0.$$

Ця ситуація виникає, коли всі елементи матриці  $A^k$  у рядках  $i > k$  дорівнюють нулю. Система в цьому випадку має вигляд:

$$\left\{ \begin{array}{l} x_1 + a_{12}^k x_2 + \dots + a_{1n}^k x_n = b_1^k; \\ 0 + a_{22}^k x_2 + \dots + a_{2n}^k x_n = b_2^k; \\ \dots; \\ 0 + \dots 0x_{k-1} + a_{kk}^k x_k + \dots + a_{kn}^k x_n = b_k^k; \\ 0 + \dots 0x_{k-1} + 0x_k + 0x_{k+1} + \dots + 0x_n = b_{k+1}^k; \\ \dots; \\ 0 + \dots 0x_{k-1} + 0x_k + 0x_{k+1} + \dots + 0x_n = b_n^k. \end{array} \right.$$

Якщо застосування формул (1.3) і (1.4) неможливо, то система має нескінченну множину розв'язків або взагалі їх немає, що може бути визначено із матриці  $B^k$ .

Якщо усі елементи  $b_i^k = 0$  за умови  $i \geq k+1$ , то система має нескінченну множину розв'язків, причому корені  $x_1, \dots, x_k$  називаються залежними і виражаються через значення  $x_{k+1}, \dots, x_n$ , які називаються незалежними. Якщо хоч один елемент  $b_i^k \neq 0$  за  $i \geq k+1$ , то розв'язків у системи немає.

За відсутності випадку проблеми ділення на нуль і отримання трикутної матриці  $A^n$  – система має єдиний розв'язок. Тоді для його пошуку застосовується зворотний хід методу Гаусса:

$$\begin{cases} x_n = b_n^n; \\ x_{n-i} = b_{n-i}^n - \sum_{k=1}^i a_{n-k+1}^n \cdot x_{n-k+1} \quad (i = \overline{1, n-1}). \end{cases} \quad (1.5)$$

**Приклад 1.1.** Розв'язати СЛАР методом Гаусса із вибором головного елемента:

$$\begin{cases} 5 \cdot x_1 + 6 \cdot x_2 + 7 \cdot x_3 + 8 \cdot x_4 = 1; \\ 10 \cdot x_1 + 10 \cdot x_2 + 11 \cdot x_3 + 12 \cdot x_4 = 2; \\ 15 \cdot x_1 + 4 \cdot x_2 - 3 \cdot x_3 + 5 \cdot x_4 = 3; \\ 2 \cdot x_1 + 20 \cdot x_3 - 2 \cdot x_4 = 4. \end{cases}$$

*Розв'язання:*

Формується матриця коефіцієнтів  $A$  і стовпець вільних членів  $B$ :

$$A = \begin{pmatrix} 5 & 6 & 7 & 8 \\ 10 & 10 & 11 & 12 \\ 15 & 4 & -3 & 5 \\ 2 & 0 & 20 & -2 \end{pmatrix}; \quad B = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}.$$

Головний елемент матриці  $A$  дорівнює  $a_{13}^{\max} = a_{43} = 20$ . Після перестановки рядків буде отримано нове значення  $a_{11}$ , а саме  $a_{11}=20$ , і значення нових матриць:

$$A = \begin{pmatrix} 2 & 0 & 20 & -2 \\ 10 & 10 & 11 & 12 \\ 15 & 4 & -3 & 5 \\ 5 & 6 & 7 & 8 \end{pmatrix}; \quad B = \begin{pmatrix} 4 \\ 2 \\ 3 \\ 1 \end{pmatrix}.$$

Розрахуємо елементи матриці  $A^1$  і стовпця  $B^1$ :

$$\left( \begin{array}{cccc|c} x_1 & x_2 & x_3 & x_4 & b \\ 2 & 0 & 20 & -2 & 4 \\ 10 & 10 & 11 & 12 & 2 \\ 15 & 4 & -3 & 5 & 3 \\ 5 & 6 & 7 & 8 & 1 \end{array} \right) = \left( \begin{array}{cccc|c} x_3 & x_2 & x_1 & x_4 & b \\ 20 & 0 & 2 & -2 & 4 \\ 11 & 10 & 10 & 12 & 2 \\ -3 & 4 & 15 & 5 & 3 \\ 7 & 6 & 5 & 8 & 1 \end{array} \right) =$$

$$= \left( \begin{array}{cccc|c} x_3 & x_2 & x_1 & x_4 & b \\ 20/20 & 0/20 & 2/20 & -2/20 & 4/20 \\ 11 & 10 & 10 & 12 & 2 \\ -3 & 4 & 15 & 5 & 3 \\ 7 & 6 & 5 & 8 & 1 \end{array} \right) = \left( \begin{array}{cccc|c} x_3 & x_2 & x_1 & x_4 & b \\ 1 & 0 & 0,1 & -0,1 & 0,2 \\ 11 & 10 & 10,0 & 12,0 & 2,0 \\ -3 & 4 & 15,0 & 5,0 & 3,0 \\ 7 & 6 & 5,0 & 8,0 & 1,0 \end{array} \right) =$$

$$= \left( \begin{array}{cccc|c} x_3 & x_2 & x_1 & x_4 & b \\ 1 & 0 & 0,1 & -0,1 & 0,2 \\ 11-(1 \cdot 11) & 10-(0 \cdot 11) & 10-(0,1 \cdot 11) & 12-(-0,1 \cdot 11) & 2-(0,2 \cdot 11) \\ -3-(1 \cdot -3) & 4-(0 \cdot -3) & 15-(0,1 \cdot -3) & 5-(-0,1 \cdot -3) & 3-(0,2 \cdot -3) \\ 7-(1 \cdot 7) & 6-(0 \cdot 7) & 5-(0,1 \cdot 7) & 8-(-0,1 \cdot 7) & 1-(0,2 \cdot 7) \end{array} \right) =$$

$$= \left( \begin{array}{cccc|c} x_3 & x_2 & x_1 & x_4 & b \\ 1 & 0 & 0,1 & -0,1 & 0,2 \\ 0 & 10 & 8,9 & 13,1 & -0,2 \\ 0 & 4 & 15,3 & 4,7 & 3,6 \\ 0 & 6 & 4,3 & 8,7 & -0,4 \end{array} \right)$$

$$A^1 = \left( \begin{array}{cccc} x_3 & x_2 & x_1 & x_4 \\ 1 & 0 & 0,1 & -0,1 \\ 0 & 10,0 & 8,9 & 13,1 \\ 0 & 4,0 & 15,3 & 4,7 \\ 0 & 6,0 & 4,3 & 8,7 \end{array} \right); \quad B^1 = \left( \begin{array}{c} 0,2 \\ -0,2 \\ 3,6 \\ -0,4 \end{array} \right)$$

Головний елемент нової матриці  $a_{33}^{\max} = a_{33}^1 = 15,3$ .

Аналогічно обчислюються елементи матриці  $A^2$  і стовпця  $B^2$ :

$$A^2 = \begin{pmatrix} x_3 & x_1 & x_2 & x_4 \\ 1 & 0,1 & 0,000 & -0,100 \\ 0 & 1,0 & 0,260 & 0,310 \\ 0 & 0,0 & 7,677 & 10,368 \\ 0 & 0,0 & 4,878 & 7,380 \end{pmatrix}; B^2 = \begin{pmatrix} 0,200 \\ 0,240 \\ -2,292 \\ -1,411 \end{pmatrix}.$$

Головний елемент нової матриці  $a_{34}^{\max} = a_{34}^2 = 10,368$ .

Аналогічно обчислюються елементи матриці  $A^3$  і стовпця  $B^3$ :

$$A^3 = \begin{pmatrix} x_3 & x_1 & x_4 & x_2 \\ 1 & 0,1 & -0,10 & 0,000 \\ 0 & 1,0 & 0,310 & 0,260 \\ 0 & 0,0 & 1,000 & 0,740 \\ 0 & 0,0 & 0,000 & -0,598 \end{pmatrix}; B^3 = \begin{pmatrix} 0,200 \\ 0,240 \\ -0,23 \\ 0,220 \end{pmatrix}.$$

Матриця  $A^3$  має трикутний вигляд. Система має єдиний розв'язок. Для його пошуку застосовується зворотний хід метода Гаусса:

$$x_2 = b_4^3 / a_{44}^3 = 0,220 / -0,598 = -0,377;$$

$$x_4 = b_3^3 - a_{34}^3 \cdot x_2 = -0,23 - 0,74 \cdot -0,377 = 0,058;$$

$$x_1 = b_2^3 - a_{24}^3 \cdot x_2 - a_{23}^3 \cdot x_4 = 0,24 - (0,26 \cdot -0,377) - (0,310 \cdot 0,058) = 0,316;$$

$$x_3 = b_1^3 - a_{14}^3 \cdot x_2 - a_{13}^3 \cdot x_4 - a_{12}^3 \cdot x_1 = \\ = 0,2 - (0 \cdot -0,377) - (-0,10 \cdot 0,058) - (0,1 \cdot 0,316) = 0,174.$$

$$\text{Розв'язок: } X = \begin{pmatrix} 0,316 \\ -0,377 \\ 0,174 \\ 0,058 \end{pmatrix}.$$

Алгоритм методу подано на рисунку 1.1.

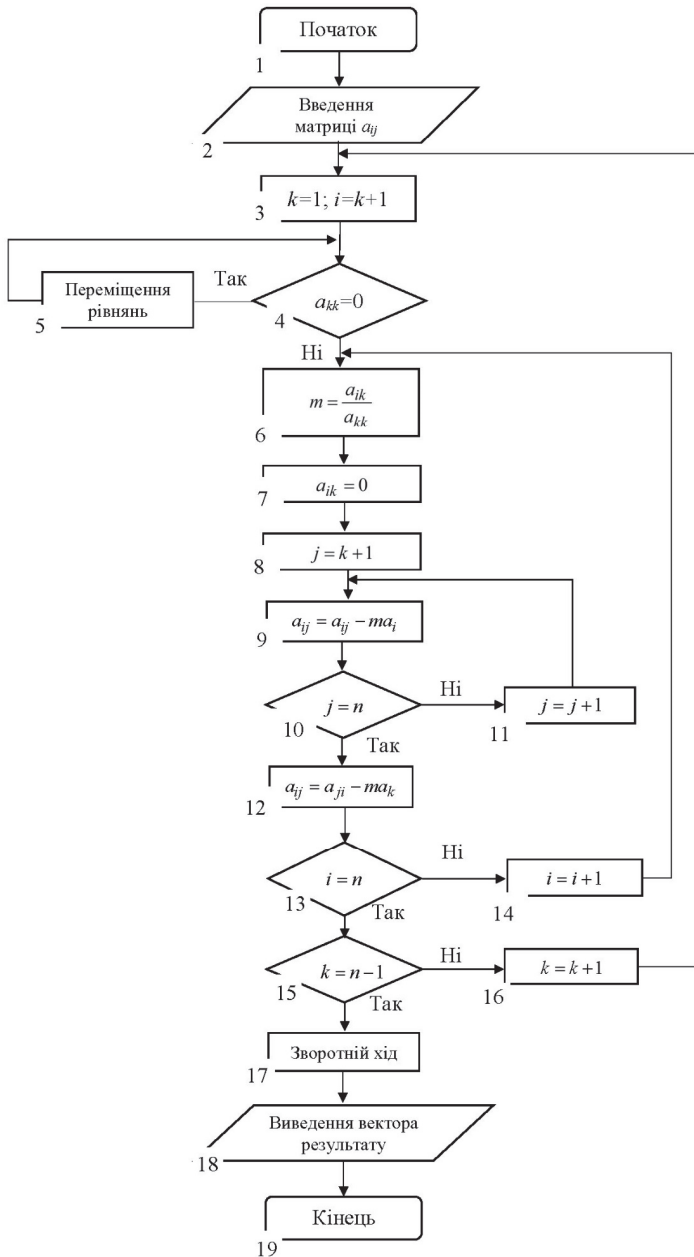


Рисунок 1.1 – Схема алгоритму метода Гаусса

Розглянемо реалізацію методу Гаусса мовою програмування C++:

```
int main()
{
    int i, j, n, m;
    //створюємо масив
    cout << "Number of equations: ";
    cin >> n;
    cout << "Number of variables: ";
    cin >> m;
    m += 1;
    float **matrix = new float *[n];
    for (i = 0; i<n; i++)
        matrix[i] = new float[m];
    for (i = 0; i<n; i++)
        for (j = 0; j<m; j++)
        {
            cout << " Element " << "[" << i + 1 << " , " << j + 1 << "]: ";
            cin >> matrix[i][j];
        }
    //виводимо масив
    cout << "matrix: " << endl;
    for (i = 0; i<n; i++)
    {
        for (j = 0; j<m; j++)
            cout << matrix[i][j] << " ";
        cout << endl;
    }
    cout << endl;
    //Метод Гаусса, прямий хід
    float tmp;
    int k;
    float *xx = new float[m];
    for (i = 0; i<n; i++)
    {
        tmp = matrix[i][i];
        for (j = n; j >= i; j--)
            matrix[i][j] /= tmp;
        for (j = i + 1; j<n; j++)
        {
            tmp = matrix[j][i];
            for (k = n; k >= i; k--)
                matrix[j][k] -= tmp*matrix[i][k];
        }
    }
    //Метод Гаусса, зворотній хід
    xx[n - 1] = matrix[n - 1][n];
    for (i = n - 2; i >= 0; i--)
    {
        xx[i] = matrix[i][n];
        for (j = i + 1; j<n; j++) xx[i] -= matrix[i][j] * xx[j];
    } //виводимо рішення
    for (i = 0; i<n; i++)
        cout << xx[i] << " ";
    cout << endl;
    delete[] matrix;
    system("pause");
    return 0;
}.
```



## Модифікований метод Гаусса

У багатьох випадках виникає необхідність розв'язання СЛАР з матрицею змінних коефіцієнтів і постійними значеннями стовпця вільних членів. Найчастіше для розв'язання таких задач використовується модифікований метод Гаусса. У цьому методі матриця коефіцієнтів  $A$  із матричного рівняння (1.1) подається у вигляді добутку лівої і правої трикутних матриць:

$$L \cdot R = A.$$

Оскільки діагональні елементи однієї з матриць дорівнюють одиниці, то їх можна не запам'ятовувати, а це тоді дозволяє обидві матриці зберігати в пам'яті комп'ютерної системи на місці матриці коефіцієнтів.

У варіанті методу Краута, використовується така послідовність знаходження елементів матриць  $L$  і  $R$ :

$$\begin{cases} l_{ik} = a_{ik} - \sum_{p=1}^{k-1} l_{ip} r_{pk} & (k=1, 2, \dots, n; i=k, k+1, \dots, n); \\ l_{kk} = \frac{1}{r_{kk}}; \\ r_{kj} = l_{kk} \left( a_{kj} - \sum_{p=1}^{k-1} l_{kp} r_{pj} \right) & (j=k+1, \dots, n); \\ r_{kk} = 1. \end{cases}$$

Система  $AX=C$  зводиться до системи  $LRX=C$ , розв'язання якої замінюється розв'язанням двох систем із трикутними матрицями:

$$\begin{cases} LY = C; \\ RX = Y. \end{cases}$$

Елементи матриць  $Y, X$  визначаються із таких співвідношень:

$$\begin{cases} y_1 = l_{11} c_1; \\ y_i = l_{ii} \left( c_i - \sum_{p=1}^{i-1} l_{ip} y_p \right) & (i=2, \dots, n); \\ x_i = y_i - \sum_{p=i+1}^n r_{ip} x_p & (i=n, n-1, \dots, 1). \end{cases}$$

Кількість арифметичних операцій, необхідних для розв'язання СЛАР цим методом –  $N = 2n^2$ .

**Приклад 1.2.** Розв'язати систему рівнянь модифікованим методом Гаусса із обчисленням прямого та зворотного ходу:

$$\begin{cases} x_1 - 3x_2 + 2x_3 = -5; \\ -2x_1 + x_2 - x_3 = 3; \\ -x_1 - 2x_2 + 3x_3 = 0. \end{cases}$$

*Розв'язання:*

Прямий хід:

1) виключити  $x_1$  з 2-го і 3-го рівнянь:

$$\begin{aligned} -m_2^{(1)} &= -a_{21}/a_{11} = 2/1 = 2; & a_{21}^{(1)} &= a_{21} + m_2^{(1)} \cdot a_{11} = -2 + 2 \cdot 1 = 0; \\ a_{22}^{(1)} &= a_{22} + m_2^{(1)} \cdot a_{12} = 1 - 2 \cdot 3 = -5; & a_{23}^{(1)} &= a_{23} + m_2^{(1)} \cdot a_{13} = -1 + 2 \cdot 2 = 3; \\ b_2^{(1)} &= b_2 + m_2^{(1)} \cdot b_1 = 3 - 2 \cdot 5 = -7. \\ \\ -m_3^{(1)} &= -a_{31}/a_{11} = 1/1 = 1; & a_{31}^{(1)} &= a_{31} + m_3^{(1)} \cdot a_{11} = -1 + 1 \cdot 1 = 0; \\ a_{32}^{(1)} &= a_{32} + m_3^{(1)} \cdot a_{12} = -2 - 1 \cdot 3 = -5; & a_{33}^{(1)} &= a_{33} + m_3^{(1)} \cdot a_{13} = 3 + 1 \cdot 2 = 5; \\ b_3^{(1)} &= b_3 + m_3^{(1)} \cdot b_1 = 0 - 1 \cdot 5 = -5; \end{aligned}$$

$$\begin{cases} x_1 - 3x_2 + 2x_3 = -5; \\ 0 \cdot x_1 - 5x_2 + 3x_3 = -7; \\ 0 \cdot x_1 - 5x_2 + 5x_3 = -5. \end{cases}$$

2) Виключити  $x_2$  з 3-го рівняння:

$$\begin{aligned} -m_3^{(2)} &= -a_{32}^{(1)}/a_{22}^{(1)} = -5/5 = -1; & a_{32}^{(2)} &= a_{32}^{(1)} + m_3^{(2)} \cdot a_{22}^{(1)} = -5 + 1 \cdot 5 = 0; \\ a_{33}^{(2)} &= a_{33}^{(1)} + a_{33}^{(2)} = a_{33}^{(1)} + m_3^{(2)} \cdot a_{23}^{(1)} = 5 - 1 \cdot 3 = 2; \\ b_3^{(2)} &= b_3^{(1)} + m_3^{(2)} \cdot b_2^{(1)} = -5 + 1 \cdot 7 = 2; \end{aligned}$$

$$\begin{cases} x_1 - 3x_2 + 2x_3 = -5; \\ -5x_2 + 3x_3 = -7; \\ 2x_3 = 2. \end{cases}$$

Зворотний хід:

$$\begin{aligned} x_3 &= b_3^{(2)}/a_{33}^{(2)} = 2/2 = 1; \\ x_2 &= (b_2^{(2)} - a_{23}^{(2)} \cdot x_3) / a_{22}^{(2)} = (-7 - 3 \cdot 1) / (-5) = 2; \\ x_1 &= (b_1^{(2)} - a_{13}^{(2)} \cdot x_3 - a_{12}^{(2)} \cdot x_2) / a_{11}^{(2)} = (-5 - (2 \cdot 1) - (-3 \cdot 2)) / 1 = -1. \end{aligned}$$

Алгоритм модифікованого методу Гаусса наведено на рисунку 1.2.

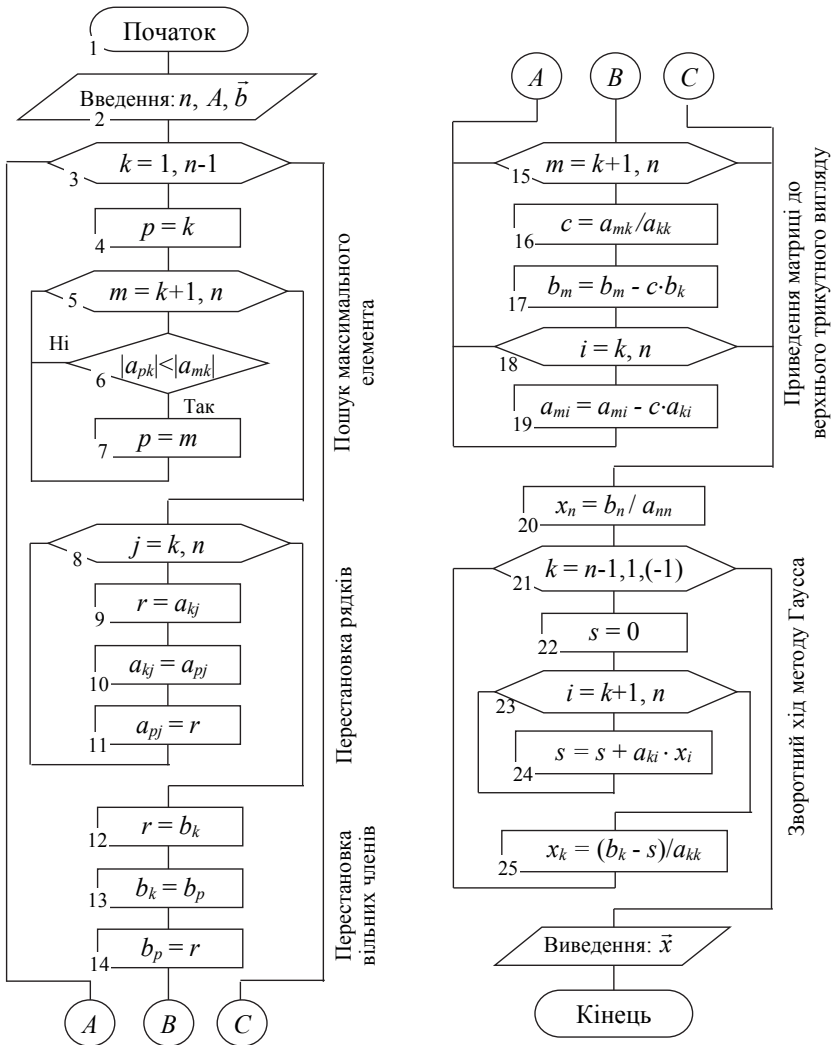


Рисунок 1.2 – Схема алгоритму модифікованого методу Гаусса

Розглянемо реалізацію модифікованого методу Гаусса мовою програмування C++:

```
const int n = 3;
float A[n][n]= {2, 4, 1,
                1, 1, 2,
                4, 2, 1};
float B[n] = {8, 6, 8};
int main(int argc, char *argv[]) {
    // Виведення початкової матриці
    for (int i=0; i<n; i++)
    {
        for (int j=0; j<n; j++)
            cout << A[i][j] << " ";
        cout << B[i] << endl;
    }
    cout << endl;
    // Модифікований метод Гаусса
    for (int i=0; i<n; i++)
    {
        for (int j=0 ; j<n; j++)
            if (i!= j)
            {
                float d = A[j][i]/A[i][i];
                for (int k=0; k<n; k++)
                    A[j][k]= A[i][k]*d - A[j][k];
                B[j] = B[i]*d - B[j]; }
    // виведення проміжних результатів
    for (int i=0; i<n; i++)
    {
        for (int j=0; j<n; j++)
            cout << A[i][j] << " ";
        cout << B[i] << endl;
    }
    cout << endl; }
    // Виведення x
    for (int j=0; j<n; j++)
        cout << "x[" << j << "] = " << B[j]/A[j][j] << endl;
    system("PAUSE");
    return 0;
}
```

### Застосування прямого ходу методу Гаусса для пошуку визначників

Для обчислення визначників матриць застосовуються два підходи:

- рекурсивний розрахунок за допомогою розкладання матриці коефіцієнтів за рядком чи стовпцем;
- обчислення на основі прямого ходу методу Гаусса.

Перший спосіб ґрунтується на використанні властивості визначників, де визначник матриці дорівнює сумі добутків елементів будь-якого рядка чи стовпця на їх алгебраїчне доповнення, тобто:

$$\det(A) = \sum_{j=1}^n a_{ij} \cdot A_{ij}, \forall i = \overline{1, n}.$$

Таким чином, обчислення одного визначника  $n$ -го порядку зводиться до розрахунку  $n$  визначників  $n-1$  порядку. Реалізується цей спосіб за допомогою рекурсії.

Рекурсивний спосіб зручно застосовувати до рядків чи стовпців, що мають велику кількість нульових елементів. Якщо ж нульових елементів у матриці немає або дуже мало, то застосування цього способу є вкрай

неефективним. Для визначника  $n$  порядку доведеться розрахувати  $n!/2$  визначників другого порядку.

Другий спосіб ґрунтується на алгоритмі прямого ходу методу Гаусса, що використовує властивість визначника трикутної матриці. Для такої матриці визначник дорівнює добутку елементів головної діагоналі.

Для обчислення визначника використовується алгоритм побудови послідовності матриць  $A \rightarrow A^1 \rightarrow A^2 \rightarrow \dots \rightarrow A^n$  прямого ходу методу Гаусса з тією відмінністю, що під час перестановки рядків чи стовпців знак визначника змінюється на протилежний. Значення визначника обчислюється за формулою:

$$\det(A) = (-1)^m \cdot a_{11} \cdot a_{22}^1 \cdot \dots \cdot a_{nn}^{n-1},$$

де  $m$  – кількість перестановок.

Цей метод дозволяє обчислювати визначники матриць великих порядків.

### Метод оберненої матриці

Якщо задача розв'язання СЛАР вирішується у пакеті прикладних програм, в якому реалізована функція обчислення оберненої матриці, то для пошуку розв'язку можна застосовувати формулу:

$$X = A^{-1}B,$$

де  $A^{-1}$  – обернена матриця.

Нагадаємо означення оберненої матриці.

Оберненою до квадратної матриці  $A$  називається така матриця  $A^{-1}$ , для якої виконується співвідношення:

$$A \cdot A^{-1} = A^{-1} \cdot A = E,$$

де  $E$  – одинична матриця.

### Метод Крамера

Цей метод полягає у розрахунку визначника  $\det(A)$  матриці коефіцієнтів  $A$ , а також визначників  $\det(A_k)$  матриць  $A_k$  ( $k = \overline{1, n}$ ). Матриці  $A_k$  отримують з матриці  $A$  шляхом заміни  $k$ -го стовпця коефіцієнтів на стовпець  $B$  вільних членів СЛАР.

В цьому випадку можливі такі варіанти значень самих визначників:

1. Якщо  $\det A \neq 0$ , то система має єдиний розв'язок  $\bar{x} = (x_1, \dots, x_n)$ , який визначається за формулою:

$$x_k = \frac{\det(A_k)}{\det(A)} \quad (k = \overline{1, n}).$$

2. Якщо  $\det(A)=0$ , а також усі  $\det(A_k)=0$  за  $k = \overline{1, n}$ , то розв'язків у системи нескінченна множина.

3. Якщо  $\det(A)=0$  і хоча б один  $\det(A_k) \neq 0$ , то система розв'язків не має.

Метод Крамера (визначників) не може бути застосований у більшості практичних задач через велику складність розрахунку безпосередньо самих визначників навіть у разі невеликого зростання порядку системи.

**Приклад 1.3.** Розв'язати СЛАР методом Гаусса, матричним способом та використовуючи метод Крамера:

$$\begin{cases} 3x_1 - x_2 + x_3 = 12; \\ 5x_1 + x_2 + 2x_3 = 3; \\ x_1 + x_2 + 2x_3 = 3. \end{cases}$$

*Розв'язання:*

Розв'яжемо систему матричним способом, для цього обчислимо обернену матрицю  $A^{-1} = \frac{1}{\Delta A} \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix}$ , де  $A_{ij}$  – алгебраїчне доповнення до елементів матриці коефіцієнтів  $A$  ( $i, j=1, 2, 3$ ).

$$\text{Матриця коефіцієнтів: } A = \begin{pmatrix} 3 & -1 & 1 \\ 5 & 1 & 2 \\ 1 & 1 & 2 \end{pmatrix}.$$

Значення визначника матриці:  $\Delta A = \begin{vmatrix} 3 & -1 & 1 \\ 5 & 1 & 2 \\ 1 & 1 & 2 \end{vmatrix} = (3 \cdot 1 \cdot 2) - (3 \cdot 2 \cdot 1) - (-1 \cdot 1 \cdot 2) + (-1 \cdot 2 \cdot 1) + (1 \cdot 5 \cdot 1) - (1 \cdot 1 \cdot 1) = 12 \neq 0$  – матриця невинроджена.

Алгебраїчні доповнення матриці  $A$ :

$$A_i^j = (-1)^{i+j} \det(C_i^j),$$

де  $C_i^j$  – матриця, яка отримується із матриці коефіцієнтів  $A$  шляхом видалення рядка з номером  $i$  і стовпця з номером  $j$ .

Тоді:

$$-(i=1, j=1) A_{11} = (-1)^{1+1} \begin{vmatrix} 1 & 2 \\ 1 & 2 \end{vmatrix} = (2 - 2) = 0, (i=1, j=2)$$

$$A_{12} = (-1)^{1+2} \begin{vmatrix} 5 & 2 \\ 1 & 2 \end{vmatrix} = -(10 - 2) = -8, (i=1, j=3) A_{13} = (-1)^{1+3} \begin{vmatrix} 5 & 1 \\ 1 & 1 \end{vmatrix} = 5 - 1 = 4;$$

$$-(i=2, j=1) A_{21} = (-1)^{2+1} \begin{vmatrix} -1 & 1 \\ 1 & 2 \end{vmatrix} = -(-2 - 1) = 3, (i=2, j=2)$$

$$A_{22} = (-1)^{2+2} \begin{vmatrix} 3 & 1 \\ 1 & 2 \end{vmatrix} = 6 - 1 = 5, (i=2, j=3) A_{23} = (-1)^{2+3} \begin{vmatrix} 3 & -1 \\ 1 & 1 \end{vmatrix} = -(3 + 1) = -4;$$

$$- (i=3, j=1) A_{31} = (-1)^{3+1} \begin{vmatrix} -1 & 1 \\ 1 & 2 \end{vmatrix} = -2 - 1 = -3, (i=3, j=2)$$

$$A_{32} = (-1)^{3+2} \begin{vmatrix} 3 & 1 \\ 5 & 2 \end{vmatrix} = -(6 - 5) = -1, (i=3, j=3) A_{33} = (-1)^{3+3} \begin{vmatrix} 3 & -1 \\ 5 & 1 \end{vmatrix} = 3 + 5 = 8.$$

Значення оберненої матриці:

$$A^{-1} = \frac{1}{\Delta A} \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{pmatrix} = \frac{1}{12} \begin{pmatrix} 0 & 3 & -3 \\ -8 & 5 & -1 \\ 4 & -4 & 8 \end{pmatrix}.$$

Значення матриці (вектора) невідомих аргументів  $X$ :

$$X = A^{-1} \cdot B = \frac{1}{12} \begin{pmatrix} 0 & 3 & -3 \\ -8 & 5 & -1 \\ 4 & -4 & 8 \end{pmatrix} \cdot \begin{pmatrix} 12 \\ 3 \\ 3 \end{pmatrix} = \frac{1}{12} \begin{pmatrix} 0 \cdot 12 + 3 \cdot 3 + (-3) \cdot 3 \\ -8 \cdot 12 + 5 \cdot 3 + (-1) \cdot 3 \\ 4 \cdot 12 + (-4) \cdot 3 + 8 \cdot 3 \end{pmatrix} = \begin{pmatrix} 0 \\ -7 \\ 5 \end{pmatrix}.$$

Розв'язання СЛАР методом Крамера.

$$\text{Головний визначник матриці коефіцієнтів } A: \Delta = \det A = \begin{vmatrix} 3 & -1 & 1 \\ 5 & 1 & 2 \\ 1 & 1 & 2 \end{vmatrix}.$$

Головний визначник матриці коефіцієнтів  $A$  може бути розкладений за елементами першого рядка:

$$\Delta = \det A = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11} \cdot A_{11} + a_{12} \cdot A_{12} + a_{13} \cdot A_{13},$$

$$\Delta = \begin{vmatrix} 3 & -1 & 1 \\ 5 & 1 & 2 \\ 1 & 1 & 2 \end{vmatrix} = 3 \cdot (-1)^{1+1} \cdot \begin{vmatrix} 1 & 2 \\ 1 & 2 \end{vmatrix} + (-1) \cdot (-1)^{1+2} \begin{vmatrix} 5 & 2 \\ 1 & 2 \end{vmatrix} + 1 \cdot (-1)^{1+3} \begin{vmatrix} 5 & 1 \\ 1 & 1 \end{vmatrix} = 12 \neq 0.$$

Запишемо та обчислимо допоміжні визначники:

$$\Delta_1 = \begin{vmatrix} b_1 & a_{12} & a_{13} \\ b_2 & a_{22} & a_{23} \\ b_3 & a_{32} & a_{33} \end{vmatrix}; \Delta_2 = \begin{vmatrix} a_{11} & b_1 & a_{13} \\ a_{21} & b_2 & a_{23} \\ a_{31} & b_3 & a_{33} \end{vmatrix}; \Delta_3 = \begin{vmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ a_{31} & a_{32} & b_3 \end{vmatrix};$$

$$\Delta x_1 = \begin{vmatrix} 12 & -1 & 1 \\ 3 & 1 & 2 \\ 3 & 1 & 2 \end{vmatrix} = 12 \cdot (-1)^{1+1} \cdot \begin{vmatrix} 1 & 2 \\ 1 & 2 \end{vmatrix} + (-1) \cdot (-1)^{1+2} \begin{vmatrix} 3 & 2 \\ 3 & 2 \end{vmatrix} + 1 \cdot (-1)^{1+3} \begin{vmatrix} 3 & 1 \\ 3 & 1 \end{vmatrix} = 0;$$

$$\Delta x_2 = \begin{vmatrix} 3 & 12 & 1 \\ 5 & 3 & 2 \\ 1 & 3 & 2 \end{vmatrix} = 3 \cdot (-1)^{1+1} \cdot \begin{vmatrix} 3 & 2 \\ 3 & 2 \end{vmatrix} + 12 \cdot (-1)^{1+2} \begin{vmatrix} 5 & 2 \\ 1 & 2 \end{vmatrix} + 1 \cdot (-1)^{1+3} \begin{vmatrix} 5 & 3 \\ 1 & 3 \end{vmatrix} = -84;$$

$$\Delta x_3 = \begin{vmatrix} 3 & -1 & 12 \\ 5 & 1 & 3 \\ 1 & 1 & 3 \end{vmatrix} = 3 \cdot (-1)^{1+1} \cdot \begin{vmatrix} 3 & 2 \\ 3 & 2 \end{vmatrix} + (-1) \cdot (-1)^{1+2} \begin{vmatrix} 5 & 3 \\ 1 & 3 \end{vmatrix} + 12 \cdot (-1)^{1+3} \begin{vmatrix} 5 & 1 \\ 1 & 1 \end{vmatrix} = 60.$$

Тоді:  $x_1 = \frac{\Delta x_1}{\Delta} = \frac{0}{12} = 0$ ,  $x_2 = \frac{\Delta x_2}{\Delta} = -\frac{84}{12} = -7$ ,  $x_3 = \frac{\Delta x_3}{\Delta} = \frac{60}{12} = 5$ .

Отже, під час розв'язання усіма вищенаведеними методами однієї і тієї самої СЛАР, було отримано однакову відповідь.

Розглянемо реалізацію методу Крамера мовою програмування C++:

```
void Print_Arr(double** arr, double* barr, const int& rows){
    for(int i = 0; i < rows; ++i){
        for(int j = 0; j < rows; ++j){
            cout << arr[i][j] << " ";
        }
        cout << barr[i] << endl;
    }
}

double Det(double** a, const int& rows, const int& columns){ // функція
    пошуку детермінанта //квадратна матриця розміром n*n
    double** B = new double* [rows];
    for(int i = 0; i < rows; ++i){
        B[i] = new double[columns];
        for(int j = 0; j < columns; ++j){
            B[i][j] = a[i][j];
        }
    }
    int n = rows;
    //приведення матриці до верхнього трикутного виду
    for(int step = 0; step < n - 1; step++){
        for(int row = step + 1; row < n; row++){
            double coeff = -1 * (B[row][step]) / B[step][step]; //метод
            Гаусса
            for(int col = step; col < n; col++){
                B[row][col] += B[step][col] * coeff;
            }
        }
    }
    //Розрахунок визначника як добутку елементів головної діагоналі
    double det = 1.;
    for(int i = 0; i < n; i++){
        det *= B[i][i];
    }
    return det;
}

void MethodKramer(double** arr, double* barr, const int& n){ // функція
    методу Крамера
    double det = Det(arr, n, n);
    int j = 0;
    while(j < n){
        for(int i = 0; i < n; ++i){
            swap(arr[i][j], barr[i]);
        }
        double detj = Det(arr, n, n);
        cout << "x" << j + 1 << " = " << detj/det << endl;
        for(int i = 0; i < n; ++i){
            swap(arr[i][j], barr[i]);
        }
        ++j;
    }
}
```



Алгоритм методу Крамера наведено на рисунку 1.3.

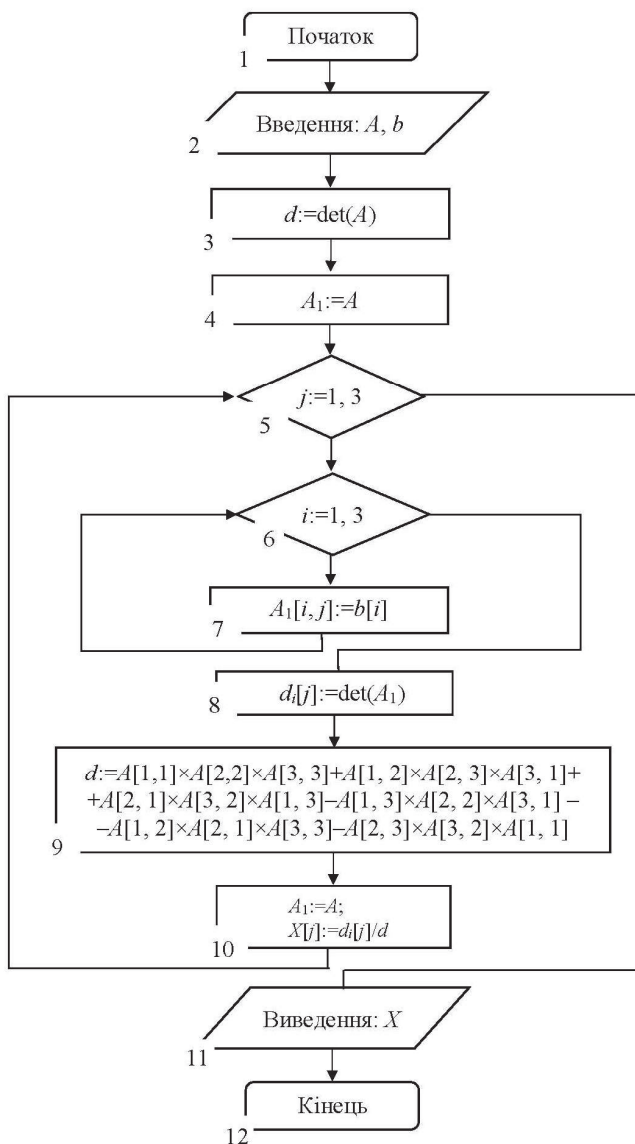


Рисунок 1.3 – Схема алгоритму методу Крамера

## Метод прогонки

Метод прогонки (*sweep method*) застосовується для розв'язання СЛАР із тридіагональною матрицею. Така система рівнянь записується у вигляді:

$$a_i x_{i-1} + b_i x_i + c_i x_{i+1} = d_i \quad (i=1, 2, \dots, n), \quad (1.6)$$

де  $a_1=0, c_n=0$ .

Метод прогонки є окремим випадком методу Гаусса і складається з прямого та зворотного ходу обчислення. Прямий хід обчислення полягає у вилученні елементів матриці системи (1.6), що лежать нижче за головну діагональ. У кожному рівнянні залишиться не більше двох невідомих і формулу зворотного ходу можна записати в такому вигляді:

$$x_i = U_i x_{i+1} + V_i \quad (i = n, n-1, \dots, 1). \quad (1.6)$$

Якщо виразити  $x_{i-1} = U_{i-1} x_i + V_{i-1}$  і підставити у систему (1.6), то матимемо  $a_i(U_{i-1} x_i + V_{i-1}) + b_i x_i + c_i x_{i+1} = d_i$ , звідки:

$$x_i = -\frac{c_i}{a_i U_{i-1} + b_i} x_{i+1} + \frac{d_i - a_i V_{i-1}}{a_i U_{i-1} + b_i}. \quad (1.7)$$

Прирівнюючи (1.7) і (1.8) отримаємо:

$$U_i = -\frac{c_i}{a_i U_{i-1} + b_i}, \quad V_i = \frac{d_i - a_i V_{i-1}}{a_i U_{i-1} + b_i} \quad (i=1, 2, \dots, n), \quad (1.8)$$

Оскільки  $a_1 = 0$ , то

$$U_1 = -c_1 / b_1, \quad V_1 = d_1 / b_1. \quad (1.9)$$

Тепер за формулами (1.9) і (1.10) можуть бути обчислені прогоночні коефіцієнти  $U_i$  та  $V_i$  ( $i=1, 2, \dots, n$ ) на прямому ході прогонки. Знаючи прогоночні коефіцієнти, за формулою (1.7) можна обчислити усі  $x_i$  (зворотний хід прогонки).

Метод прогонки легко алгоритмізується, що визначає його часте використання в стандартному математичному забезпеченні комп'ютерних систем. Застосування методу прогонки дозволяє значно скоротити число арифметичних операцій порівняно із методом Гаусса  $N \approx 3n$ .

**Приклад 1.4.** Розв'язати СЛАР за допомогою методу прогонки:

$$\begin{cases} 10x_1 + x_2 & = 5; \\ -2x_1 + 9x_2 + x_3 & = -1; \\ 0,1x_2 + 4x_3 - x_4 & = -5; \\ -x_3 + 8x_4 & = 40. \end{cases}$$

*Розв'язання:*

Коефіцієнти записуються у вигляді таблиці 1.1

Таблиця 1.1 – Значення коефіцієнтів СЛАР

$i$	$a_i$	$b_i$	$c_i$	$d_i$
1	0,0	10,0	1,0	5,0
2	-2,0	9,0	1,0	-1,0
3	0,1	4,0	-1,0	-5,0
4	-1,0	8,0	0,0	40,0

Прямий хід прогонки. За формулами (1.11) і (1.12) визначаються прогнорні коефіцієнти  $U_i$  та  $V_i$ :

$$U_1 = -c_1 / b_1 = -1/10 = -0,1;$$

$$V_1 = d_1 / b_1 = 5/10 = 0,5;$$

$$U_2 = -c_2 / (a_2 U_1 + b_2) = -1 / (2 \cdot 0,1 + 9) = -0,1087;$$

$$V_2 = (d_2 - a_2 V_1) / (a_2 U_1 + b_2) = (-1 + 2 \cdot 0,5) / (2 \cdot 0,1 + 9) = 0;$$

$$U_3 = -c_3 / (a_3 U_2 + b_3) = 1 / (-0,1 \cdot 0,1087 + 4) = 0,2507;$$

$$V_3 = (d_3 - a_3 V_2) / (a_3 U_2 + b_3) = (-5 - 0,1 \cdot 0) / (-0,1 \cdot 0,1087 + 4) = -1,2534;$$

$$U_4 = -c_4 / (a_4 U_3 + b_4) = 0, \text{ так як } c_4 = 0;$$

$$V_4 = (d_4 - a_4 V_3) / (a_4 U_3 + b_4) = (40 - 1 \cdot 1,2534) / (-1 \cdot 0,2507 + 8) = 5,0.$$

Зворотний хід прогонки. За формулами (1.9) обчислюються усі невідомі значення  $x_i$ :

$$x_4 = V_4 = 5,0 \quad (U_4 = 0);$$

$$x_3 = U_3 x_4 + V_3 = 0,2507 \cdot 5 - 1,2534 = 0,0001 \approx 0;$$

$$x_2 = U_2 x_3 + V_2 = -1,1087 \cdot 0,0001 + 0 = -0,0001 \approx 0;$$

$$x_1 = U_1 x_2 + V_1 = 0,1 \cdot 0,0001 + 0,5 = 0,5001 \approx 0,5.$$

Наведемо реалізацію методу прогонки мовою програмування C++.

```
// Прямий хід методу прогонки
int N1 = N - 1;
y = matA[0][0];
a[0] = -matA[0][1] / y;
b[0] = matB[0] / y;
for (int i = 1; i < N1; i++) {
y = matA[i][i] + matA[i][i - 1] * a[i - 1];
a[i] = -matA[i][i + 1] / y;
b[i] = (matB[i] - matA[i][i - 1] * b[i - 1]) / y;
}
// На зворотному ході розраховуються корені системи рівняння:
matRes[N1] = (matB[N1] - matA[N1][N1 - 1] * b[N1 - 1]) / (matA[N1][N1] +
matA[N1][N1 - 1] * a[N1 - 1]);
for (int i = N1 - 1; i >= 0; i--) {
matRes[i] = a[i] * matRes[i + 1] + b[i];
}
```

Алгоритм методу прогонки наведено на рисунку 1.4.

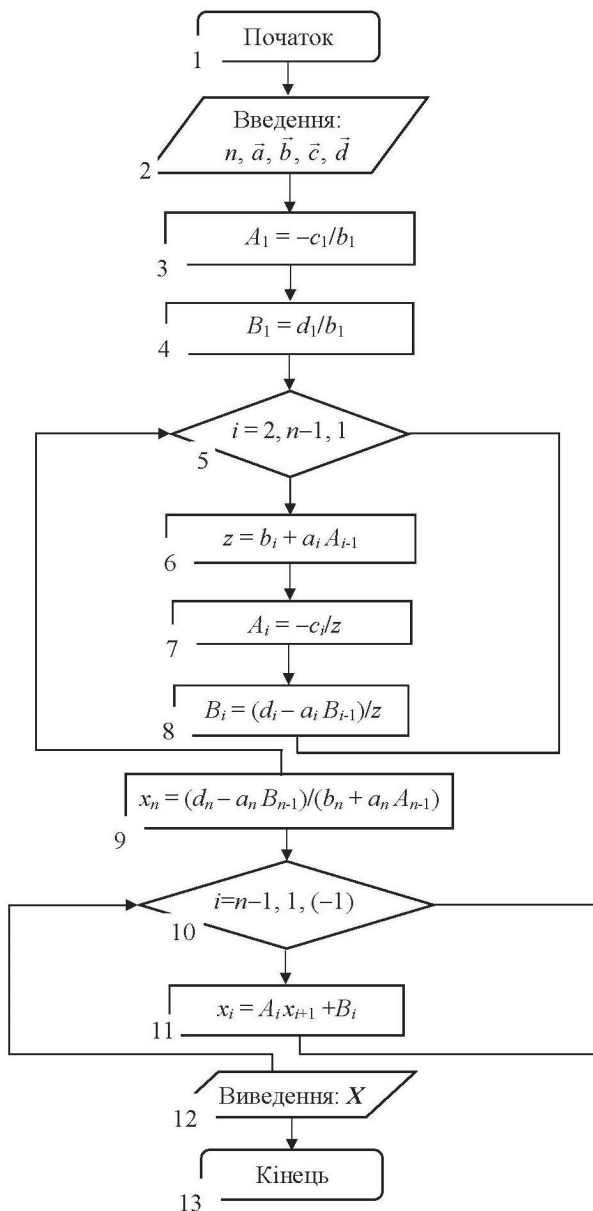


Рисунок 1.4 – Схема алгоритму методу прогонки



$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1; \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2; \\ \dots\dots\dots; \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n = b_n. \end{cases} \quad (1.11)$$

Щоб таку систему розв'язати методом Якобі (прості ітерації), необхідно систему (1.11) звести до такого вигляду:

$$\begin{cases} x_1 = -\frac{a_{12}}{a_{11}}x_2 - \frac{a_{13}}{a_{11}}x_3 - \dots - \frac{a_{1n}}{a_{11}}x_n + \frac{b_1}{a_{11}}; \\ x_2 = -\frac{a_{21}}{a_{22}}x_1 - \frac{a_{23}}{a_{22}}x_3 - \dots - \frac{a_{2n}}{a_{22}}x_n + \frac{b_2}{a_{22}}; \\ \dots\dots\dots; \\ x_n = -\frac{a_{n1}}{a_{nn}}x_1 - \frac{a_{n2}}{a_{nn}}x_2 - \dots - \frac{a_{nn-1}}{a_{nn}}x_{n-1} + \frac{b_n}{a_{nn}}. \end{cases} \quad (1.12)$$

У системі (1.11)  $i$ -те рівняння являє собою  $i$ -те рівняння системи (1.12), розв'язане відносно  $i$ -ї невідомої ( $i = \overline{1, n}$ ).

Метод розв'язання СЛАР (1.11) за допомогою зведення до системи (1.12) із подальшим її розв'язанням ітераційним методом називається методом Якобі (прості ітерації) для системи (1.14).

Таким чином, формули для методу Якобі (прості ітерації) розв'язання системи (1.11) матимуть такий вигляд:

$$\begin{cases} x_1^{k+1} = -\frac{a_{12}}{a_{11}}x_2^k - \frac{a_{13}}{a_{11}}x_3^k - \dots - \frac{a_{1n}}{a_{11}}x_n^k + \frac{b_1}{a_{11}} = -\sum_{j=2}^n \frac{a_{1j}}{a_{11}}x_j^k + \frac{b_1}{a_{11}}; \\ x_2^{k+1} = -\frac{a_{21}}{a_{22}}x_1^k - \frac{a_{23}}{a_{22}}x_3^k - \dots - \frac{a_{2n}}{a_{22}}x_n^k + \frac{b_2}{a_{22}} = -\sum_{j=1, j \neq 2}^n \frac{a_{2j}}{a_{22}}x_j^k + \frac{b_2}{a_{22}}; \\ \dots\dots\dots; \\ x_n^{k+1} = -\frac{a_{n1}}{a_{nn}}x_1^k - \frac{a_{n2}}{a_{nn}}x_2^k - \dots - \frac{a_{nn-1}}{a_{nn}}x_{n-1}^k + \frac{b_n}{a_{nn}} = -\sum_{j=1}^{n-1} \frac{a_{nj}}{a_{nn}}x_j^k + \frac{b_n}{a_{nn}}, \end{cases} \quad (1.13)$$

$(k = 0, 1, 2, \dots)$ .

Формули (1.13) можуть бути записані у такому вигляді:

$$x_i^{k+1} = -\sum_{j=1, j \neq i}^n \frac{a_{ij}}{a_{ii}}x_j^k + \frac{b_i}{a_{ii}} \quad (i = 1, 2, \dots, n; k = 0, 1, 2, \dots).$$

**Приклад 1.5.** Розв'язати СЛАР (система з переважаючими діагональними коефіцієнтами) чисельним методом Якобі (простої ітерації) із заданою точністю  $\Delta = 0,065$ :

$$\begin{cases} 4,300x_1 + 0,217x_2 = 2,663; \\ 0,100x_1 - 3,400x_2 - 0,207x_3 = 2,778; \\ 0,090x_2 + 2,500x_3 + 0,197x_4 = 2,533; \\ 0,080x_3 - 1,600x_4 = 1,928. \end{cases}$$

*Розв'язання*

Запишемо еквівалентну систему рівнянь:

$$\begin{cases} x_1 = \frac{2,663}{4,300} - \frac{0,217}{4,300}x_2 = 0,6193 - 0,050x_2; \\ x_2 = -\frac{2,778}{3,4} - \frac{0,207}{3,4}x_3 + \frac{0,1}{3,4}x_1 = -0,817 - 0,061x_3 + 0,029x_1; \\ x_3 = \frac{2,533}{2,500} - \frac{0,197}{2,500}x_4 - \frac{0,09}{2,50}x_2 = 1,013 - 0,079x_4 - 0,036x_2; \\ x_4 = -\frac{1,928}{1,600} + \frac{0,08}{1,60}x_3 = -1,205 + 0,05x_3. \end{cases}$$

Оскільки сума коефіцієнтів по рядках у правій стороні системи очевидно менша одиниці, буде виконуватись «слабка» умова збіжності. За нульове наближення можуть бути взяті такі значення:

$$x_1^{(0)} = 0,619; \quad x_2^{(0)} = -0,817; \quad x_3^{(0)} = 1,013; \quad x_4^{(0)} = -1,205.$$

На основі системи виконується обчислення першої ітерації:

$$\begin{cases} x_1^{(1)} = 0,6193 - 0,050x_2^{(0)} = 0,6193 - 0,050 \cdot (-0,817) = 0,6604; \\ x_2^{(1)} = -0,817 - 0,061x_4^{(0)} + 0,029x_1^{(0)} = -0,817 - 0,061 \cdot (-1,205) + \\ + 0,029 \cdot 0,619 = -0,8606; \\ x_3^{(1)} = 1,013 - 0,079x_4^{(0)} - 0,036x_2^{(0)} = 1,013 - 0,079 \cdot (-1,205) - \\ - 0,036 \cdot (-0,817) = 1,1373; \\ x_4^{(1)} = -1,205 + 0,05x_3^{(0)} = -1,205 + 0,05 \cdot 1,013 = -1,155. \end{cases}$$

Після чого виконується обчислення другої ітерації:

$$\begin{cases} x_1^{(2)} = 0,6193 - 0,050x_2^{(1)} = 0,6193 - 0,050 \cdot (-0,8606) = 0,6623; \\ x_2^{(2)} = -0,817 - 0,061x_4^{(1)} + 0,029x_1^{(1)} = -0,817 - 0,061 \cdot (-1,155) + \\ + 0,029 \cdot 0,6604 = -0,8678; \\ x_3^{(2)} = 1,013 - 0,079x_4^{(1)} - 0,036x_2^{(1)} = 1,013 - 0,079 \cdot (-1,155) - \\ - 0,036 \cdot (-0,8606) = 1,072; \\ x_4^{(2)} = -1,205 + 0,05x_3^{(1)} = -1,205 + 0,05 \cdot 1,1373 = -1,148. \end{cases}$$

Виконується обчислення третьої ітерації:

$$\begin{cases} x_1^{(3)} = 0,6193 - 0,050x_2^{(2)} = 0,6193 - 0,050 \cdot (-0,8678) = 0,6623; \\ x_2^{(3)} = -0,817 - 0,061x_4^{(2)} + 0,029x_1^{(2)} = -0,817 - 0,061 \cdot (-1,148) + \\ + 0,029 \cdot 0,6623 = -0,862; \\ x_3^{(3)} = 1,013 - 0,079x_4^{(2)} - 0,036x_2^{(2)} = 1,013 - 0,079 \cdot (-1,148) - \\ - 0,036 \cdot (-0,8678) = 1,133; \\ x_4^{(3)} = -1,205 + 0,05x_3^{(2)} = -1,205 + 0,05 \cdot 1,072 = -1,1218. \end{cases}$$

Виконаємо оцінення похибки обчислень:

$$\begin{cases} |x_1^{(2)} - x_1^{(1)}| = |0,6623 - 0,6604| = 0,0019; \\ |x_2^{(2)} - x_2^{(1)}| = |-0,8678 - (-0,8606)| = 0,0072; \\ |x_3^{(2)} - x_3^{(1)}| = |1,072 - 1,1373| = 0,0653; \\ |x_4^{(2)} - x_4^{(1)}| = |(-1,148) - (-1,155)| = 0,0070; \end{cases}$$

$$\Delta_1 = \max \{0,0019; 0,0072; 0,0653; 0,0070\} = 0,0653 > \Delta = 0,065;$$

$$\begin{cases} \Delta_1^{(2)} = |x_1^{(3)} - x_1^{(2)}| = |0,6623 - 0,6623| = 0,0000; \\ \Delta_2^{(2)} = |x_2^{(3)} - x_2^{(2)}| = |(-0,8620) - (-0,8678)| = 0,0058; \\ \Delta_3^{(2)} = |x_3^{(3)} - x_3^{(2)}| = |1,133 - 1,072| = 0,0610; \\ \Delta_4^{(2)} = |x_4^{(3)} - x_4^{(2)}| = |(-1,1218) - (-1,1480)| = 0,0262; \end{cases}$$

$$\Delta_2 = \max \{0,0000; 0,0058; 0,0610; 0,0262\} = 0,0610 < \Delta = 0,065.$$

Виконання умови  $\Delta_2 < 0,065$  свідчить про досягнення заданої точності обчислення, тому розв'язок системи із похибкою  $\Delta = 0,065$ :

$$\begin{cases} x_1^{(3)} = 0,6623; & x_2^{(3)} = -0,8620; \\ x_3^{(3)} = 1,1330; & x_4^{(3)} = -1,1218. \end{cases}$$



У методі послідовної верхньої релаксації нові значення кожної змінної обчислюються як:

$$x_i^{(m+1)} = x_i^{(m)} + \omega(\bar{x}_i^{(m+1)} - x_i^{(m)}),$$

де  $\bar{x}_i^{(m+1)}$  – уточнене значення  $x_i^{(m)}$  за методом Гаусса-Зейделя,

$\omega$  – параметр релаксації ( $1,0 \leq \omega \leq 2,0$ ).

За  $\omega = 1,0$  цей метод тотожний методу Гаусса-Зейделя. Швидкість збіжності залежить від значення параметра релаксації  $\omega$ .

Алгоритм чисельної реалізації методу Якобі (прості ітерації) для розв'язку СЛАР подано на рисунку 1.5.

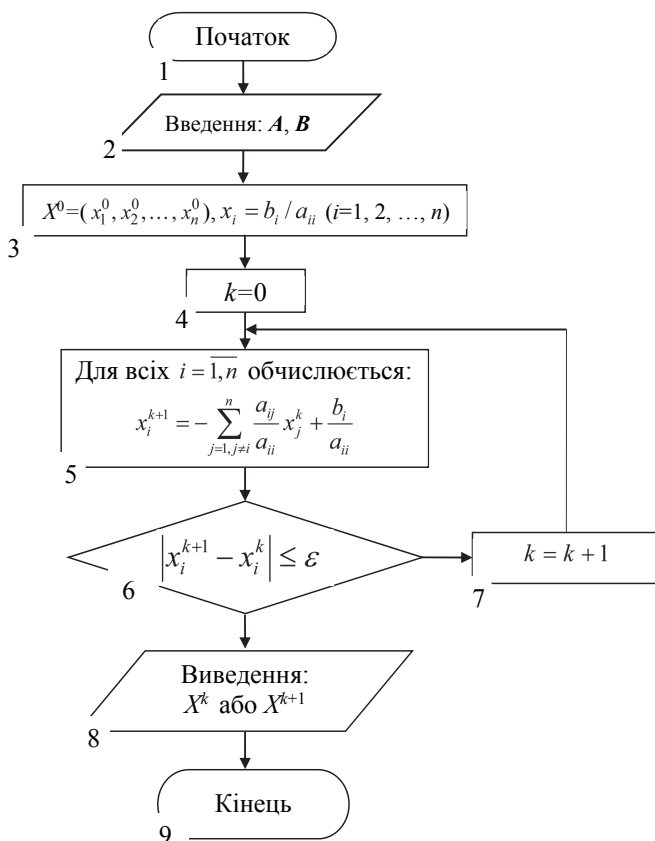


Рисунок 1.5 – Схема алгоритму методу Якобі (прості ітерації) для СЛАР

Розглянемо реалізацію методу простої ітерації мовою програмування C++:

```
float G1 (float x1, float x2, float x3, float x4)
{ return -0.6 + 0.7*x1 - 0.5*x2 - 0.3*x3- 0.5*x4;}
float G2 (float x1, float x2, float x3, float x4)
{ return -0.3 - 0.3*x1 + 0.6*x2 -0.1*x3 -0.2*x4;}
float G3 (float x1, float x2, float x3, float x4)
{ return -0.8 - 0.6*x1 -0.2*x2 +0.2*x3 - 0.2*x4;}
float G4 (float x1, float x2, float x3, float x4)
{ return -0.8 - 0.3*x1 -0.5*x2 -0.3*x3 + 0.3*x4;}
int main(int argc, char *argv[]) {
    float delta = 0.01;
    float x1 = 1, x2 = 1, x3 = 1, x4 = 1;
    float x10, x20, x30, x40;
    do
    {
        x10 = x1;
        x20 = x2;
        x30 = x3;
        x40 = x4;
        x1 = G1(x10, x20, x30, x40);
        x2 = G2(x10, x20, x30, x40);
        x3 = G3(x10, x20, x30, x40);
        x4 = G4(x10, x20, x30, x40);
    }
    while ((fabs(x1-x10) >= delta) && (fabs(x2-x20)>= delta) && (fabs(x3-x30)
    >= delta)&&(fabs(x4-x40) >= delta));
    //
    cout << "x1 = " << x1 << endl;
    cout << " x2 = " << x2 << endl;
    cout << " x3 = " << x3 << endl;
    cout << " x4 = " << x4 << endl << endl;
    system("PAUSE");
    return 0;}
```

## Метод Гаусса-Зейделя

У випадку використання методу Гаусса-Зейделя під час обчислення  $(k+1)$ -го наближення невідомого значення  $x_i$  ( $i > 1$ ) використовуються обчислені раніше  $(k+1)$ -е наближення невідомих  $x_1, x_2, \dots, x_{i-1}$ .

Розглянемо цей метод на прикладі розв'язання СЛАР:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1; \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2; \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3. \end{cases}$$

Припустимо, що діагональні елементи  $a_{11}, a_{22}, a_{33}$  відмінні від нуля. Тоді, виражаючи невідомі значення  $x_1, x_2$  та  $x_3$  відповідно з першого, другого і третього рівнянь вихідної системи, отримаємо:

$$\begin{cases} x_1 = \frac{1}{a_{11}}(b_1 - a_{12}x_2 - a_{13}x_3); \\ x_2 = \frac{1}{a_{22}}(b_2 - a_{21}x_1 - a_{23}x_3); \\ x_3 = \frac{1}{a_{33}}(b_3 - a_{31}x_1 - a_{32}x_2). \end{cases} \quad (1.14)$$

Якщо задати деякі початкові (нульові) наближення значень невідомих  $x_1 = x_1^{(0)}$ ,  $x_2 = x_2^{(0)}$ ,  $x_3 = x_3^{(0)}$  і підставити їх у праву частину рівнянь системи (1.14), буде отримано нове наближення (перша ітерація) для  $x_1$ ,  $x_2$  і  $x_3$ , відповідно:

$$\begin{cases} x_1^{(1)} = \frac{1}{a_{11}}(b_1 - a_{12}x_2^{(0)} - a_{13}x_3^{(0)}); \\ x_2^{(1)} = \frac{1}{a_{22}}(b_2 - a_{21}x_1^{(1)} - a_{23}x_3^{(0)}); \\ x_3^{(1)} = \frac{1}{a_{33}}(b_3 - a_{31}x_1^{(1)} - a_{32}x_2^{(1)}). \end{cases}$$

Для другої ітерації будуть використовуватись нові значення для  $x_1$ ,  $x_2$  та  $x_3$ , а саме:  $x_1 = x_1^{(2)}$ ,  $x_2 = x_2^{(2)}$ ,  $x_3 = x_3^{(2)}$  і т. д.

Тоді  $k$ -те наближення може бути подано у вигляді:

$$\begin{cases} x_1^{(k)} = \frac{1}{a_{11}}(b_1 - a_{12}x_2^{(k-1)} - a_{13}x_3^{(k-1)}); \\ x_2^{(k)} = \frac{1}{a_{22}}(b_2 - a_{21}x_1^{(k)} - a_{23}x_3^{(k-1)}); \\ x_3^{(k)} = \frac{1}{a_{33}}(b_3 - a_{31}x_1^{(k)} - a_{32}x_2^{(k)}). \end{cases}$$

Ітераційний процес триває до тих пір, поки значення  $x_1^{(k)}$ ,  $x_2^{(k)}$ ,  $x_3^{(k)}$  не стануть близькими із заданою похибкою до значень  $x_1^{(k-1)}$ ,  $x_2^{(k-1)}$ ,  $x_3^{(k-1)}$ .

Розглядаємо тепер систему  $n$  лінійних рівнянь з  $n$  невідомими ( $i = 1, 2, \dots, n$ ). Припустимо, що всі діагональні елементи відмінні від нуля. Випишемо  $i$ -те рівняння:

$$a_{i1}x_1 + \dots + a_{i,i-1}x_{i-1} + a_{ii}x_i + a_{i,i+1}x_{i+1} + \dots + a_{in}x_n = b_i.$$

Тоді, відповідно до методу Гаусса-Зейделя  $k$  наближення розв'язку можна записати у вигляді:

$$x_i^{(k)} = \frac{1}{a_{ii}} (b_i - a_{i1}x_1^{(k)} - \dots - a_{i,i-1}x_{i-1}^{(k)} - a_{i,i+1}x_{i+1}^{(k-1)} - \dots - a_{in}x_n^{(k-1)}).$$

Ітераційний процес буде тривати до тих пір, поки усі значення  $x_i^{(k)}$  не стануть близькими до  $x_i^{(k-1)}$ . Близість цих значень характеризується максимальною абсолютною величиною їх різниці  $\delta$ . Тоді за заданої допустимої похибки  $\Delta > 0$  умову закінчення ітераційного процесу може бути записано у вигляді:

$$\delta = \max_{1 \leq i \leq n} |x_i^{(k)} - x_i^{(k-1)}| < \Delta.$$

Ця умова є критерієм на основі абсолютного відхилення, яка може бути замінена критерієм на основі відносних різниць. Тобто умова закінчення ітераційного процесу може бути записати у вигляді (для  $|x_i| \gg 1$ ):

$$\max_{1 \leq i \leq n} \left| \frac{x_i^{(k)} - x_i^{(k-1)}}{x_i^{(k)}} \right| < \varepsilon.$$

У разі виконання однієї з цих умов, ітераційний процес розв'язання методом Гаусса-Зейделя називається збіжним. У цьому випадку максимальні різниці  $\delta$  між значеннями змінних у двох наступних ітераціях спадають, а самі ці значення будуть наближатись до розв'язку системи рівнянь.

Однією із головних умов щодо застосування ітераційних методів є виконання умови збіжності. Аналіз збіжності ітераційних методів розв'язку СЛАР пов'язаний із використанням поняття норми матриці. Норма матриці  $\mathbf{B}$  – це таке характеристичне число  $\|\mathbf{B}\|$ , яке має такі властивості: норма завжди більше нуля ( $\|\mathbf{B}\| > 0$ ), причому дорівнює нулю тільки для нульової матриці; якщо матриця  $\mathbf{B}$  множиться на дійсний коефіцієнт  $\lambda$ , тоді і норма має перемножатися на модуль цього коефіцієнта –  $|\lambda| \cdot \|\mathbf{B}\|$ ; норма суми двох матриць  $\mathbf{A}$ ,  $\mathbf{B}$  не більше суми норм (умова адитивності норми –  $\|\mathbf{A} + \mathbf{B}\| \leq \|\mathbf{A}\| + \|\mathbf{B}\|$ ), і норма добутку двох матриць  $\mathbf{A}$ ,  $\mathbf{B}$  не більше добутку норм (умова мультиплікативності норми –  $\|\mathbf{AB}\| \leq \|\mathbf{A}\| \cdot \|\mathbf{B}\|$ ).

Найбільшого поширення отримали такі норми матриць:

$$\|\mathbf{B}\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^n |b_{ij}| \quad - \text{перша норма}; \quad \|\mathbf{B}\|_2 = \max_{1 \leq j \leq n} \sum_{i=1}^n |b_{ij}| \quad - \text{друга норма};$$

$$\|\mathbf{B}\|_E = \sqrt{\sum_{i=1}^n \sum_{j=1}^n b_{ij}^2} \quad - \text{Е-норма (евклідова норма)}.$$

Існує декілька підходів до визначення збіжності за допомогою оцінення норм. У загальному випадку достатньо, щоб хоча б одна з норм матриці  $\mathbf{B}$  була меншою за одиницю, а саме:  $\|\mathbf{B}\| < 1$ .

У математиці така умова називається «звичайною» або «сильною». У багатьох випадках збіжність забезпечується і за виконання так званої «слабкої» ознаки. Наприклад, «слабка» ознака збіжності на основі сум рядків, а саме: для всіх сум коефіцієнтів рядків ( $i = 1, 2, \dots, n$ ) виконується

співвідношення  $-\sum_{j=i}^n |b_{ij}| \leq 1$ , але є один рядок  $p$ , для якого  $\sum_{j=i}^n |b_{pj}| < 1$ .

Аналогічно визначається «слабка» ознака сум для стовпців матриці.

«Слабка» ознака використовується в тих випадках, коли матриця коефіцієнтів  $A$  СЛАР не є розкладною, тобто це квадратна матриця  $A$ , яку (на відміну від розкладної) не можна звести до вигляду:

$$\begin{pmatrix} A_1 & A_2 \\ 0 & A_3 \end{pmatrix},$$

де  $A_1, A_2, A_3$  – квадратні матриці.

Для розкладних матриць СЛАР розпадається на дві системи рівнянь, що розв'язуються послідовно. У першоджерелах, список яких наведено наприкінці цього підручника, міститься багато детальніших доведень й аналізів властивостей та ознак для оцінення збіжності, але для загальноінженерного підходу розв'язання багатьох практичних задач достатньо наведені вище дані.

На практиці для збіжності ітераційного процесу достатньо, щоб модулі діагональних коефіцієнтів кожного рівняння системи були не менші суми модулів усіх інших коефіцієнтів:

$$|a_{ij}| \geq \sum_{i \neq j} |a_{ij}|,$$

Ця умова є достатньою для збіжності методу, але не є необхідною, бо для деяких систем ітерації збігаються і у разі порушення цих умов.

**Приклад 1.6** Розв'язати СЛАР (табл. 1.2) чисельним методом Гаусса-Зейделя із заданою точністю  $\Delta=0,001$ :

Таблиця 1.2 – Вихідні дані СЛАР

Матриця системи				Права частина
0,401	0,301	0,000	0,000	0,122
-0,029	-0,500	-0,018	0,000	-0,253
0,000	-0,050	-1,400	-0,039	-0,988
0,000	0,000	-0,007	-2,300	-2,082

*Розв'язання:*

Розширена матриця системи (табл. 1.2) має такий вигляд:

$$C = \begin{pmatrix} 0,401 & 0,301 & 0,000 & 0,000 & 0,122 \\ -0,029 & -0,500 & -0,018 & 0,000 & -0,253 \\ 0,000 & -0,050 & -1,400 & -0,039 & -0,988 \\ 0,000 & 0,000 & -0,007 & -2,300 & -2,082 \end{pmatrix}.$$

Зведемо систему до еквівалентного вигляду:

$$\begin{cases} x_1 = \frac{1}{0,401}(0,122 - 0,301x_2) = 0,304 - 0,751x_2; \\ x_2 = -\frac{1}{0,5}(-0,253 + 0,018x_3 + 0,029x_1) = 0,506 - 0,036x_3 - 0,058x_1; \\ x_3 = -\frac{1}{1,4}(-0,988 + 0,050x_2 + 0,039x_4) = 0,706 - 0,036x_2 - 0,028x_4; \\ x_4 = -\frac{1}{2,3}(-2,082 + 0,007x_3) = 0,905 - 0,003x_3. \end{cases}$$

Оскільки сума коефіцієнтів по рядках у правій стороні системи очевидно менша одиниці, то буде виконуватись «слабка» умова збіжності.

За початкові наближення беруться усі  $x_i$ , що дорівнюють нулю у правій частині системи, а саме:  $x_1^{(0)}=x_2^{(0)}=x_3^{(0)}=x_4^{(0)}=0$ .

Тоді значення невідомих аргументів на першій ітерації:

$$\begin{cases} x_1^{(1)} = 0,304 - 0,751x_2^{(0)} = 0,304 - 0,751 \cdot 0 = 0,304; \\ x_2^{(1)} = 0,506 - 0,036x_3^{(0)} - 0,058x_1^{(1)} = 0,506 - 0,036 \cdot 0 - 0,058 \cdot 0,304 = 0,488; \\ x_3^{(1)} = 0,706 - 0,036x_2^{(1)} - 0,028x_4^{(0)} = 0,706 - 0,036 \cdot 0,488 - 0,028 \cdot 0 = 0,688; \\ x_4^{(1)} = 0,905 - 0,003x_3^{(1)} = 0,905 - 0,003 \cdot 0,688 = 0,903. \end{cases}$$

На основі результатів обчислення першої ітерації виконується друга обчислювальна ітерація:

$$\begin{cases} x_1^{(2)} = 0,304 - 0,751x_2^{(1)} = 0,304 - 0,751 \cdot 0,488 = -0,062; \\ x_2^{(2)} = 0,506 - 0,036x_3^{(1)} - 0,058x_1^{(2)} = 0,506 - 0,036 \cdot 0,688 - \\ - 0,058 \cdot (-0,062) = 0,485; \\ x_3^{(2)} = 0,706 - 0,036x_2^{(2)} - 0,028x_4^{(1)} = 0,706 - 0,036 \cdot 0,485 - 0,028 \cdot 0,903 = 0,663; \\ x_4^{(2)} = 0,905 - 0,003x_3^{(2)} = 0,905 - 0,003 \cdot 0,663 = 0,903. \end{cases}$$

На основі результатів двох ітерацій перевіряється точність поточного обчислення:

$$\begin{cases} |x_1^{(2)} - x_1^{(1)}| = |-0,062 - 0,304| = 0,366; \\ |x_2^{(2)} - x_2^{(1)}| = |0,488 - 0,485| = 0,003; \\ |x_3^{(2)} - x_3^{(1)}| = |0,663 - 0,688| = 0,025; \\ |x_4^{(2)} - x_4^{(1)}| = |0,903 - 0,903| = 0; \end{cases}$$

$$\Delta_1 = \max \{0,366; 0,003; 0,025; 0\} = 0,366 > \Delta = 0,001.$$

Значення  $\Delta_1 > 0,001$  означає, що заданої точності обчислення ще не досягнуто, що свідчить про необхідність виконання третьої обчислювальної ітерації:

$$\begin{cases} x_1^{(3)} = 0,304 - 0,751x_2^{(2)} = 0,304 - 0,751 \cdot 0,485 = -0,060; \\ x_2^{(3)} = 0,506 - 0,036x_3^{(2)} - 0,058x_1^{(3)} = 0,506 - 0,036 \cdot 0,663 - 0,058 \cdot (-0,060) = 0,486; \\ x_3^{(3)} = 0,706 - 0,036x_2^{(3)} - 0,028x_4^{(2)} = 0,706 - 0,036 \cdot 0,486 - 0,028 \cdot (0,903) = 0,663; \\ x_4^{(3)} = 0,905 - 0,003x_3^{(3)} = 0,905 - 0,003 \cdot 0,663 = 0,903. \end{cases}$$

Виконується перевірка точності поточного обчислення:

$$\begin{cases} |x_1^{(3)} - x_1^{(2)}| = |-0,060 - 0,062| = 0,002; \\ |x_2^{(3)} - x_2^{(2)}| = |0,486 - 0,485| = 0,001; \\ |x_3^{(3)} - x_3^{(2)}| = |0,663 - 0,663| = 0; \\ |x_4^{(3)} - x_4^{(2)}| = |0,903 - 0,903| = 0; \end{cases}$$

$$\Delta_2 = \max\{0,002; 0,001; 0; 0\} = 0,002 > \Delta = 0,001.$$

Оскільки  $\Delta_2 > 0,001$ , то заданої точності обчислення також не досягнуто, що свідчить про необхідність виконання четвертої ітерації:

$$\begin{cases} x_1^{(4)} = 0,304 - 0,751x_2^{(3)} = 0,304 - 0,751 \cdot 0,486 = -0,060; \\ x_2^{(4)} = 0,506 - 0,036x_3^{(3)} - 0,058x_1^{(4)} = 0,506 - 0,036 \cdot 0,663 - \\ - 0,058 \cdot (-0,060) = 0,486; \\ x_3^{(4)} = 0,706 - 0,036x_2^{(4)} - 0,028x_4^{(3)} = 0,706 - 0,036 \cdot 0,486 - \\ - 0,028 \cdot (0,903) = 0,663; \\ x_4^{(4)} = 0,905 - 0,003x_3^{(4)} = 0,905 - 0,003 \cdot 0,663 = 0,903. \end{cases}$$

Виконується перевірка точності поточного обчислення:

$$\begin{cases} |x_1^{(4)} - x_1^{(3)}| = |-0,060 - 0,060| = 0; \\ |x_2^{(4)} - x_2^{(3)}| = |0,486 - 0,486| = 0; \\ |x_3^{(4)} - x_3^{(3)}| = |0,663 - 0,663| = 0; \\ |x_4^{(4)} - x_4^{(3)}| = |0,903 - 0,903| = 0; \end{cases}$$

$$\Delta_3 = \max\{0; 0; 0; 0\} = 0 < \Delta = 0,001.$$

Тепер  $\Delta_3 < 0,001$ , а це свідчить про досягнення заданої точності обчислення. Тому розв'язок системи із похибкою  $\Delta = 0,001$ :

$$\begin{cases} x_1^{(4)} = -0,060; \\ x_2^{(4)} = 0,486; \\ x_3^{(4)} = 0,663; \\ x_4^{(4)} = 0,903. \end{cases}$$

Алгоритм чисельної реалізації методу Гаусса-Зейделя для розв'язку СЛАР подано на рисунку 1.6.

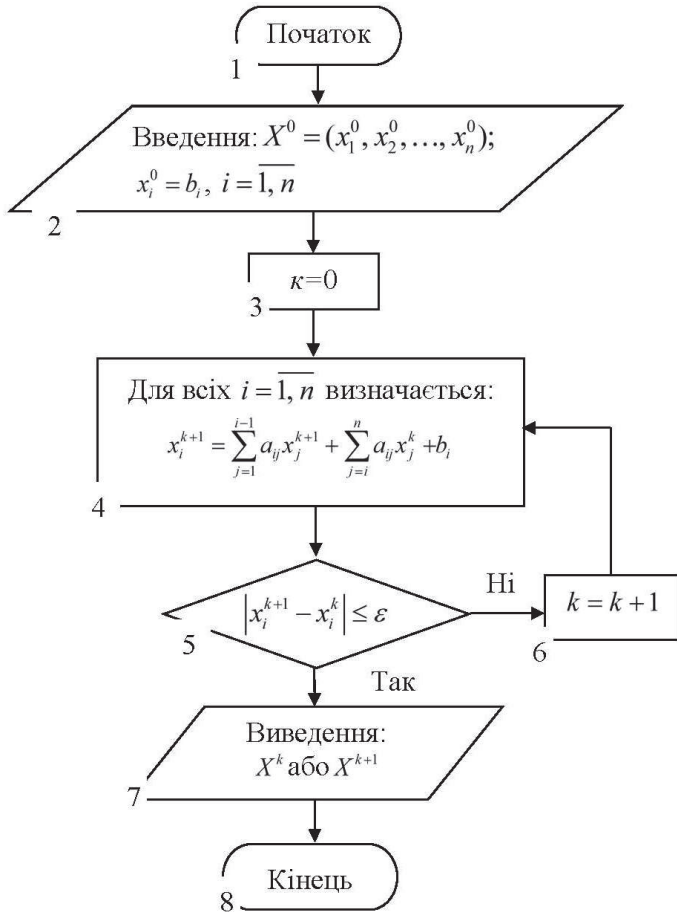


Рисунок 1.6 – Схема алгоритму методу Гаусса-Зейделя



Розглянемо реалізацію методу Гаусса-Зейделя мовою програмування C++:

```
public class GauseZeydel {
public static void GauseZeydelMethod()
{
Scanner scanner = new Scanner(System.in);
PrintWriter printwriter = new PrintWriter(System.out);
int size;
System.out.println("Введіть кількість невідомих"); size =
scanner.nextInt();
double[][] matrix = new double[size][size + 1]; System.out.println("Введіть
матрицю");
for (int i = 0; i < size; i++)
{
for (int j = 0; j < size + 1; j++)
{
matrix[i][j] = scanner.nextDouble();
}
}
double eps;
System.out.println("Введіть точність"); eps = scanner.nextDouble();
double[] previousVariableValues = new double[size]; for (int i = 0; i <
size; i++)
{
previousVariableValues[i] = 0.0;
}
while (true)
{
double[] currentVariableValues = new double[size]; for (int i = 0; i <
size; i++)
{
currentVariableValues[i] = matrix[i][size]; for (int j = 0; j < size; j++)
{
if (j < i)
{
currentVariableValues[i] -= matrix[i][j] * currentVariableValues[j];
}
if (j > i)
{
currentVariableValues[i] -= matrix[i][j] * previousVariableValues[j];
}
}
currentVariableValues[i] /= matrix[i][i];
}
double error = 0;
for (int i = 0; i < size; i++)
{
error += Math.abs(currentVariableValues[i] - previousVariableValues[i]);
}
if (error < eps)
{
break;
}
previousVariableValues = currentVariableValues;
}
for (int i = 0, j=1; i < size; i++, j++)
{
printwriter.print("x" + j + "=" + previousVariableValues[i] + " ");
}
scanner.close(); printwriter.close();
}
}.
```

## Висновки щодо застосування методів розв'язання СЛАР

У випадках невеликих порядків системи (до 6–7) можна використовувати метод Крамера, який дозволяє отримати точний розв'язок і який за таких порядків не потребує надто великої кількості обчислювальних операцій. Такі задачі часто виникають під час розрахунків невеликих електричних мереж чи підсистем автоматичного керування. Але із зростанням кількості рівнянь у системі дуже швидко починає зростати кількість операцій на обчислення визначників матриць, що призводить до спадання ефективності методу.

Починаючи з 9–10 порядку СЛАР використання методів Гаусса та його модифікацій мають переваги і є найбільш надійними методами. Але якщо матриці коефіцієнтів дуже розріджені, тобто містять багато нулів, реалізація методу Гаусса починає потребувати великих обчислювальних витрат на зміну порядку рядків матриць, щоб забезпечити ненульове значення головного елемента на певному кроці.

Задачі із розрідженими матрицями часто виникають у задачах обробки даних та математичної фізики, в яких для забезпечення точності проводиться достатньо щільна дискретизація з невеликим кроком. Узагалі, для задач розв'язання СЛАР великого порядку (десятки, сотні, тисячі і т. д.) ітераційні методи не мають конкурентів за умови забезпечення самої збіжності. Необхідно відзначити, що у багатьох практичних задачах умова збіжність забезпечується за самою постановкою задачі (наприклад, в процесі розв'язання рівняння Лапласа у задачах математичної фізики).

## Контрольні запитання та завдання

1. Чим відрізняються прямі та непрямі методи розв'язання СЛАР? Дати порівняльну оцінку.
2. Розкрити суть методів Крамера, Гаусса (та його різновидів), прогонки.
3. У якій формі подається СЛАР, для розв'язання яких використовуються ітераційні методи?
4. Розв'язати наведену нижче систему рівнянь методом Крамера. Скласти алгоритм і програму розв'язання.

$$\begin{cases} x_1 + x_2 + x_3 + x_4 + x_5 = 15; \\ x_1 - x_3 + 7x_4 = 26; \\ x_1 - 2x_2 + 3x_3 - 4x_4 = -10; \\ x_1 - x_2 + 2x_3 + 3x_5 = 20; \\ x_1 + x_3 - x_4 + 10x_5 = 50. \end{cases}$$

5. Розв'язати СЛАР з прикладу 4 методом Гаусса. Скласти обчислювальний алгоритм й програму.

6. Розв'язати СЛАР з прикладу 4 модифікованим методом Гаусса. Скласти обчислювальний алгоритм й програму.

7. Розв'язати методом Якобі (простих ітерацій) наведену нижче СЛАР. Скласти обчислювальний алгоритм й програму. Виконати перевірку умови збіжності.

$$\begin{cases} x_1 = \frac{x_2}{2} + \frac{x_3}{3} - \frac{x_4}{4}; \\ x_2 = \frac{x_1}{10} + \frac{x_2}{10} - \frac{x_3}{10} + \frac{x_4}{2}; \\ x_3 = \frac{x_1}{5} + \frac{x_2}{5} + \frac{x_3}{3} + \frac{x_4}{10}; \\ x_4 = \frac{x_3}{3} + \frac{x_4}{4} + 2. \end{cases}$$

8. Розв'язати методом Гаусса-Зейделя СЛАР з прикладу 4. Скласти обчислювальний алгоритм й програму. Виконати перевірку умови збіжності.

9. Як перевірити умову збіжності ітераційних алгоритмів обчислення СЛАР?

10. Чим відрізняються алгоритми ітераційних методів Якобі та Гаусса-Зейделя?

11. Розв'язати СЛАР методом Крамера:

$$\begin{cases} 2x_1 - x_2 + x_3 + 3x_4 = -1; \\ x_1 + x_2 - x_3 - 4x_4 = 6; \\ 3x_1 - x_2 + x_3 + x_4 = 4; \\ x_1 - 3x_2 + 3x_4 = -5. \end{cases}$$

12. Розв'язати СЛАР методом Гаусса:

$$\text{а) } \begin{cases} 0,542x + 2,43y - 3,75z - 0,208 = 0; \\ 2,31x - 3,68y - 4,51z + 4,08 = 0; \\ 13,4x + 2,36y - 9,75z - 36,4 = 0. \end{cases}$$

$$\text{б) } \begin{cases} 0,542x + 2,43y - 3,75z - 0,208 = 0; \\ 2,31x - 3,68y - 4,51z + 4,08 = 0; \\ 13,4x + 2,36y - 9,75z - 36,4 = 0. \end{cases}$$

$$в) \begin{cases} x_1 - 4x_2 - x_4 = 6; \\ x_1 + x_2 + 2x_3 + 3x_4 = -1; \\ 3x_1 - 5x_2 + 4x_3 = 0; \\ x_1 + 17x_2 + 4x_3 = 0. \end{cases}$$

13. Розв'язати систему рівнянь методом ітерацій:

$$а) \begin{cases} x = 0,12x - 0,06y + 0,09z + 0,32; \\ y = 0,17x + 0,11y - 0,07z - 0,21; \\ z = 0,14x - 0,15y - 0,08z + 0,18. \end{cases}$$

$$б) \begin{cases} x = 0,12x - 0,06y + 0,09z + 0,32; \\ y = 0,17x + 0,11y - 0,07z - 0,21; \\ z = 0,14x - 0,15y - 0,08z + 0,18. \end{cases}$$

$$в) \begin{cases} 60x - 2y + 6z = 30; \\ 10x - 80y - 4z = 20; \\ 12x + 6y - 90z = 45. \end{cases}$$

14. Розв'язати СЛАР ітераційним методом Гаусса-Зейделя із похибкою обчислень  $\Delta = 0,0001$ :

$$а) \begin{cases} 0,63x_1 + 1,00x_2 + 0,71x_3 + 0,34x_4 = 2,08; \\ 1,17x_1 + 0,18x_2 - 0,65x_3 + 0,71x_4 = 0,17; \\ 2,71x_1 - 0,75x_2 + 1,17x_3 - 2,35x_4 = 1,28; \\ 3,58x_1 + 0,28x_2 - 3,45x_3 - 1,18x_4 = 0,05. \end{cases}$$

$$б) \begin{cases} 7,6x_1 + 0,5x_2 + 2,4x_3 = 1,9; \\ 2,2x_1 + 9,1x_2 + 4,4x_3 = 9,7; \\ -1,3x_1 + 0,2x_2 + 5,8x_3 = -1,4. \end{cases}$$

15. Розв'язати СЛАР методом ітерацій із похибкою обчислень  $\Delta = 0,001$ :

$$а) \begin{cases} 8x_1 + x_2 + x_3 = 26; \\ x_1 + 5x_2 - x_3 = 7; \\ x_1 - x_2 + 5x_3 = 7. \end{cases}$$

$$б) \begin{cases} 7,6x_1 + 0,5x_2 + 2,4x_3 = 1,9; \\ 2,2x_1 + 9,1x_2 + 4,4x_3 = 9,7; \\ -1,3x_1 + 0,2x_2 + 5,8x_3 = -1,4. \end{cases}$$

16. Розв'язати СЛАР методом Гаусса із похибкою обчислень  $\Delta=0,001$ :

$$a) \begin{cases} 1,14x_1 - 2,15x_2 - 5,11x_3 = 2,05; \\ 0,42x_1 - 1,13x_2 + 7,05x_3 = 0,80; \\ -0,71x_1 + 0,83x_2 - 0,02x_3 = -1,07. \end{cases}$$

$$б) \begin{cases} 1,14x_1 - 2,15x_2 - 5,11x_3 = 2,05; \\ 0,42x_1 - 1,13x_2 + 7,05x_3 = 0,80; \\ -0,71x_1 + 0,83x_2 - 0,02x_3 = -1,07. \end{cases}$$

17. Розв'язати СЛАР, задану у матричному вигляді  $X=CX+d$  (табл. 1.3) методом простої ітерації із похибкою обчислень  $\Delta=0,001$ .

Таблиця 1.3 – Вихідні дані до завдання

№	C	d	№	C	d
1	2	3	1	2	3
1	$\begin{pmatrix} 0 & 0,3 & -0,1 & 0,2 \\ 0,2 & 0 & -0,21 & 0,2 \\ 0,3 & -0,1 & 0 & 0,3 \\ 0,3 & -0,1 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -4 \\ 2 \\ 0,1 \end{pmatrix}$	2	$\begin{pmatrix} 0 & 0,13 & -0,4 & 0,2 \\ 0,25 & 0 & -0,14 & 0,2 \\ 0,3 & -0,1 & 0 & 0,3 \\ 0,3 & -0,4 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -4 \\ 2 \\ 0,1 \end{pmatrix}$
3	$\begin{pmatrix} 0 & 0,27 & -0,1 & 0,2 \\ 0,2 & 0 & -0,26 & 0,2 \\ 0,3 & -0,1 & 0 & 0,5 \\ 0,2 & -0,1 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -4 \\ 2 \\ 0,1 \end{pmatrix}$	4	$\begin{pmatrix} 0 & 0,23 & -0,2 & 0,2 \\ 0,1 & 0 & -0,24 & -0,1 \\ 0,2 & -0,1 & 0 & 0,2 \\ 0,23 & -0,4 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -2 \\ 2 \\ 0,1 \end{pmatrix}$
5	$\begin{pmatrix} 0 & 0,3 & -0,1 & 0,2 \\ 0,2 & 0 & 0,1 & -0,2 \\ 0,3 & -0,1 & 0 & 0,3 \\ 0,3 & 0,1 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -4 \\ 2 \\ 0,1 \end{pmatrix}$	6	$\begin{pmatrix} 0 & 0,3 & 0,4 & 0,2 \\ 0,1 & 0 & -0,14 & 0,14 \\ 0,1 & 0,1 & 0 & 0,3 \\ 0,3 & -0,4 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -1 \\ 2 \\ 0,1 \end{pmatrix}$
7	$\begin{pmatrix} 0 & 0,3 & -0,4 & 0,2 \\ 0,1 & 0 & -0,14 & -0,1 \\ 0,1 & -0,1 & 0 & 0,3 \\ 0,3 & -0,4 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -1 \\ 2 \\ 0,1 \end{pmatrix}$	8	$\begin{pmatrix} 0 & 0,1 & -0,1 & 0,2 \\ 0,2 & 0 & -0,2 & 0,1 \\ 0,13 & -0,2 & 0 & 0,3 \\ 0,1 & -0,1 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -1 \\ 2 \\ 0,1 \end{pmatrix}$
9	$\begin{pmatrix} 0 & 0,1 & -0,4 & 0,2 \\ 0,15 & 0 & 0,1 & 0,2 \\ 0,3 & -0,1 & 0 & 0,3 \\ 0,1 & -0,14 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -2 \\ 2 \\ 0,1 \end{pmatrix}$	10	$\begin{pmatrix} 0 & 0,3 & -0,1 & 0,2 \\ 0,2 & 0 & 0,1 & -0,2 \\ 0,1 & -0,2 & 0 & 0,1 \\ 0,1 & 0,2 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -0,5 \\ 2 \\ 0,1 \end{pmatrix}$
11	$\begin{pmatrix} 0 & 0,3 & 0,1 & -0,2 \\ 0,2 & 0 & -0,1 & -0,2 \\ -0,1 & 0,2 & 0 & -0,1 \\ 0,1 & 0,2 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ 0,5 \\ -2 \\ 0,1 \end{pmatrix}$	12	$\begin{pmatrix} 0 & 0,3 & 0,14 & 0,2 \\ 0,11 & 0 & -0,41 & -0,1 \\ 0,1 & 0,1 & 0 & 0,13 \\ 0,13 & -0,4 & -0,2 & 0 \end{pmatrix}$	$\begin{pmatrix} -1 \\ -1 \\ 2 \\ 0,1 \end{pmatrix}$

18. Розв'язати СЛАР  $AX=B$  (табл. 1.4) методом Гаусса-Зейделя із похибкою обчислень  $\Delta=0,001$ :

Таблиця 1.4 – Вихідні дані до завдання

Ва-ріант	$A$	$B$	Ва-ріант	$A$	$B$
1	2	3	1	2	3
1	$\begin{pmatrix} 0,9 & 0,3 & -0,1 & 0,2 \\ 0,2 & -2 & 0 & -0,2 \\ 0,1 & -0,2 & 1 & -0,1 \\ 0,1 & 0,2 & -0,2 & -1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 5 \\ 2 \\ 0,1 \end{pmatrix}$	2	$\begin{pmatrix} 9 & 3 & -1 & 2 \\ 2 & -7 & 1 & -1 \\ 1 & -2 & 6 & -2 \\ 1 & 1 & -2 & -9 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 5 \\ 2 \\ 1 \end{pmatrix}$
3	$\begin{pmatrix} 1 & 0,1 & 0 & 0,2 \\ 0,2 & -1 & 0 & -0,2 \\ 0,1 & -0,2 & 1 & -0,1 \\ 0,1 & 0,2 & -0,2 & -1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 5 \\ 2 \\ 0,1 \end{pmatrix}$	4	$\begin{pmatrix} 7 & 3 & 0 & 2 \\ 1 & -4 & 0 & 1 \\ 1 & -1 & 6 & -2 \\ 1 & 2 & -2 & -9 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 6 \\ 2 \\ 11 \end{pmatrix}$
5	$\begin{pmatrix} 1 & 0,23 & -0,1 & 0,2 \\ 0,2 & -1 & 0 & 0,2 \\ 0,1 & -0,2 & 1 & 0 \\ 0,14 & 0,2 & -0,2 & -1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 2 \\ 2 \\ 1 \end{pmatrix}$	6	$\begin{pmatrix} 12 & 3 & -2 & 2 \\ 1 & -7 & 1 & -1 \\ 1 & -2 & 11 & -2 \\ 11 & 2 & -2 & -19 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 6 \\ 2 \\ 11 \end{pmatrix}$
7	$\begin{pmatrix} 2 & 0,3 & -0,1 & 0,2 \\ 0,2 & -3 & 0,1 & -0,2 \\ 0,1 & -0,52 & 2 & -0,1 \\ 0,1 & 0,2 & -0,2 & -2 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 5 \\ 2 \\ 0,1 \end{pmatrix}$	8	$\begin{pmatrix} 7 & 3 & -1 & 2 \\ 1 & -7 & 1 & 0 \\ 1 & 0 & 6 & -2 \\ 1 & 2 & -2 & -9 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 6 \\ 2 \\ 11 \end{pmatrix}$
9	$\begin{pmatrix} 4 & 0,2 & -0,1 & 0,3 \\ -0,1 & 3 & 0,2 & -0,3 \\ 0,1 & -0,5 & 2 & -0,1 \\ 0,3 & 0,2 & -0,2 & -2 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 2 \\ 5 \\ 0,1 \end{pmatrix}$	10	$\begin{pmatrix} 6 & 1 & -1 & 2 \\ 3 & 5 & 1 & 0 \\ 2 & 1 & -6 & 1 \\ 1 & 4 & 3 & -8 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 3 \\ 2 \\ 5 \end{pmatrix}$
11	$\begin{pmatrix} 1 & 0,13 & -0,15 & 0,24 \\ 0,2 & -1 & 0,11 & -0,42 \\ 0,11 & -0,2 & 1 & -0,11 \\ 0,1 & 0,12 & -0,42 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 5 \\ 2 \\ 0,1 \end{pmatrix}$	12	$\begin{pmatrix} 7 & 3 & -1 & 0 \\ 1 & -7 & 0 & -1 \\ 1 & -1 & 6 & -2 \\ 1 & 2 & 0 & -9 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 6 \\ -2 \\ 11 \end{pmatrix}$

## РОЗДІЛ 2 ЗАДАЧІ НЕЛІНІЙНОЇ МАТЕМАТИКИ

На сучасному етапі розвиток багатьох фундаментальних і прикладних наук тісно пов'язаний із застосуванням методів та засобів математичного й комп'ютерного моделювання. Ідеологія математичного моделювання передбачає формалізацію вихідного об'єкта-оригіналу за допомогою математичного і/або алгоритмічного опису (моделі), й дослідження поведінки об'єкта на основі програмно реалізованих обчислювальних експериментів. Це дозволило перейти від найпростіших розрахунків та оцінок різних конструкцій або процесів до нової стадії роботи – детального математичного моделювання (обчислювального експерименту), яке суттєво скорочує потребу в натурному експерименті, а в ряді випадків може їх взагалі замінити. В основі обчислювального експерименту покладено розв'язання математичної моделі чисельними методами. Часто математична модель зводиться до нелінійних рівнянь, алгебраїчних чи трансцендентних, причому наявність будь-якої відмінної від ступеневого поліному нелінійної функції перетворює рівняння у трансцендентні. На практиці задача зводиться до відшукування одного кореня на заданому інтервалі чи всіх існуючих коренів. Така проблема виникає, наприклад, під час оцінювання стійкості систем автоматичного управління. Прикладів виникнення таких задач в процесі проектування та дослідження об'єктів різної природи багато.

Зокрема, в такій галузі науки як балістика актуальною є задача визначення параметрів польоту артилерійського снаряда. Математична модель польоту снаряда подана рівняннями руху тіла за умови, що сила опору повітря руху снаряда пропорційна його швидкості ( $R = -kmv$ ):

– у параметричній формі (параметр  $t$  – час польоту снаряда) в проєкції на вертикальну вісь ординат  $y$ :

$$y = \frac{1}{k^2} (g + kv_0 \sin \alpha) (1 - e^{-kt}) - \frac{g}{k} t; \quad (2.1)$$

– у проєкціях на горизонтальну й вертикальну вісь  $x$ у:

$$y = \frac{1}{k} \left( \frac{g + kv_0 \sin \alpha}{v_0 \cos \alpha} \right) x + \frac{g}{k^2} \ln \left( 1 - \frac{k}{v_0 \cos \alpha} x \right), \quad (2.2)$$

де  $m$  – маса снаряда,  $\alpha$  – кут до горизонту, під яким снаряд вилетів із артилерійського ствола із початковою швидкістю  $v_0$ ,  $k$  – коефіцієнт пропорційності,  $t$  – час польоту снаряда,  $g = 9,82 \text{ м/с}^2$  – прискорення вільного падіння.

За допомогою рівняння (2.1) можна визначити загальний час польоту снаряда  $t$ , прирівнюючи ліву частину до нуля й розв'язуючи відповідне трансцендентне рівняння. Із рівняння (2.2), також прирівнюючи до нуля ліву частину і розв'язуючи відповідне трансцендентне рівняння, можна знайти або початкову швидкість  $v_0$  снаряда, або дальність польоту снаряда  $x = S$ .

Особливо актуальною нині є задача моделювання в сфері економіки, а саме – в галузі фінансів. Типовою задачею є визначення процентної ставки  $i$  дохідності облігацій. Математична модель дохідності подана таким трансцендентним рівнянням :

$$(I + P) \cdot (1 + i)^{-n} + P \cdot (1 + i)^{-n+1} + A_n \cdot (1 + i) - (A_n + I) = 0, \quad (2.3)$$

де  $A_n$  – поточна вартість облігації (ум. грош. од.),  $P$  – номінальна вартість (ум. грош. од.);  $N$  – термін, що пройшов після випуску облігації (в роках);  $T$  – термін погашення (в роках);  $n=T-N$  – термін, що залишився до погашення облігації (в роках);  $k$  – купонна процентна ставка (у частках від од.);  $I = P \cdot k$  – величина купонних платежів (добуток номінальної вартості  $P$  на купонну процентну ставку  $k$ ) (ум. грош. од.).

Відомою задачею в будівельній галузі є визначення критичної сили (втрата вертикальної рівноваги стержня), прикладеної вздовж стержня, один кінець якого закріпленй, а інший може переміщуватися у вертикальному напрямку, яка визначається рівнянням:

$$\operatorname{tg} \left( \sqrt{\frac{PL}{EI}} \right) - \sqrt{\frac{PL}{EI}} = 0, \quad (2.4)$$

де  $P$  – критична сила (Н),  $EI$  – згинальна жорсткість стержня (Н·м<sup>2</sup>),  $L$  – довжина стержня (м).

Метрологи часто розв'язують задачу відшукування нульової точки нелінійної вимірювальної чи градуовальної характеристики, що зводиться до розв'язання тих самих нелінійних рівнянь.

Подані та згадані математичні моделі (2.1) – (2.4) не можуть бути досліджені методами точного розв'язання рівнянь, оскільки змінні аргументи знаходяться під знаком трансцендентної функції чи нелінійної функції високого порядку (а відомо, що точні аналітичні формули для обчислення коренів існують тільки для алгебраїчних рівнянь не вище третього порядку). Проте точний розв'язок рівняння у технічних задачах не є безумовно необхідним. Задача пошуку коренів рівняння може вважатись практично вирішеною, якщо є можливість визначити корені із гарантованою точністю і вказати межі можливої похибки. Якщо для степеневих функцій існують точні аналітичні формули, то для трансцендентних рівнянь і будь-яких систем рівнянь таких методів взагалі не існує і потрібно користуватися тільки наближеними ітераційними методами та алгоритмами, найпоширеніші з яких буде розглянуто нижче.

У цьому розділі розглянуто розв'язання обчислювальних задач нелінійної математики: розв'язання трансцендентних рівнянь і систем нелінійних рівнянь, а також пошук комплексних коренів алгебраїчних рівнянь (рівнянь поліномів).



## 2.1 Узагальнена постановка задачі і процедура локалізації коренів рівняння

Дослідження математичних моделей (2.1) – (2.4) є задачею обчислення дійсних коренів рівнянь типу  $f(x) = 0$  на заданому відрізку  $[a; b]$ , де  $f: R_1 \mapsto R_2$  – алгебраїчна або трансцендентна функція. Вважається, що функція  $f(x)$  є будь-якою кусково-неперервною функцією дійсного аргументу, яка на відрізку  $[a; b]$  неперервна і має кусково-неперервну похідну.

Число  $x = \zeta$  називається коренем функції  $f(x)$ , якщо  $f(\zeta) \equiv 0$ .

Число  $\zeta$  називається коренем  $k$ -ої кратності, якщо за  $x = \zeta$  разом із функцією дорівнюють нулю усі її похідні до порядку  $k-1$  включно:

$$f(\zeta) = f'(\zeta) = \dots = f^{(k-1)}(\zeta) = 0, \text{ але } f^{(k)}(\zeta) \neq 0. \quad (2.5)$$

Під час визначення наближених значень коренів рівняння  $f(x) = 0$  необхідно розв'язати дві задачі:

1) відокремлення коренів, тобто визначення достатньо малих проміжків, в кожному із яких знаходиться один і тільки один корінь рівняння (простий або кратний);

2) уточнення коренів із наперед заданим числом правильних знаків.

У разі графічного відокремлення коренів рівняння  $f(x) = 0$  необхідно це рівняння перетворити до вигляду:

$$\varphi_1(x) = \varphi_2(x), \quad (2.6)$$

і побудувати графіки функцій:  $y_1 = \varphi_1(x)$ ,  $y_2 = \varphi_2(x)$ .

Дійсно, коренями рівняння  $f(x) = \varphi_1(x) - \varphi_2(x) = 0$  є абсциси точок перетину цих графіків.

Із усіх способів, якими можна рівняння  $f(x) = 0$  перетворити до вигляду (2.6), обирається той, який забезпечує найпростішу побудову графіків  $y_1 = \varphi_1(x)$  і  $y_2 = \varphi_2(x)$ . Зокрема, можна взяти  $\varphi_2(x) = 0$  і тоді прийдемо до побудови графіка функції  $y = f(x)$ , точки перетину (або дотику) якої з прямою  $y_2 = \varphi_2(x)$ , тобто із віссю абсцис, і є шуканими коренями рівняння  $f(x) = 0$ .

У загальному випадку під час побудови графіків:

$$y_1 = \varphi_1(x); \quad y_2 = \varphi_2(x), \quad (2.7)$$

насамперед потрібно визначити поведінку кожної із функцій  $\varphi_1(x)$  і  $\varphi_2(x)$  за умови  $x \rightarrow -\infty$  і  $x \rightarrow +\infty$ , знайти значення  $x$ , за яких  $\varphi_j(x) = \infty$  ( $j=1, 2$ ), визначити точки перетину цих функцій з осями  $x$  і  $y$  і обчислити ряд проміжних, найхарактерніших значень, починаючи із значень  $\varphi_j(x)$  для  $x = \pm 1$ , за яких зазвичай простіше обчислюються значення будь-якої функції.

**Приклад 2.1.** Відокремити дійсні корені рівняння:

$$\frac{x+1}{x-1} - \sin x = 0. \quad (2.8)$$

### Розв'язання:

Записавши рівняння (2.8) у вигляді  $\sin x = \frac{x+1}{x-1}$ , будемо графіки  $y_1 = \sin(x)$ ,  $y_2 = \frac{x+1}{x-1}$ , абсиси точок перетину яких і визначають собою усі дійсні корені вихідного рівняння (рис. 2.1). У такому випадку усі корені від'ємні, оскільки права частина гіперболи  $y_2 = \frac{x+1}{x-1}$  ніде не перетинає синусоїду  $y_1 = \sin(x)$ . Хоча кількість коренів нескінченна, усі вони ізольовані, оскільки тільки один корінь зустрічається в кожному із проміжків  $(0; -\pi/2)$ ,  $(-\pi/2; -3\pi/2)$ , ...,  $((1-2k)\pi/2; (-1-2k)\pi/2)$ , де  $k = 1, 2, 3, \dots$ .

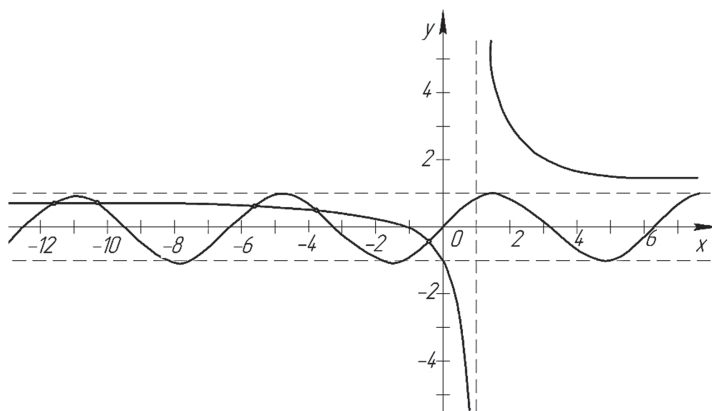


Рисунок 2.1 – Діаграма графічного визначення коренів рівняння

Із аналітичних методів відокремлення коренів використовують два найпоширеніші загальні методи, однаково придатні як для алгебраїчних, так і для трансцендентних рівнянь. Один з них – пошук простішого рівняння, що має корені, які приблизно дорівнюють невідомим кореням цього рівняння. Це дуже часто можна реалізувати, нехтуючи в заданому рівнянні малими членами. У другому методі використовуються теореми, які безпосередньо впливають із відомих властивостей неперервних функцій.

**Теорема 2.1** (теорема про корінь неперервної функції). Якщо функція  $f(x)$  неперервна на відрізку  $[a; b]$ , причому значення її на кінцях відрізка  $f(a)$  і  $f(b)$  мають протилежні знаки, то між точками  $a$  і  $b$  є хоча б один дійсний корінь рівняння  $f(x) = 0$ .

**Теорема 2.2.** Якщо функція  $f(x)$  строго монотонна на відрізку  $[a; b]$ , тобто строго зростає або спадає на  $[a; b]$ , тоді на цьому відрізку рівняння  $f(x) = 0$  не може мати більше одного кореня.

**Приклад 2.2.** Для функції  $f(x) = x^3 - 4x + 2$  необхідно знайти інтервали. Розв'язуючи нерівність  $f'(x) = 3x^2 - 4 > 0$ , отримаємо:  $x \in \left(-\infty; -\frac{2}{\sqrt{3}}\right) \cup \left(\frac{2}{\sqrt{3}}; +\infty\right)$ . На цих двох інтервалах функція зростає.

Зрозуміло, що на інтервалі  $x \in \left(-\frac{2}{\sqrt{3}}; \frac{2}{\sqrt{3}}\right)$  функція спадає.

Значення функції в точках екстремуму:

$$f\left(-\frac{2}{\sqrt{3}}\right) = 2 + \frac{16}{3\sqrt{3}} > 0, \quad f\left(\frac{2}{\sqrt{3}}\right) = 2 - \frac{16}{3\sqrt{3}} = \frac{8(\sqrt{3}-2)}{3\sqrt{3}} < 0$$

означає, що на відрізку спадання  $\left[-\frac{2}{\sqrt{3}}; \frac{2}{\sqrt{3}}\right]$  відокремлений корінь  $\xi^{**}$ .

Оскільки очевидно, що  $f(x) \rightarrow -\infty$  за  $x \rightarrow -\infty$  і  $f(x) \rightarrow +\infty$  за  $x \rightarrow +\infty$ , то є ще два кореня:  $\xi^* \in \left(-\infty; -\frac{2}{\sqrt{3}}\right)$  і  $\xi^{***} \in \left(\frac{2}{\sqrt{3}}; +\infty\right)$ .

Для відокремлення цих коренів додатково обчислюється, наприклад, значення в точках  $-3$  і  $3$ :  $f(-3) = -13 < 0$  і  $f(3) = 17 > 0$ . Отже, отримано такі відрізки, на яких відокремлені корені:

$$\xi^* \in \left[-3; -\frac{2}{\sqrt{3}}\right]; \quad \xi^{***} \in \left[\frac{2}{\sqrt{3}}; 3\right].$$

Функція  $f(x)$  змінює знак під час переходу через корінь  $\xi^*$ , тобто якщо  $f'(\xi^*) \neq 0$ .

У загальному випадку, якщо зазначені процедури ускладнені для аналізу, уся область визначення або певна її частина, яка з деяких міркувань викликає інтерес, розбивається на відрізки точками  $x_i$ , розташованими на умовно незначній відстані  $h$ . Визначивши значення у всіх цих точках, перевіряється виконання умови  $f(x_{i-1}) \cdot f(x_i) \leq 0$ . Якщо попередньо відомо число коренів у досліджуваній області, тоді, звужуючи крок  $h$  пошуку, можна або їх усі локалізувати, або стверджувати, що можлива наявність пар коренів, не визначених з точністю  $h = \varepsilon$ .

## 2.2 Чисельні методи розв'язання нелінійних алгебраїчних рівнянь

Після того, як дослідження рівняння  $f(x) = 0$  закінчено і для кожного дійсного кореня  $\xi$  встановлено інтервал, в якому цей корінь знаходиться, переходить до розв'язання другої задачі – уточнення знайдених коренів.

Відокремлення кореня  $\xi$ , тобто встановлення подвійної нерівності  $a < \xi < b$ , само собою дає можливість отримати його грубе наближене значення, як таке можна узяти, наприклад, центр інтервалу  $[a; b]$ . У такому

разі абсолютна похибка буде меншою числа  $\varepsilon < \frac{|a-b|}{2}$ .

Підставивши знайдене  $\xi_1$  в рівняння  $f(x)=0$  і переконавшись, що воно не забезпечує необхідну точність, вибирається  $f(\xi_1) = a_1$  або  $f(\xi_1) = b_1$  і як результат буде виконуватись більш точна нерівність:  $a_1 < \xi < b$  або  $a < \xi < b_1$ .

### Метод половинного ділення (дихотомії)

Припустимо, що корінь  $\xi$  рівняння  $f(x)=0$  визначений на відрізку  $[a; b]$  і знаки функцій  $f(a)$  і  $f(b)$  у цьому разі різні (функція  $f(x)$  змінює знак в процесі переходу через значення кореня  $\xi$ ).

Покладемо, що  $a_0 = a$  і  $b_0 = b$  та обчислимо значення функції в лівому кінці відрізка  $f(a_0)$ , а також у його середині  $f(c_0)$ , де  $c_0 = \frac{a_0 + b_0}{2}$ . Порівняємо знаки чисел функцій  $f(a_0)$  і  $f(c_0)$ , якщо ці знаки різні, тоді значення кореня  $\xi$  лежить в інтервалі  $[a_0; c_0]$ ; якщо однакові, тоді різні знаки значення функцій  $f(c_0)$  і  $f(b_0)$ , і корінь лежить в інтервалі  $[c_0; b_0]$ . Можливий випадок коли  $f(c_0) = 0$ , тоді значення кореня  $\xi = c_0$  вже буде вважатись знайденим. В обох випадках зміни знака значення кореня буде знаходитися на відрізку  $[a_0; c_0]$  або  $[c_0; b_0]$ , значення довжини якого буде в два рази меншим за довжину вихідного відрізка  $[a_0; b_0] = [a; b]$ . Цей відрізок позначається як половина довжини через  $[a_1; b_1]$  (тобто покладається  $a_1 = a_0, b_1 = c_0$  у випадку, коли значення  $f(a_0)$  і  $f(c_0)$  різних знаків;  $a_1 = c_0, b_1 = b_0$  у випадку, коли  $f(a_0)$  і  $f(c_0)$  одного знака).

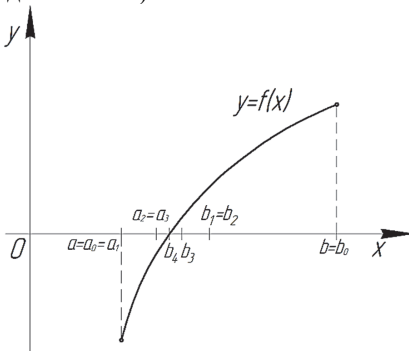


Рисунок 2.2 – Схема половинного поділу відрізка і наближення до значення кореня  $\xi$

Після чого типовий процес повторюється для відрізка  $[a_1; b_1]$ , а саме – відбувається пошук середини  $c_1$  і пошук значення функції  $f(c_1)$  із одночасним порівнянням знака цього числа зі знаком значення функції  $f(a_1)$ . Якщо ці знаки різні, тоді значення кореня лежить на відрізку  $[a_2; b_2] = [a_1; b_1]$ ; якщо однакові, тоді знаходиться на  $[a_2; b_2] = [c_1; b_1]$  (у випадку якщо  $f(c_1) = 0$ , то значення кореня буде вважатись знайденим). Водночас довжина відрізка, на якому знаходиться корінь, зменшилась ще в два рази.

Повторюючи типовий процес, отримаємо, що після  $k$  поділів довжина відрізка, на якому знаходиться корінь, скорочується в  $2^k$  разів і дорівнює  $\delta_k = \frac{b-a}{2^k}$ , якщо значення кореня  $\xi$  не було точно визначено на якомусь попередньому етапі, тобто не збігається зі значенням  $c_i$  за деякого  $i$ .

Алгоритм методу половинного ділення (дихотомії) наведено на рисунку 2.3.

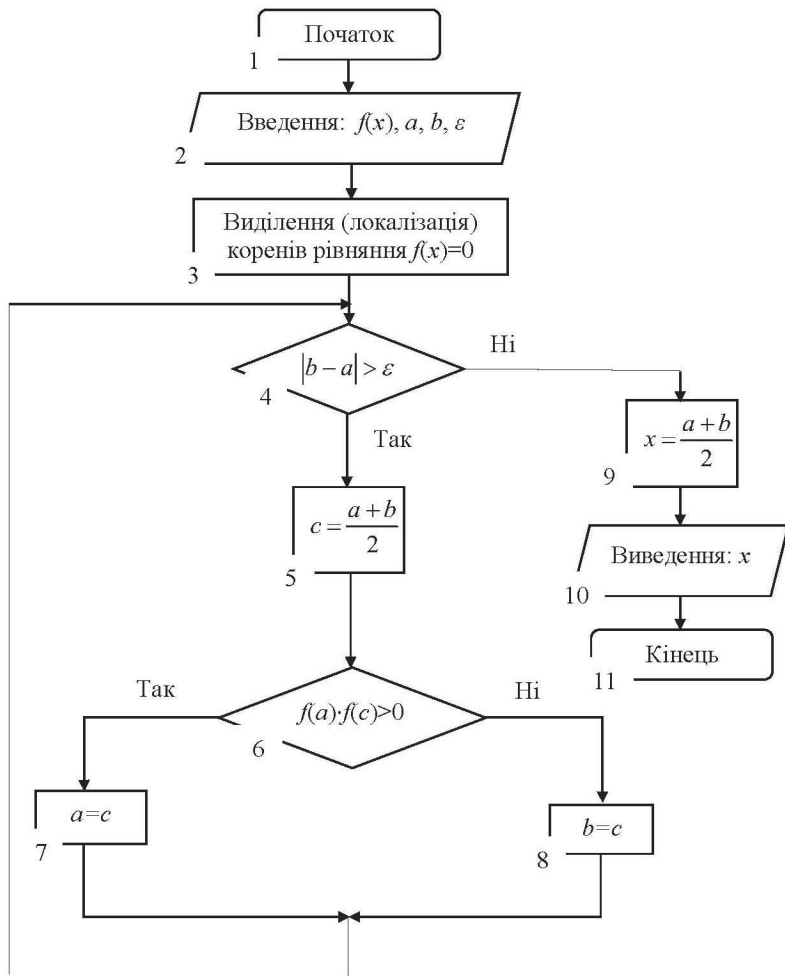


Рисунок 2.3 – Схема алгоритму методу половинного ділення (дихотомії)

**Приклад 2.3.** Розв'язати рівняння чисельним методом половинного ділення (дихотомії) із точністю  $\varepsilon = 0,001$ :

$$x^3 + 2x^2 + 3x + 5 = 0.$$

## Розв'язання:

Розв'язання починається із використання методу половинного ділення на відрізку  $[-2; -1]$ , на якому методом графічної побудови графіка функції  $\varphi(x) = x^3 + 2x^2 + 3x + 5$  відокремлено корінь  $\zeta$  (рис. 2.4).

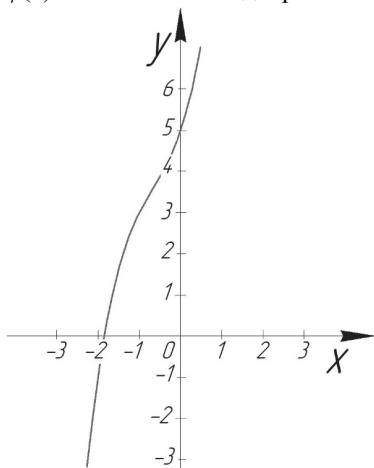


Рисунок 2.4 – Графічне визначення інтервалу належності коренів рівняння

Послідовно визначаються значення функції в серединах отриманих відрізків:  $f(-1,5) = 1,625$ ;  $f(-1,75) = 0,515625$ ;  $f(-1,875) = -0,185547$ ; ...;  $f(-1.841797) = 0.011269$ , після чого обчислення зупиняється на дев'ятому кроці, оскільки черговий відрізок має довжину  $\frac{1}{2^9} = \frac{1}{512} < 2\varepsilon = \frac{1}{500}$ . У цьому

випадку середина останнього відрізка – це точка  $-1,842773$ . Було отримано, що наближене значення  $\bar{x}$  кореня  $\zeta$  із точністю до 0,001 дорівнює  $\bar{x} \approx -1,843$ .

Відзначимо, що метод поділу відрізка навпіл, як і метод простого перебору, не висуває жодних вимог до гладкості функції (тобто до існування її похідної): достатньо, щоб функція була неперервною.

Розглянемо реалізацію методу половинного ділення (дихотомії) мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
x = np.arange(-5, 3, 0.1)
#визначаємо функцію
func_x=(x**3)+(2*x**2)+(3*x)+5
#будуємо графік і локалізуємо корені рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
plt.plot(x, func_x)
plt.legend(['f(x)=x^3+2x^2+3x+5'], loc=1)
plt.grid(True)
plt.xlim([-4, 1])
plt.ylim([-5, 5])
plt.show()
#задаємо початкові межі локалізації кореня рівняння
a=-2; b=-1; e=0.001
#задаємо саму функцію
def f(x):
    return (x**3)+(2*x**2)+(3*x)+5
from timeit import default_timer as timer
```

```

#визначаємо корінь рівняння
i=0
start= timer()
while abs(b-a)>=e:
    i=i+1
    z = (a+b)/2
    if f(a)*f(z)<=0:
        b=z
    else:
        a=z
end = timer()
print("x =", z, "Time taken:", end-start, 'Number of iterations:', i).

```

## Метод Ньютона (дотичних)

Метод Ньютона, або метод дотичних, належить до одного з найбільш використовуваних методів уточнення коренів, однаково придатних як для алгебраїчних, так і для трансцендентних рівнянь.

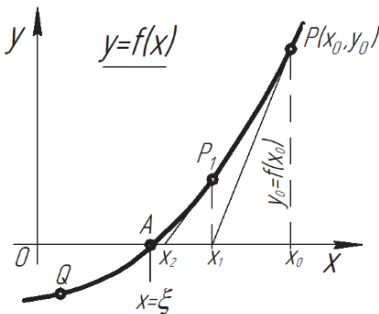


Рисунок 2.5 – Схема визначення коренів рівняння методом Ньютона

через точку  $P(x_0, y_0)$ :

$$y - y_0 = f'(x_0)(x - x_0). \quad (2.9)$$

Поклавши в рівнянні (2.9)  $y = 0$ ,  $y_0 = f(x_0)$ , визначаємо точку перетину із віссю абсцис ( $y = 0$ ), яку позначимо через  $x_1$ :

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}. \quad (2.10)$$

Через точку  $P_1(x_1, y_1=f(x_1))$  знову необхідно провести дотичну і, продовжуючи цей процес, приходимо до формули Ньютона:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \quad (n = 0, 1, 2, \dots), \quad (2.11)$$

яка дає можливість крок за кроком обчислювати усе точніші значення

кореня. Іншими словами, значення  $x_0, x_1, x_2, \dots$ , обчислені за формулою (2.11), утворюють послідовність, яка наближається до значення кореня  $f(x) = 0$ .

Якщо ми почнемо процес, виходячи із точки  $Q$ , в якій крива обернена до осі  $Ox$  увігнутістю, тоді перший крок приведе на іншу сторону від осі  $Ox$ , де крива обернена до неї випуклістю, так що в подальшому будемо наближатися до значення кореня так само, як і раніше.

У тих випадках, коли обчислення другої похідної для функції  $f(x)$  обернена випуклістю до осі  $Ox$  у тих точках, для яких виконується співвідношення:

$$f(x) \cdot f''(x) > 0, \quad (2.12)$$

тоді цю умову і має задовольняти обране початкове значення  $x_0$ .

В процесі обчислення кореня за методом Ньютона похибка кожного нового наближення зменшується пропорційно квадрату похибки попереднього наближення.

Алгоритм методу Ньютона (дотичних) наведено на рисунку 2.6.

**Приклад 2.4.** Обчислити найменший додатний корінь рівняння:

$$e^x - 3x = 0 \quad (2.13)$$

із точністю  $\varepsilon=0,0001$ .

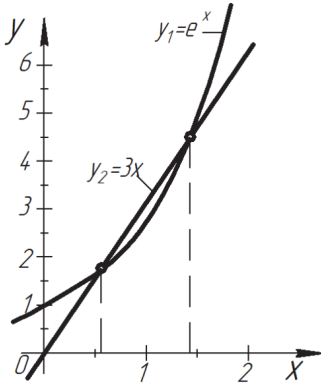


Рисунок 2.7 – Графічне визначення інтервалу належності коренів рівняння

*Розв'язання:*

Для того, щоб відокремити корені, рівняння (2.13) зводиться до вигляду  $e^x = 3x$  і будуються графіки функцій  $y_1 = e^x$  і  $y_2 = 3x$  (рис. 2.7), причому масштаби по осях  $Ox$  і  $Oy$  можуть бути різними.

Згідно з рисунком 2.7 найменший додатний корінь лежить в інтервалі  $[0; 1]$ . Переходячи до уточнення у цьому прикладі необхідно визначити:

$f(x) = e^x - 3x$ ;  $f'(x) = e^x - 3$ ;  $f''(x) = e^x$ ,  
тоді формула (2.11) набуває вигляду:

$$x_{n+1} = x_n - \alpha_n, \quad (2.14)$$

де  $\alpha_n = \frac{f(x_n)}{f'(x_n)}$



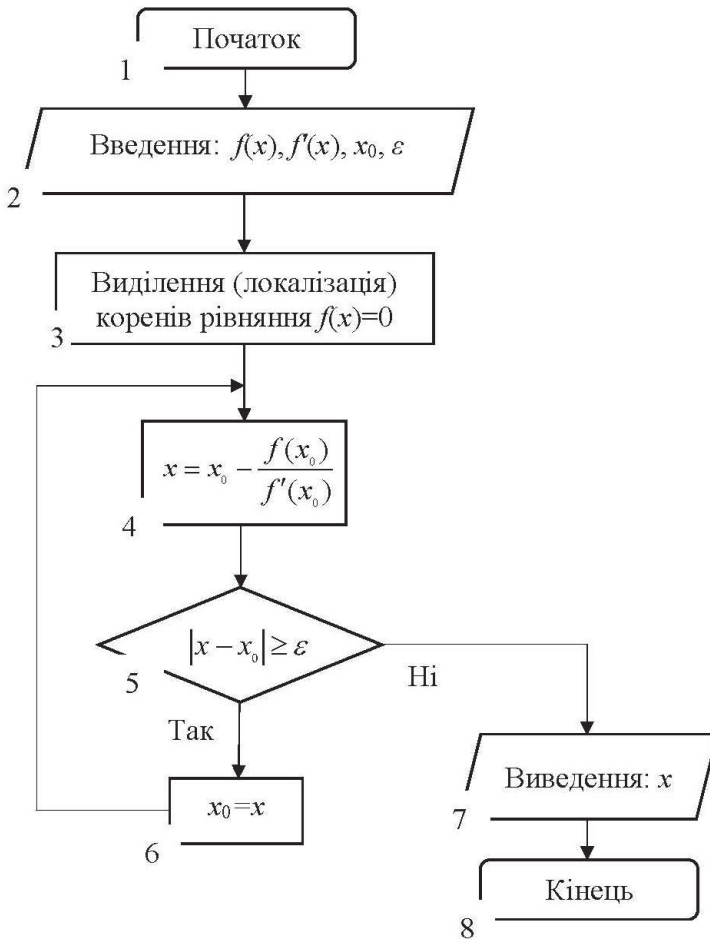


Рисунок 2.6 – Схема алгоритму  
метода Ньютона (дотичних)

Із двох можливих значень  $x_0 = 0$  або  $x_0 = 1$  необхідно обрати перше, оскільки згідно з критерієм (2.12):

$$f(x) \cdot f''(x) \Big|_{x=0} = (e^x - 3x)e^x \Big|_{x=0} = 1 > 0,$$

у той час як

$$f(x) \cdot f''(x) \Big|_{x=1} = (e^x - 3x)e^x \Big|_{x=1} = (e - 3)e < 0.$$

Усі подальші наближення, обчислені за формулою (2.14), наведено у таблиці 2.1. Таким чином, невідомий корінь із необхідною точністю  $\varepsilon = 0,0001$  є  $x_3 = 0,6191$ .

Таблиця 2.1 – Результати розв’язання рівняння

$n$	$x=x_n$	$e^x$	$3x$	$f(x_n)$	$f'(x_n)$	$\alpha_n$
0	0,10000	1,0000	0,0000	1,0000	-2,00	-0,500000
1	0,50000	1,6500	1,5000	0,1500	-1,35	-0,110000
2	0,61000	1,8404	1,8300	0,0104	-1,16	-0,009000
3	0,61900	1,8571	1,8570	0,0001	-1,14	-0,000088
4	0,61909	1,8573	1,8573	0,0000	-	-

Виконавши ще один крок обчислень, отримуємо більш точне значення цього кореня  $x_4 = 0,61906129$ .

Під час обчислення коренів рівняння  $f(x) = 0$  методом Ньютона процес послідовних наближень завжди збігається, якщо нульове наближення  $x_0$  взято настільки близько до кореня  $x = \zeta$ , що на сегменті  $[\zeta, x_0]$ :

- 1) нахил кривої  $y = f(x)$  не дорівнює нулю, тобто  $f'(x) \neq 0$ ;
- 2) крива  $y = f(x)$  не має точок перегину, тобто  $f''(x) \neq 0$ .

Практично це означає, що за формулою (2.11) корінь рівняння  $f(x) = 0$  можна обчислити із будь-яким як завгодно високим ступенем точності, якщо тільки корінь  $x = \zeta$  не є кратним ( $f'(x) \neq 0$ ) і якщо нульове наближення  $x_0$  взято достатньо близько до шуканого кореня  $\zeta$ .

Кратні корені можуть також визначатись за формулою Ньютона, як такі, що відповідають кореням рівняння  $f'(x) = 0$ , або, в більш загальному випадку,  $f^{(k)}(x) = 0$  ( $k = 1, 2, 3, \dots$ ).

Розглянемо реалізацію методу Ньютона (дотичних) мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
x = np.arange(-5, 5, 0.1)
#визначаємо функцію
value_funcm=[]
for i in x:
    value_funcm.append(math.exp(i)-(3*i))
M = np.array(value_funcm)
#будуємо графік і локалізуємо корені рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontSize=15, color='blue')
plt.ylabel('y',fontSize=15, color='blue')
```

```

plt.plot(x, M)
plt.legend(['f(x)=exp(x)-3*x'], loc=1)
plt.grid(True)
plt.xlim([0, 2])
plt.ylim([-2, 2])
plt.show()
#задаємо саму функцію
def f(x):
    return math.exp(x)-(3*x)
from sympy import *
#визначаємо першу похідну функції
x_arg = Symbol('x_arg')
func_m=exp(x_arg)-(3*x_arg)
m_first=func_m.diff(x_arg)
func_m = lambdify(x_arg, m_first)
#визначаємо другу похідну функції
m_second=m_first.diff(x_arg)
func_mm = lambdify(x_arg, m_second)
#визначаємо першу похідну функції
print("Перша похідна функції f'(x)=",m_first,". Друга похідна функції
      f''(x)=",m_second)
#задаємо початкову ітераційну точку і точність визначення кореня рівняння
x_n=0; e=0.001
if f(x_n)*func_mm(x_n)<0:
    print("Invalid starting point value. Choose another point value!")
from timeit import default_timer as timer
#визначення ітераційної формули Ньютона
def x_nn(x):
    return x_n-(f(x_n)/func_m(x_n))
k=0
#визначаємо корінь рівняння
start= timer()
while abs(x_nn(x_n)-x_n)>=e:
    k=k+1
    x_n=x_nn(x_n)

end = timer()
print("x =", x_n, "time taken:", end-start, "Number of iterations:", k).

```

## Метод хорд (лінійної інтерполяції)

Ідея методу полягає в тому, що по двох точках  $M_{i-1}(x_{i-1}, f(x_{i-1}))$  і  $M_i(x_i, f(x_i))$  необхідно побудувати пряму  $M_{i-1}M_i$  (хорда, яка з'єднає дві точки функції  $y=f(x)$  на декартовій осі координат) і взяти як наступне наближення  $x_{i+1}$  абсцису точки перетину цієї прямої із віссю  $Ox$ . Тобто виконується заміна на цьому кроці функції  $f(x)$  її лінійною інтерполяцією, визначеною за двома значеннями  $x$ :  $x_{i-1}$  і  $x_i$ . Буде вважатись, що лінійною інтерполяцією функції  $f(x)$  називається така лінійна функція  $l(x)$ , значення якої збігаються із значеннями  $f(x)$  в двох фіксованих точках, у цьому випадку – в точках  $x_{i-1}$  і  $x_i$ .

Залежно від того, чи лежать точки  $x_{i-1}$  і  $x_i$  по різні сторони від кореня  $\zeta$  або по одну і ту саму сторону, будуть отримані такі принципові схеми на визначення коренів рівняння в декартовій системі координат, які показані на рисунку 2.8.

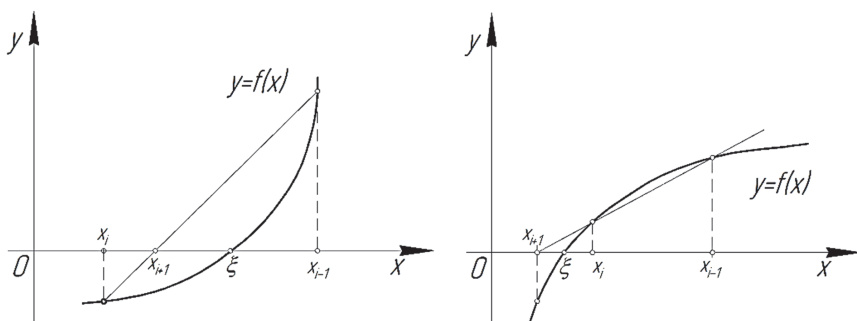


Рисунок 2.8 – Схеми визначення коренів рівняння методом хорд

Чергове послідовне наближення буде залежати від двох попередніх  $x_{i+1} = \varphi(x_{i-1}, x_i)$ . Інтерполяційна лінійна функція  $l(x)$  буде визначатись як функція із кутовим коефіцієнтом, який дорівнює різниці до відношенню:

$$k_i = \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}, \quad (2.15)$$

побудованому для відрізка між  $x_{i-1}$  і  $x_i$ , графік якої проходить через точку  $M_i$ :

$$l(x) = f(x_i) + \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}(x_i - x_{i-1}). \quad (2.16)$$

Розв'язуючи рівняння (2.16) за умови, що  $l(x) = 0$ , визначимо:

$$x_{i+1} = \frac{x_{i-1}f(x_i) - x_i f(x_{i-1})}{f(x_i) - f(x_{i-1})} = x_i - \frac{f(x_i)}{\frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}} \quad (2.17)$$

або

$$x_{i+1} = x_i - \frac{f(x_i)}{k_i}. \quad (2.18)$$

Величина  $k_i$  може розглядатись як різничеве наближення для похідної  $f'(x)$  в точці  $x_i$ . Тобто отримана формула (2.17) – це різницевий аналог ітераційної формули методу Ньютона.

Обчислення за формулою (2.18) набагато прийнятніше, ніж обчислення за формулою (2.17), хоча обидва вони математично тотожні, оскільки за використання формули (2.18) у разі обчислень з округленнями досягається менша втрата значущих цифр.

Існує два способи застосування формули (2.18). Перший спосіб: обчислення ведеться безпосередньо за формулою (2.18) для  $i = 1, 2, 3, \dots$ , починаючи із двох наближень  $x_0$  і  $x_1$ , взятих якомога ближче до кореня  $\xi$ . Водночас не передбачається, що  $\xi$  лежить між  $x_0$  і  $x_1$  (значення функції  $f(x)$  в точках  $x_0$  і  $x_1$  мають різні знаки), а також не гарантується, що корінь

потрапить на відрізок між  $x_{i-1}$  і  $x_i$  на якому-небудь наступному кроці (хоча це і не виключено). У такому випадку важко дати оцінку похибки, з якою  $x_{i+1}$  наближує істинне значення кореня  $\xi$ , і тому задовольняються таким емпіричним правилом: обчислення зупиняються, коли буде виконана нерівність  $|x_{i+1} - x_i| < \varepsilon$ , де  $\varepsilon$  – встановлена точність знаходження кореня рівняння. У такому випадку вибирається наближене значення кореня, яке дорівнює  $\xi = x_{i+1}$ .

Зазначимо, що достатні умови, які забезпечують обчислення кореня рівняння  $f(x) = 0$  за методом хорд із довільним заданим ступенем точності, будуть такими самими, що і за методом Ньютона. Але метод Ньютона у більшості випадків дає кращу збіжність, ніж метод хорд.

У тих випадках, коли обчислення  $f(x)$  та її похідної  $f'(x)$  трудомістке, метод хорд дає більшу економію затрат. У такому разі головним фактором, який впливає на прискорення збіжності процесу, є величина сегмента  $[x_i; \xi]$ , що розглядається, де  $\xi$  – істинне значення кореня. Тому за інших однакових умов необхідно віддавати перевагу сегменту  $[x_{n-1}; x_n]$ , меншому в абсолютній величині.

Алгоритм методу хорд (лінійної інтерполяції) наведено на рисунку 2.9.

**Приклад 2.5.** Визначити значення дійсного кореня рівняння:

$$x^3 - 2x - 5 = 0 \quad (2.19)$$

із точністю  $\varepsilon=0,00001$ .

*Розв'язання:*

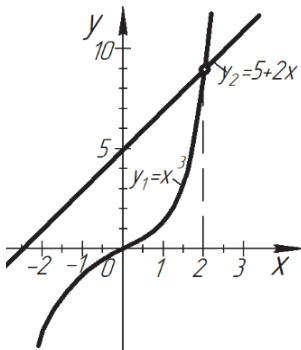


Рисунок 2.10 – Схема графічного визначення інтервалу належності коренів рівняння

Рівняння (2.19) для початку доцільно подати у вигляді  $y_1=x^3$ ,  $y_2=5+2x$ . Тоді графік на рисунку 2.10 дозволяє обрати два вихідних значення, необхідних для методу хорд:  $x_1=2,2$  і  $x_2=2,0$ . Використавши формулу (2.17), покроково визначаємо корінь із необхідною точністю. Водночас  $x_1=2,200000$  має фіксоване значення, а  $x_2$  уточнюється в кожному наступному кроці. Числа  $x_1$  і  $f(x_1)=1.248$ , які використовуються у всіх наступних кроках, можна вважати точними і вибирати в них стільки десяткових знаків, скільки необхідно в цьому кроці. Усі необхідні обчислення наведено в таблиці 2.2, згідно з якою із заданою точністю визначається корінь рівняння  $x = 2,09455$ .

Таблиця 2.2 – Результати розв’язання рівняння

$n$	$x=x_n$	$x_n^3$	$f(x_n)$	$f(x_i) - f(x_{i-1})$
0	2,200000	10,648000	1,248000	–
1	2,000000	8,000000	– 1,000000	2,248000
2	2,090000	9,129000	– 0,051000	1,299000
3	2,094300	9,185790	– 0,002810	1,250810
4	2,094530	9,188820	– 0,000240	1,248240
5	2,094550	9,189084	– 0,000016	–

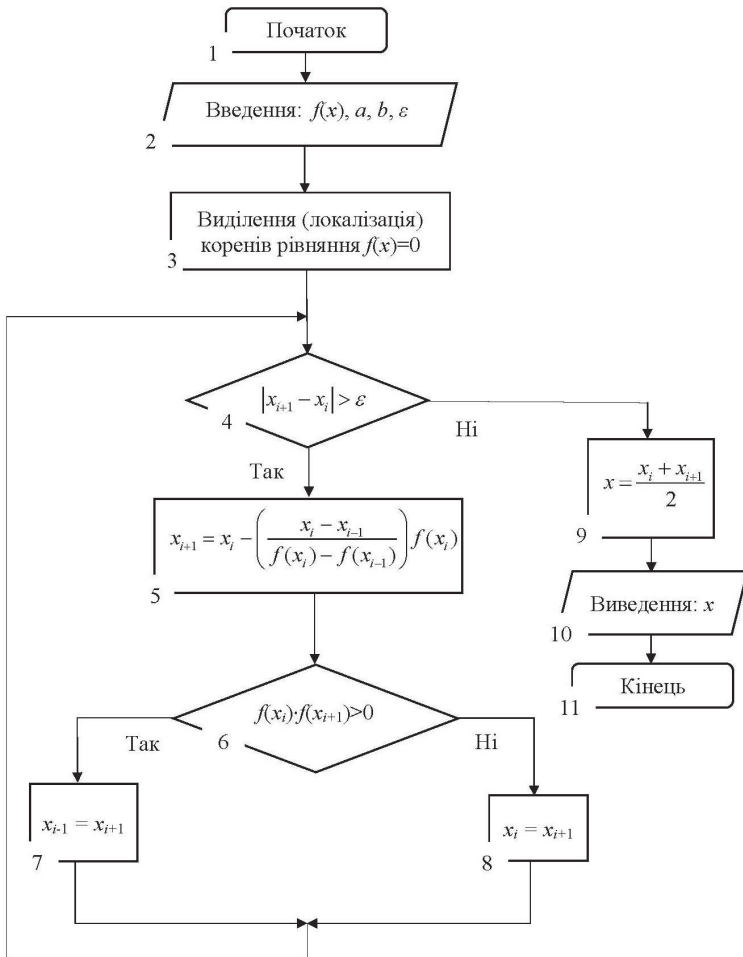


Рисунок 2.9 – Схема алгоритму методу хорд (лінійної інтерполяції)

Розглянемо реалізацію методу Ньютона (дотичних) мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
x = np.arange(-5, 4, 0.1)
#визначаємо функцію
func_x=(x**3)-(2*x)-5
#будуємо графік і локалізуємо корені рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
plt.plot(x, func_x)
plt.legend(['f(x)=x^3-2x-5'], loc=1)
plt.grid(True)
plt.xlim([-2, 4])
plt.ylim([-7, 4])
plt.show()
#задаємо саму функцію
def f(x):
    return (x**3)-(2*x)-5
from timeit import default_timer as timer
#задаємо початкові межі локалізації кореня рівняння
x_n=1; x_nm=2; e=0.000001
#визначення ітераційної формули метода хорд
def x_np(x_n, x_nm):
    k=(f(x_n)-f(x_nm))/(x_n-x_nm)
    return x_n-(f(x_n)/k)
k=0
#визначаємо корінь рівняння
start= timer()
while abs(x_np(x_n, x_nm)-x_n)>=e:
    k=k+1
    x_n=x_np(x_n, x_nm)
    if f(x_n)*f(x_np(x_n, x_nm))>0:
        x_nm=x_np(x_n, x_nm)
    else:
        x_n=x_np(x_n, x_nm)
end = timer()
print("x =", (x_n+x_np(x_n, x_nm))/2,"Time taken:", end-start, "Number of
iterations:", k).
```

## Метод простих ітерацій

Алгебраїчне або трансцендентне рівняння  $f(x)=0$  може бути зведене до такого вигляду:

$$x = \varphi(x),$$

що можна виконати різними шляхами і отримати різні вирази для функції  $\varphi(x)$ .

Для наближеного значення кореня  $x_0$  визначається точніший результат за допомогою формули  $x_1 = \varphi(x_0)$  або в загальнішому вигляді:

$$x_{n+1} = \varphi(x_n) \quad (n = 0, 1, 2, \dots). \quad (2.20)$$

Повторюючи цей процес, тобто інтегруючи декілька разів, можна отримати значення кореня із будь-яким ступенем точності, якщо виконується достатня умова:

$$|\varphi'(x)| < 1 \text{ на сегменті } [\xi; x_0], \quad (2.21)$$

де  $x = \xi$  – точне значення кореня.

Якщо умова (2.21) не виконується, тоді рівняння  $f(x) = 0$  завжди можна подати у вигляді  $x = x - c \cdot f(x)$ , і константу  $c$  підібрати таким чином, щоб для функції  $\varphi(x) = x - c \cdot f(x)$  умова (2.21) мала місце. Тоді згідно з формулою (2.20) отримаємо:

$$x_{n+1} = x_n - c f(x). \quad (2.22)$$

Метод ітерацій дає можливість «вгадувати» нові значення  $x_n$  під час здійснення будь-якого кроку  $n$ , що еквівалентно початку ітерацій за нового, більш успішного значення  $x_0$ . Відповідно, у тих випадках, коли процес збігається повільно, можна вносити відповідні корективи, враховуючи результати попередніх кроків.

Зазначені властивості методу ітерацій, його простота і необмежені можливості, закладені у формулі (2.22), дозволили ефективно використовувати його під час розв'язання диференціальних, інтегральних, інтегродиференціальних рівнянь.

Геометричний зміст розв'язку рівняння  $x = \varphi(x)$  методом простих ітерацій подано на рисунку 2.11.

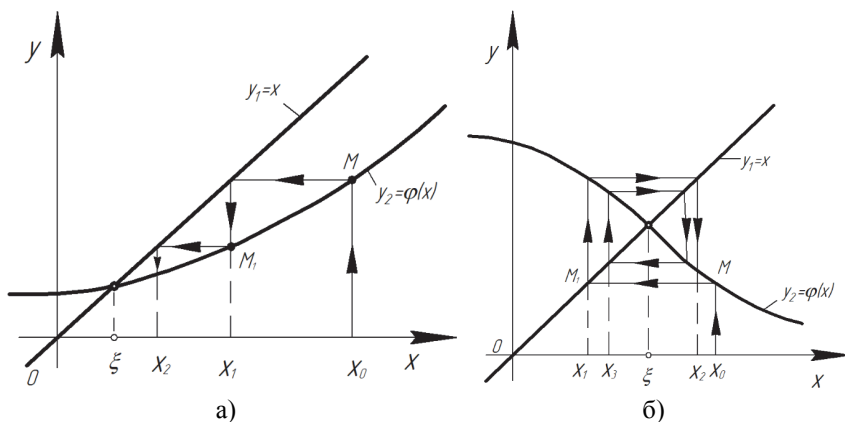


Рисунок 2.11 – Схема визначення коренів рівняння методом простих ітерацій:

а)  $0 < \varphi'(x) < 1$ ; б)  $-1 < \varphi'(x) < 0$



На рисунку 2.11, а) побудовано графіки функцій  $y = \varphi(x)$  і  $y = x$ . Коренем рівняння є абсциса точки перетину кривої  $y = \varphi(x)$  із бісектрисою координатного кута. Якщо  $x_0$  – нульового наближення, тоді  $x_1 = \varphi(x_0)$  дорівнює ординаті відповідної точки  $M$  кривої або абсцисі точки  $M_1$ . Аналогічно визначаються наступні наближення (див. рис. 2.11, а). Також можна встановити роль умови  $|\varphi'(x)| < 1$ .

На рисунку 2.11, а) зображено випадок, коли  $0 < \varphi'(x) < 1$ , так що крива перетинає бісектрису зліва направо і справа лежить під бісектрисою. Ітераційний процес у такому випадку збігається, причому наближення монотонно спадають, якщо  $x_0 > \bar{x}$ , або монотонно зростають за  $x_0 < \bar{x}$ . На рисунку 2.11, б) наведено випадок, коли похідна  $\varphi'(x)$  – від’ємна ( $-1 < \varphi'(x) < 0$ ). Якщо водночас й  $|\varphi'(x)| < 1$ , тоді ітераційний процес збігається, але наближення коливаються близько істинного значення кореня.

Для визначення втрати точності в методі ітерації необхідно проаналізувати вплив помилок округлення проміжних результатів на кінцеві результати.

Зокрема, під час розв’язання рівняння за методом ітерацій для двох послідовних наближень можна встановити такий зв’язок:

$$x_{n+1} - \xi \approx \varphi'(\xi)(x_n - \xi), \quad (2.23)$$

де  $\xi$  – точне значення шуканого кореня.

Таким чином, точність результату в процесі визначення кореня за методом ітерацій зростає, приблизно як геометрична прогресія із знаменником  $\varphi'(\xi)$ .

Алгоритм методу простих ітерацій наведено на рисунку 2.12.

**Приклад 2.6.** Визначити значення коренів рівняння:

$$4x - 5 \ln x = 5 \quad (2.24)$$

із точністю  $\varepsilon = 0,00001$ .

*Розв’язання:*

Записавши рівняння у формі  $\ln x = \frac{4x-5}{5}$ , визначається нульове наближення графічно (рис. 2.13), знаходячи перетин логарифмічної кривої  $y_2$  із прямою  $y_1 = \frac{4}{5}x - 1$ . Із рисунка 2.13 визначаються два наближених значення корені:  $x_0 = 2,28$  і  $x_0 = 0,57$ , які і будуть взяті за нульове наближення.

Для точнішого пошуку правого кореня рівняння (2.24) записується у вигляді  $x = 1,25(1 + \ln x)$ , де  $\varphi(x) = 1,25(1 + \ln x)$ . Ітераційний процес є збіжним, оскільки значення похідної функції  $\varphi'(x) = \frac{1,25}{x}$  в околі правого кореня є додатним і меншим одиниці за умовою (2.21). Обчислення наведено в таблиці 2.3.

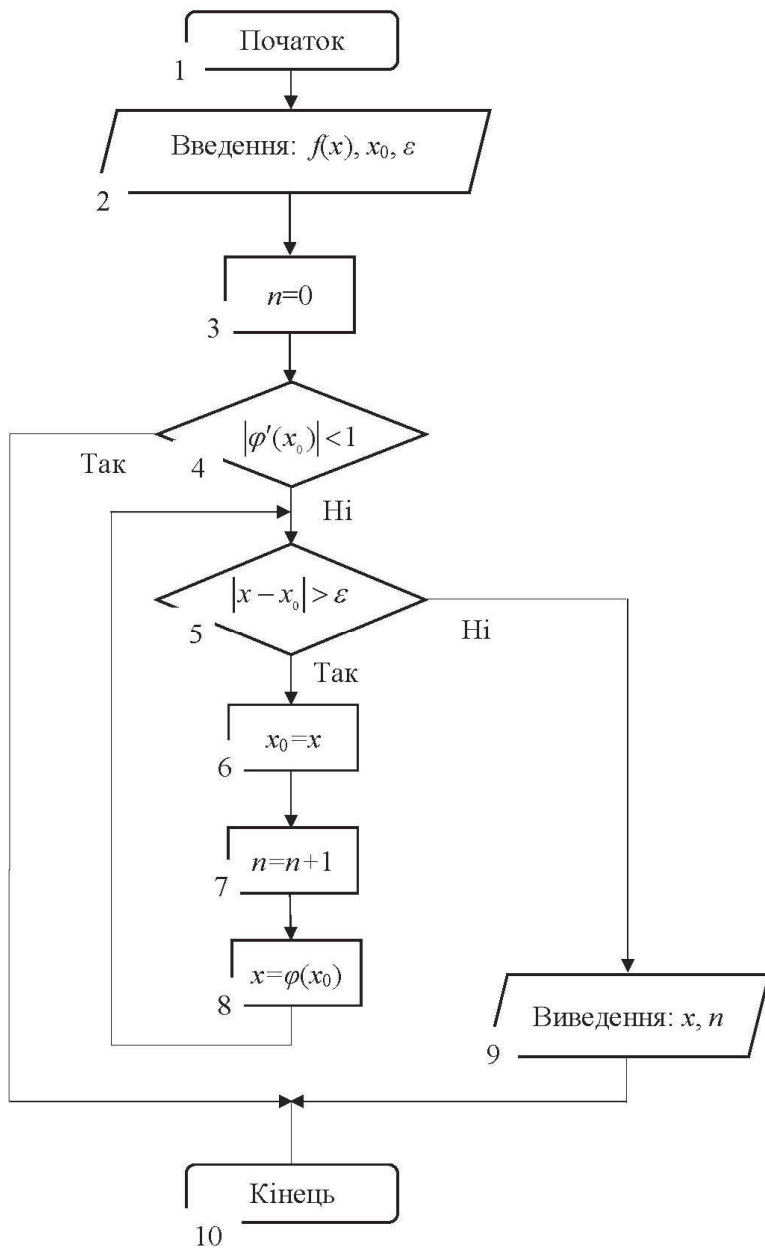


Рисунок 2.12 – Схема алгоритму методу простих ітерацій

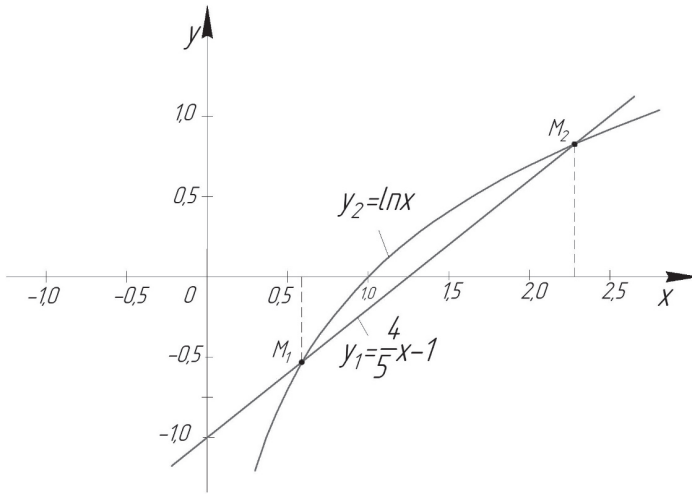


Рисунок 2.13 – Схема графічного визначення інтервалу належності коренів рівняння

Таблиця 2.3 – Результати розв’язання рівняння

$n$	$x_n$	$\ln(x_n)+1$	$1,25(\ln(x_n)+1)$
0	2,28000	1,82418	2,28022
1	2,28022	1,82427	2,28034
2	2,28034	1,82432	2,28040
3	2,28040	1,82435	2,28044
4	2,28044	1,82437	2,28046
5	2,28046	1,82438	2,28048
6	2,28048	1,82439	2,28049
7	2,28049	1,82439	2,28049

Для пошуку кореня рівняння із точністю до п’яти знаків було виконано вісім кроків. Висока швидкість збіжності пояснюється малим значенням похідної в околі кореня 0.55, а також успішним вибором нульового наближення.

В процесі пошуку лівого кореня ітераційний процес є розбіжним, тому що  $\varphi'(x) = 1,25/x$  в області  $x = 0,57$  має значення близько 2,2. Тому початкове рівняння (2.24) необхідно переписати у вигляді:

$$x = e^{0,8x-1}, \quad (2.25)$$

тоді  $\varphi(x) = e^{0,8x-1}$  і  $\varphi'(x) = 0,8e^{0,8x-1}$ .

За  $x_0 = 0,57$  значення функції  $\varphi'(x) \approx 0,46 < 1$  та ітераційний процес збіжний. Як показують обчислення (табл. 2.4) для пошуку кореня із точністю до п'яти знаків після коми необхідно вже одинадцять кроків. У такому разі процес збігається повільніше, ніж у попередньому випадку, хоча значення похідної менше. Причиною сповільнення є менш успішний, аніж раніше, вибір нульового наближення. Якщо у першому випадку нульове наближення відрізнялось від істинного кореня на величину порядку  $10^{-4}$ , то у другому випадку – на величину порядку  $10^{-2}$ .

Таблиця 2.4 – Результати розв'язання рівняння

$n$	$x_n$	$0,8x_n$	$0,8x_n - 1$	$e^{0,8x_n - 1}$
0	0,57000	0,45600	-0,55500	0,58042
1	0,58042	0,46434	-0,53566	0,58528
2	0,58528	0,46822	-0,53178	0,58756
3	0,58756	0,47005	-0,52995	0,58863
4	0,58863	0,47090	-0,52910	0,58913
5	0,58913	0,47130	-0,52870	0,58937
6	0,58937	0,47150	-0,52850	0,58949
7	0,58949	0,47159	-0,52841	0,58954
8	0,58954	0,47163	-0,52837	0,58957
9	0,58957	0,47166	-0,52834	0,58958
10	0,58958	0,47166	-0,52834	0,58958

Розглянемо реалізацію методу Ньютона (дотичних) мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
x = np.arange(-5, 4, 0.1)
from numpy import log as ln
#визначаємо функцію
func_x=(4*x)-(5*ln(x))-5
#будуємо графік і локалізуємо корені рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
plt.plot(x, func_x)
plt.legend(['f(x)=(4*x)-(5*ln(x))-5'], loc=1)
plt.grid(True)
plt.xlim([-2, 4])
plt.ylim([-2, 4])
plt.show()
#перша форма вираження функції через x
def f_first(x):
    return 1.25*(1+ln(x))
#друга форма вираження функції через x
def f_second(x):
    return math.exp((0.8*x)-1)
```

```

from sympy import *
#визначаємо похідні функцій через які буди виражені шуканий аргумент x
x_arg = Symbol('x_arg')
func_first=1.25*(1+ln(x_arg))
func_second=exp((0.8*x_arg)-1)
d_first=func_first.diff(x_arg)
d_second=func_second.diff(x_arg)
print("Похідна 1-ої функції f1'(x)=",d_first,". Похідна 2-ої функції
      f2'(x)=",d_second)
from timeit import default_timer as timer
##задаємо точку нульового наближення для визначення кореня рівняння
x[0]=2.2; x[1]=0.55; e=0.00001; k_1=0; k_2=0
func_first = lambdify(x_arg, d_first)
func_second = lambdify(x_arg, d_second)
for i in range(0,2):
    if abs(func_first(x[i]))<1:
        #визначаємо корінь рівняння
        start= timer()
        x_first=x[i]
        while abs(f_first(x_first)-x_first)>e:
            k_1=k_1+1
            x_first=f_first(x_first)
        end = timer()
        time_1=end-start
    if abs(func_second(x[i]))<1:
        x_second=x[i]
        #визначаємо корінь рівняння
        start= timer()
        while abs(f_second(x_second)-x_second)>e:
            k_2=k_2+1
            x_second=f_second(x_second)
        end = timer()
        time_2=end-start
print("x1 =", x_first, "Time taken for x1:",time_1, "Number of iterations
      for x1:",k_1)
print("x2 =", x_second, "Time taken for x2:",time_2, "Number of iterations
      for x2:",k_2).

```

### 2.3 Метод визначення комплексних коренів

Для визначення комплексних коренів (*complex roots*) можна застосовувати ті самі методи, що й для дійсних коренів, але оперують вже арифметикою комплексних чисел (контроль збіжності та похибки ведеться за модулем комплексного числа), що не завжди зручно для користувача.

Існує низка спеціальних методів, що дозволяють оцінювати комплексні корені, проводячи обчислення з дійсними числами. Одним із відомих підходів є метод Ліна, який базується на перетворенні початкового алгебраїчного рівняння  $P_n(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = 0$  на добуток квадратичних співмножників типу  $x^2 + px + q$ , а саме:

$$P_n(x) = (x^2 + px + q)Q_{n-2}(x) + R(x), \quad (2.26)$$

де  $Q_{n-2}(x) = b_{n-2}x^{n-2} + b_{n-3}x^{n-3} + \dots + b_{n-2-j}x^{n-2-j} + \dots + b_1x + b_0$  ( $j = 0, 1, 2, \dots, n-2$ ), а залишковий член  $R(x)$  дорівнює нулю.

Якщо поліном  $P_n(x)$  містить комплексні корені, то вони можуть бути виражені через коефіцієнти  $p$  і  $q$ , а саме

$$x_{1,2} = \alpha \pm i\beta \quad (\alpha = -0,5p; \quad \beta = \sqrt{q - 0,25p^2}).$$

Для відокремлення дійсних і комплексних коренів рівняння поліному типу

$$P_n(x) \equiv x^n + a_{n-1}x^{n-1} + \dots + a_1x + a_0 = 0, \quad (2.27)$$

задалегідь визначається наявність і кількість коренів за допомогою таких теорем.

**Теорема 2.3** (про число коренів алгебраїчного рівняння). Алгебраїчне рівняння (2.27)  $n$ -го ступеня має число  $n$  коренів (дійсне або комплексне) за умови, що кожен корінь враховується стільки разів, якою є його кратність.

**Теорема 2.4** (про властивість парної спряженості комплексних коренів рівняння (2.43)). Якщо  $x_{*i} = \alpha + i\beta$  – корінь алгебраїчного рівняння (2.27) кратності  $k$ , тоді число  $\overline{x_{*i}} = \alpha - i\beta$  також є коренем тієї самої кратності. Наслідком цієї теореми є те, що алгебраїчне рівняння непарного степеня має щонайменше один дійсний корінь.

**Теорема 2.5** (теорема Декарта про кількість дійсних коренів алгебраїчних рівнянь алгебри). Число  $S_1$  кількості додатних коренів (з урахуванням їх кратності) алгебраїчного рівняння  $P_n(x)=0$  дорівнює числу змін знаків у послідовності коефіцієнтів  $a_n, a_{n-1}, \dots, a_0$  (коефіцієнти, які дорівнюють нулю, не враховуються) многочлена  $P_n(x)=0$  або менше цього числа на парне число. Число  $S_2$  кількості від'ємних коренів (з урахуванням їх кратності) алгебраїчного рівняння  $P_n(x)=0$  дорівнює числу змін знаків у послідовності коефіцієнтів  $a_n, a_{n-1}, \dots, a_0$  (коефіцієнти, які дорівнюють нулю, не враховуються) многочлена  $P_n(-x)=0$  або менше цього числа на парне число.

**Теорема 2.6** (теорема Гюа про необхідну умову дійсності всіх коренів алгебраїчного рівняння). Якщо алгебраїчне рівняння (2.27) має усі дійсні корені, тоді квадрат кожного некрайнього коефіцієнта більше добутку двох його сусідніх коефіцієнтів. Наслідком цієї теореми є те, якщо за будь-якого  $k$  виконується нерівність  $a_k^2 \leq a_{k-1}a_{k+1}$ , тоді рівняння (2.27) має хоча б одну пару комплексних коренів.

Розкриваючи дужки в рівнянні (2.26) і прирівнюючи коефіцієнти при однакових степенях  $x$ , отримуємо:

$$\begin{cases} b_{n-2} = a_n, \\ b_{n-3} = a_{n-1} - pb_{n-2}, \\ b_{j-2} = a_j - pb_{j-1} - qb_j, \\ qb_1 + b_0p = a_1, \\ qb_0 = a_0, \\ j = n-2, n-3, \dots, 2. \end{cases} \quad (2.28)$$

Розв'язуючи систему лінійних рівнянь (2.28), визначають значення квадратного рівняння  $p$  і  $q$ , а також коефіцієнти полінома  $b$ , що має тепер порядок на два нижче за вихідний ( $b_1 = b_0 = 0$ ). Пошук коренів квадратного рівняння  $x^2 + px + q = 0$  проводиться також за класичною схемою.

У методі Ліна для пошуку розв'язку системи рівнянь (2.28) використовується метод прогонки або метод простих ітерацій.

Задаючись початковими значеннями  $p^0, q^0$ , послідовно визначаються значення  $b_{n-1}^0, b_{n-2}^0, \dots, b_2^0$ , а також значення  $p^1, q^1$ . Вибираючи значення  $p$  і  $q$  як наступні наближення, повторюють процедуру пошуку  $b, p, q$  до самої збіжності. На рисунку 2.14 наведено схему алгоритму пошуку комплексних коренів алгебраїчного рівняння  $n$ -ого порядку методом Ліна.

Також потрібно відзначити, що метод простих ітерацій є умовно-збіжним, проте практично прийнятних критеріїв збіжності цього методу немає. Під час практичної реалізації методу Ліна необхідно стежити за кількістю ітерацій. Якщо вона виходить за розумну межу, потрібно вважати, що метод є розбіжним для обраного алгебраїчного рівняння.

**Приклад 2.7.** Визначити значення комплексних коренів алгебраїчного рівняння:

$$x^4 - 2x^3 + 18x^2 + 3x - 5 = 0, \quad (2.29)$$

із точністю  $\varepsilon = 0,0001$ .

*Розв'язання:*

Для початку необхідно визначити наявність дійсних і комплексних коренів алгебраїчного рівняння (2.29). Згідно з теоремою 2.3 алгебраїчне рівняння (2.29) має чотири корені рівняння, оскільки степінь поліному  $n=4$ . Зважаючи, що  $a_k^2 \leq a_{k-1}a_{k+1}$ , де  $k=2, a_1=1, a_2=-2$  і  $a_3=18$ , згідно з теоремою 2.6 алгебраїчне рівняння (2.29) має одну пару комплексних коренів.

Алгебраїчне рівняння (2.29) запишемо у формі (2.26):

$$x^4 - 2x^3 + 18x^2 + 3x - 5 = (x^2 + px + q)(x^2 + b_1x^2 + b_0) = 0. \quad (2.30)$$

Розкриваючи дужки в рівнянні (2.30) і прирівнюючи коефіцієнти при однакових степенях  $x$ , отримаємо:

$$\begin{cases} b_1 = a_3 - p; \\ b_0 = a_2 - pb_1 - q; \\ qb_1 + b_0p = a_1; \\ qb_0 = a_0, \end{cases} \quad (2.31)$$

де  $a_0 = -5, a_1 = 3, a_2 = 18, a_3 = -2$ .

Для початкових значень наближення  $p = 1$  і  $q = 1$  виконується ітераційне розв'язання системи рівнянь (2.31). Результати розв'язання системи рівнянь (2.31) по ітераціях подано в таблиці 2.5.

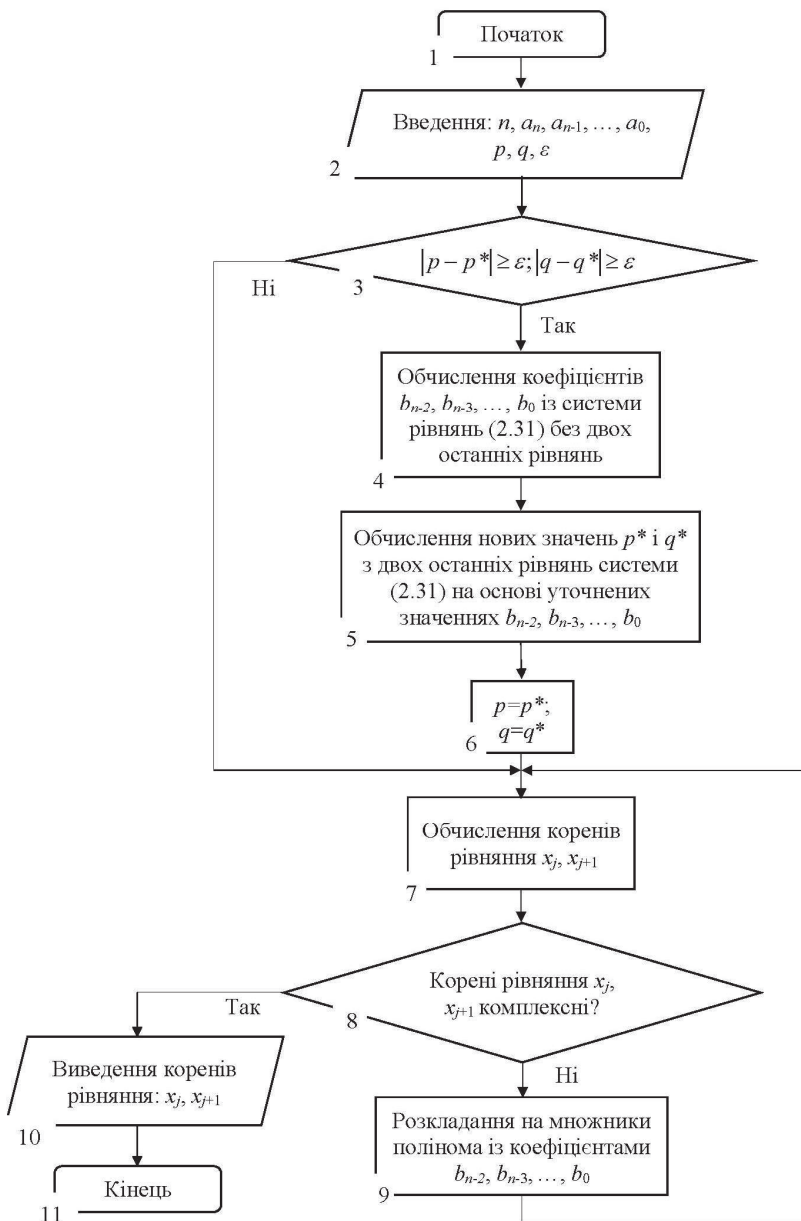


Рисунок 2.14 – Схема алгоритму метода Ліна



Таблиця 2.5 – Результати ітераційного розв’язання системи рівнянь

$n$	$p$	$q$	$b_0$	$b_1$	$b_2$
0	1,00000	1,00000	24,0000	-4,00000	1,0
1	0,45833	-0,20833	19,33507	-2,45833	1,0
2	0,12867	-0,25860	18,52349	-2,12867	1,0
3	0,13217	-0,26980	18,55161	-2,13070	1,0
4	0,13070	-0,26957	18,54801	-2,13070	1,0
5	0,13078	-0,26957	18,54800	-2,13070	1,0
6	0,13077	-0,26956	18,54824	-2,13078	1,0

На основі результатів ітераційного розв’язку системи рівнянь (2.31) було отримано розклад поліному (2.29) на такі квадратичні множники:

$$\begin{aligned} x^4 - 2x^3 + 18x^2 + 3x - 5 &= (x^2 + px + q)(x^2 + b_1x^2 + b_0) = \\ &= (x^2 + 0,13077x - 0,26956)(x^2 - 2,13078x^2 + 18,54824) = 0. \end{aligned} \quad (2.32)$$

Результатом розв’язання квадратичних множників (2.32) є дійсні і комплексні корені рівняння (2.29):

$$x_1 = 0,45791; \quad x_2 = -0,58868; \quad x_3 = 1,06539 \pm i\sqrt{4,17291}.$$

Розглянемо реалізацію методу Ліна на мові програмування PYTHON:

```
n=4 #значення степеню полінома
a=[-5, 3, 18, -2, 1] #значення коефіцієнтів полінома
#загальний вигляд поліному
print('Необхідно визначити комплексні корені алгебраїчного рівняння
вигляду:')
print('{0}*x^{1}'.format(a[n],n),end='')
for j in range(n-1,0,-1):
    print('+{0}*x^{1}'.format(a[j],j),end='')
print('+{0}=0'.format(a[0]))
#визначаємо умову наявності комплексних коренів рівняння
for k in range(1,n):
    if (a[k]^2)<=a[k-1]*a[k+1]:
        print('Алгебраїчне рівняння має мінімум одну пару комплексних
коренів')
        break
p=1; q=1 #початкові значення наближення для чисел p і q
e=0.00001 #значення степені точності обчислення комплексних коренів рівняння
b=[0]*(n+1)
p_new=p+1; q_new=q+1
reg=0
while (abs(p_new-p)>=e) and (abs(q_new-q)>=e):
    p=p_new; q=q_new
    for i in range(n-2,-1,-1):
        b[i]=a[i+2]-(p*b[i+1])-(q*b[i+2])
    reg=reg+1
    print('Значення коефіцієнтів поліному співмножника (x^2+px+q) на', reg,
'ітерації-', b)
    q_new=a[0]/b[0]
    p_new=(a[1]/b[0])-(b[1]/b[0])*q
```

```

    print('Значення коефіцієнтів множника p_new=', p_new, 'q_new=', q_new)
    print('-----')
#перевірка на наявність комплексних коренів рівняння для квадратичного
# множника поліному (x^2+px+q)
import math
discr=(p_new*p_new)-(4*q_new)
if discr<0:
    alfa=-0.5*p_new
    bet=math.sqrt(abs(discr))
    print('Значення комплексних коренів рівняння:')
    print('x1={0}+i*{1}'.format(alfa,bet))
    print('x2=({0})-i*({1})'.format(alfa,bet))
else:
    print('Значення дійсних коренів рівняння:')
    print('x1=',(-0.5*p_new)+(0.5*math.sqrt(discr)))
    print('x2=',(-0.5*p_new)-(0.5*math.sqrt(discr)))
#визначення комплексних коренів рівняння із залишкового співмножника
    поліному (x^2+px+q)
c=[]
for cell in b:
    if cell!=0: c.append(cell)
print (c)
discr_2=(c[1]*c[1])-(4*c[0]*c[2])
if discr_2<0:
    alfa=(-0.5*c[1])/c[2]
    bet=0.5*math.sqrt(abs(discr_2))/c[2]
    print('Значення комплексних коренів рівняння:')
    print('x3={0}+i*{1}'.format(alfa,bet))
    print('x4=({0})-i*({1})'.format(alfa,bet))
else:
    print('Значення дійсних коренів рівняння:')
    print('x3=',(alfa)+(0.5*math.sqrt(discr_2)/c[2]))
    print('x4=',(alfa)-(0.5*math.sqrt(discr_2)/c[2]))

```

## 2.4 Чисельні методи розв'язання систем нелінійних алгебраїчних рівнянь

У загальному випадку система із  $n$  нелінійних рівнянь з  $n$  невідомими подається у вигляді:

$$\begin{cases} f_1(x_1, x_2, \dots, x_n) = 0; \\ f_2(x_1, x_2, \dots, x_n) = 0; \\ \vdots \\ f_n(x_1, x_2, \dots, x_n) = 0. \end{cases} \quad (2.33)$$

Оскільки нелінійні функції, що входять до системи (2.33), неможливо описати якоюсь певною загальною формою, то не може бути запропоновано будь-якого аналітичного прямого методу для розв'язання такої системи. З наближених ітераційних методів найбільш простим є метод простої ітерації, що базується на приведенні системи (2.33) до системи нелінійних рівнянь у вигляді:



$$\left\{ \begin{array}{l} f_1(x_1 + \Delta x_1, \dots, x_n + \Delta x_n) = f_1(x_1, \dots, x_n) + \Delta x_1 \frac{\partial f_1}{\partial x_1} + \dots + \Delta x_n \frac{\partial f_1}{\partial x_n} + R_n; \\ \dots; \\ f_n(x_1 + \Delta x_1, \dots, x_n + \Delta x_n) = f_n(x_1, \dots, x_n) + \Delta x_1 \frac{\partial f_n}{\partial x_1} + \dots + \Delta x_n \frac{\partial f_n}{\partial x_n} + R_n, \end{array} \right.$$

де  $R_n$  – члени другого та більших порядків, які в подальшому відкидаються.

Задача зводиться до розв’язання системи лінійних рівнянь:

$$\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \dots & \dots & \dots \\ \frac{\partial f_n}{\partial x_1} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \end{bmatrix} = \begin{bmatrix} -f_1 \\ -f_2 \\ \vdots \\ -f_n \end{bmatrix}. \quad (2.38)$$

У цій системі матрицю частинних похідних називають матрицею Якобі і позначають  $\mathbf{W}(\mathbf{X})$ .

Знайдені для певного  $(m+1)$  кроку ітерації значення  $\Delta x_i$  використовуються як поправки до попередніх наближень:

$$\left\{ \begin{array}{l} x_1^{(m+1)} = x_1^{(m)} + \Delta x_1; \\ \dots; \\ x_n^{(m+1)} = x_n^{(m)} + \Delta x_n. \end{array} \right. \quad (2.39)$$

Загальна ітераційна формула в матричному поданні має вигляд:

$$\mathbf{X}^{(m+1)} = \mathbf{X}^{(m)} - \mathbf{W}^{-1}[\mathbf{X}^{(m)}] \mathbf{F}[\mathbf{X}^{(m)}], \quad (2.40)$$

де  $\mathbf{F}[\mathbf{X}^{(m)}]$  – вектор-стовпець значень функцій  $f_1, f_2, \dots, f_n$  для наближень  $\mathbf{X}^{(m)}$ ,  $\mathbf{W}^{-1}[\mathbf{X}^{(m)}]$  – обернена матриця Якобі.

Певні труднощі в процесі реалізації алгоритму методу Ньютона виникають під час обернення матриці Якобі. Для цього використовуються відомі з лінійної алгебри способи обернення матриць.

Також існує багато варіантів застосування методу Ньютона. Наприклад, модифікований метод Ньютона:

$$\mathbf{X}^{(m+1)} = \mathbf{X}^{(m)} - \mathbf{W}^{-1}[\mathbf{X}^{(0)}] \mathbf{F}[\mathbf{X}^{(m)}]. \quad (2.41)$$

У цьому методі не потрібно обчислювати обернену матрицю Якобі на кожному кроці розрахунків, що спрощує алгоритм, але уповільнює збіжність і робить метод більш чутливим до вибору початкового наближення.

Також існує Метод Ньютона з параметром  $\tau$ :

$$\mathbf{X}^{(m+1)} = \mathbf{X}^{(m)} - \tau \mathbf{W}^{-1} \left[ \mathbf{X}^{(m)} \right] \mathbf{F} \left[ \mathbf{X}^{(m)} \right]. \quad (2.42)$$

Цей метод дещо схожий із методом послідовної верхньої релаксації для систем лінійних рівнянь. Застосовуються також різноманітні гібридні методи, в яких поєднується метод Ньютона з методом простої ітерації.

Збіжність методу Ньютона оцінюється шляхом обчислення показника:

$$q = \frac{M^2 LP}{2} < 1, \quad (2.43)$$

де  $M \geq \|\mathbf{W}^{-1}(\mathbf{X})\|$ ,  $L \geq \|\mathbf{W}(\mathbf{X})\|$ ,  $P \geq \|\mathbf{F}(\mathbf{X})\|$ , причому  $\lim_{l \rightarrow \infty} MP \sum_{m=0}^l q^{2^{m-1}} \rightarrow 0$ .

Похибка на  $m$ -ій ітерації визначається нерівністю:

$$\Delta \leq MP \frac{q^{2^m - 1}}{1 - q^{2^m}}. \quad (2.44)$$

Алгоритм методу наведено на рисунку 2.15.

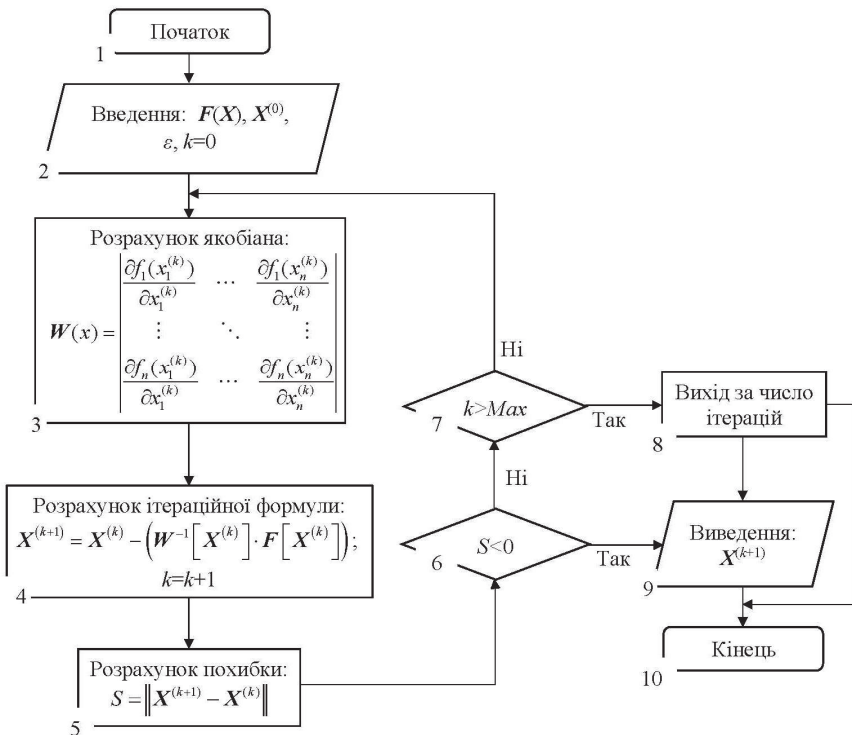


Рисунок 2.15 – Схема алгоритму методу Ньютона розв'язання системи нелінійних рівнянь

**Приклад 2.8.** Визначить значення коренів системи нелінійних рівнянь:

$$\begin{cases} x^2 + y^2 + z^2 = 1; \\ 2x^2 + y^2 - 4z^2 = 0; \\ 3x^2 - 4y + z^2 = 0. \end{cases} \quad (2.45)$$

із точністю  $\varepsilon = 0,001$ .

*Розв'язання:*

Вводимо позначення:

$$\mathbf{X} = \begin{vmatrix} x \\ y \\ z \end{vmatrix}; \quad \mathbf{F}(\mathbf{X}) = \begin{vmatrix} f_1 \\ f_2 \\ f_3 \end{vmatrix} = \begin{vmatrix} x^2 + y^2 + z^2 - 1 \\ 2x^2 + y^2 - 4z^2 \\ 3x^2 - 4y + z^2 \end{vmatrix};$$

$$\mathbf{W}(\mathbf{X}) = \begin{vmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial z} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial z} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial z} \end{vmatrix} = \begin{vmatrix} 2x & 2y & 2z \\ 4x & 2y & -4z \\ 6x & -4 & 2z \end{vmatrix}.$$

Обирається початкове наближення:

$$\mathbf{X}^0 = \begin{vmatrix} x_0 \\ y_0 \\ z_0 \end{vmatrix} = \begin{vmatrix} 0,5 \\ 0,5 \\ 0,5 \end{vmatrix}; \quad \mathbf{F}(\mathbf{X}^0) = \begin{vmatrix} f_1^{(0)} \\ f_2^{(0)} \\ f_3^{(0)} \end{vmatrix} = \begin{vmatrix} x_0^2 + y_0^2 + z_0^2 - 1 \\ 2x_0^2 + y_0^2 - 4z_0^2 \\ 3x_0^2 - 4y_0 + z_0^2 \end{vmatrix} = \begin{vmatrix} -0,25 \\ -0,25 \\ -1,00 \end{vmatrix};$$

$$\mathbf{W}(\mathbf{X}_0) = \begin{vmatrix} 2x_0 & 2y_0 & 2z_0 \\ 4x_0 & 2y_0 & -4z_0 \\ 6x_0 & -4 & 2z_0 \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 \\ 2 & 1 & -4 \\ 3 & -4 & 1 \end{vmatrix}.$$

Визначається значення оберненої матриці якобіана функції  $\mathbf{F}(\mathbf{X})$ :

$$\mathbf{W}^{-1}(\mathbf{X}_0) = \begin{vmatrix} 0,375 & 0,125 & 0,125 \\ 0,350 & 0,050 & -0,150 \\ 0,275 & -0,175 & 0,025 \end{vmatrix}.$$

Використовуючи модифікований метод Ньютона, на основі ітераційної формули (2.41) виконуємо ітераційні обчислення:

$$\begin{aligned} \mathbf{X}^{(1)} &= \mathbf{X}^{(0)} - \mathbf{W}^{-1} [\mathbf{X}^{(0)}] \mathbf{F} [\mathbf{X}^{(0)}] = \\ &= \begin{pmatrix} 0,5 \\ 0,5 \\ 0,5 \end{pmatrix} - \begin{pmatrix} 0,375 & 0,125 & 0,125 \\ 0,350 & 0,050 & -0,150 \\ 0,275 & -0,175 & 0,025 \end{pmatrix} \begin{pmatrix} -0,25 \\ -0,25 \\ -1,00 \end{pmatrix} = \begin{pmatrix} 0,750 \\ 0,450 \\ 0,550 \end{pmatrix}, \end{aligned}$$

$$\mathbf{F}(\mathbf{X}^{(1)}) = \begin{pmatrix} f_1^{(1)} \\ f_2^{(1)} \\ f_3^{(1)} \end{pmatrix} = \begin{pmatrix} 0,0675 \\ 0,1175 \\ 0,1900 \end{pmatrix};$$

$$\begin{aligned} \mathbf{X}^{(2)} &= \mathbf{X}^{(1)} - \mathbf{W}^{-1} [\mathbf{X}^{(1)}] \mathbf{F} [\mathbf{X}^{(1)}] = \\ &= \begin{pmatrix} 0,750 \\ 0,450 \\ 0,550 \end{pmatrix} - \begin{pmatrix} 0,375 & 0,125 & 0,125 \\ 0,350 & 0,050 & -0,150 \\ 0,275 & -0,175 & 0,025 \end{pmatrix} \begin{pmatrix} 0,0675 \\ 0,1175 \\ 0,1900 \end{pmatrix} = \begin{pmatrix} 0,68625 \\ 0,44900 \\ 0,54725 \end{pmatrix}, \end{aligned}$$

$$\mathbf{F}(\mathbf{X}^{(2)}) = \begin{pmatrix} f_1^{(2)} \\ f_2^{(2)} \\ f_3^{(2)} \end{pmatrix} = \begin{pmatrix} -0,0279 \\ -0,0545 \\ -0,0837 \end{pmatrix}; \dots; \mathbf{X}^{(7)} = \begin{pmatrix} 0,70641 \\ 0,44896 \\ 0,54748 \end{pmatrix}; \mathbf{F}(\mathbf{X}^{(7)}) = \begin{pmatrix} f_1^{(3)} \\ f_2^{(3)} \\ f_3^{(3)} \end{pmatrix} = \begin{pmatrix} -0,000786 \\ -0,001571 \\ -0,002357 \end{pmatrix}.$$

На цьому кроці обчислень було отримано похибку вимірювань:

$$E_x = \|\mathbf{X}^{(7)} - \mathbf{X}^{(6)}\| = \left\| \begin{pmatrix} 0,70641 \\ 0,44896 \\ 0,54748 \end{pmatrix} - \begin{pmatrix} 0,70563 \\ 0,44896 \\ 0,54748 \end{pmatrix} \right\| = \max(|0,000786; 0; 0|) = 0,000786 \leq \varepsilon.$$

Розглянемо реалізацію методу Ньютона для розв'язання системи нелінійних рівнянь мовою програмування PYTHON:

```
import numpy as np
from scipy import *
from sympy import *
# визначаємо вектор-стовпець значень функцій системи нелінійних рівнянь
def F_X(x_num, y_num, z_num):
    F_X=np.array([(x_num**2)+(y_num**2)+(z_num**2)-1,
                  (2*(x_num**2))+(y_num**2)-(4*(z_num**2)),
                  (3*(x_num**2))-(4*y_num)+(z_num**2)])
    return F_X
def Yak_Xrev(x_num, y_num, z_num):
    x, y, z=symbols('x y z')
    f_1=(x**2)+(y**2)+(z**2)-1
    f_2=(2*(x**2))+(y**2)-(4*(z**2))
    f_3=(3*(x**2))-(4*y)+(z**2)
    g_1=[diff(f_1, var) for var in [x, y, z]]
    g_2=[diff(f_2, var) for var in [x, y, z]]
```

```

g_3=[diff(f_3,var) for var in [x,y,z]]
g_1num = lambdify([x,y,z], g_1)
g_2num = lambdify([x,y,z], g_2)
g_3num = lambdify([x,y,z], g_3)
Yak = np.array([g_1num(x_num,y_num,z_num),
                g_2num(x_num,y_num,z_num),g_3num(x_num,y_num,z_num)])
return np.linalg.inv(Yak)
# розв'язання ітераційного рівняння
X=np.array([0.4,0.4,0.4])
X_1=np.array([0.5,0.5,0.5])
e=0.001
k=0
while np.linalg.norm(X-X_1,np.inf)>=e:
    X=X_1
    X_1=X-(Yak_Xrev(0.5,0.5,0.5).dot(F_X(X[0],X[1],X[2])))
    k=k+1
    print('k=',k)
    print('X',X)
    print('F_X(X[0],X[1],X[2])=',F_X(X[0],X[1],X[2]))
    print('X_1=',X_1)
    print('Поточне значення похибки',np.linalg.norm(X-X_1,np.inf))
    print('-----').

```

### **Висновки щодо застосування методів розв'язання трансцендентних рівнянь, систем нелінійних рівнянь, а також пошуку комплексних коренів рівнянь поліномів**

Для розв'язання кінцевих нелінійних рівнянь доцільно використовувати методи: половинного ділення (дихотомії), Ньютона (дотичних), хорд (лінійної інтерполяції), а також простих ітерацій. Перед застосуванням цих методів виконується відокремлення коренів за допомогою таких аналітичних методів як: пошук більш простішого рівняння, яке має корені, що приблизно дорівнюють невідомим кореням цього рівняння (нехтування малими членами рівняння) і використанням теорем на основі відомих властивостей неперервних функцій. Необхідно відзначити, що метод ділення відрізка навпіл не висуває жодних вимог до гладкості функції, тобто достатньо, щоб функція була неперервною. Це робить його надто популярним у простих задачах із пошуком одного кореня на заданому інтервалі (наприклад, у сфері метрології). Під час обчислення коренів рівняння методом Ньютона процес послідовних наближень завжди збігається, якщо нульове наближення прийнято дуже близьким до кореня на певному сегменті. Тому він дуже актуальний у задачах з уточненням попередньо «грубо» визначеного кореня. Обчислення кореня рівняння методом хорд із будь-яким заданим ступенем точності, будуть такими самими, що і методом Ньютона. Але метод Ньютона у більшості випадків дає кращу збіжність, ніж метод хорд. У тих випадках, коли обчислення функції та її похідної трудомістке, метод хорд дає більшу економію праці. Нарівні із вищевказаними методами метод ітерацій дає можливість «вгадувати» нові значення в процесі здійснення будь-якого кроку. Зазначені властивості методу ітерацій, його простота і необмежені можливості, виражені у формулі, дозволяють ефективно використовувати для розв'язання диференціальних, інтегральних, інтегро-диференціальних



рівнянь. Такі задачі часто виникають у процесі розрахунків балістики, космонавтики або визначення кредитних умов у фінансовому секторі.

Задачі із пошуком комплексних коренів рівнянь поліномів актуальні у галузі автоматичного керування складними об'єктами. Під час визначення комплексних коренів контроль збіжності та похибки ведеться за модулем комплексного числа. У таких типах задач ефективним є використання методу Ліна, що дозволяє оцінювати комплексні корені шляхом обчислень з дійсними числами. Перед застосуванням цього методу також проводиться відокремлення дійсних і комплексних коренів рівняння полінома за допомогою теорем Гюа і Декарта. У методі Ліна для пошуку розв'язку системи ітераційних рівнянь використовується метод прогонки або метод простих ітерацій. Потрібно відзначити, що метод простих ітерацій є умовно-збіжним, проте практично прийнятних критеріїв збіжності цього методу немає. В процесі практичної реалізації методу Ліна необхідно стежити за кількістю ітерацій, якщо вона виходить за розумну межу, потрібно вважати, що метод є розбіжним для обраного алгебраїчного поліноміального рівняння.

Також поширеним є розв'язання задач динаміки складних систем і фізики твердого тіла на основі математичної моделі у вигляді системи нелінійних рівнянь. Для розв'язання систем нелінійних рівнянь найефективнішим є стійкий метод – метод Ньютона, який базується на розкладанні усіх складових рівнянь у ряд Тейлора. Загалом задача розв'язання системи нелінійних рівнянь зводиться до розв'язання системи лінійних рівнянь, в основі якого покладено використання оберненої матриці Якобі. З метою спрощення загального алгоритму може бути використаний модифікований метод Ньютона, що дозволяє не виконувати обчислення значення оберненої матриці Якобі на кожному кроці. Недоліком такого підходу є уповільнення збіжності і підвищення чутливості до вибору початкового наближення. Також має місце застосування різноманітних гібридних методів, в яких поєднується метод Ньютона із методом простої ітерації.

### **Контрольні запитання та завдання**

1. Яке число називається коренем функції?
2. Які відомі способи існують для відокремлення коренів функції?
3. Якою умовою є існування коренів рівняння в певній області визначення функції?
4. Розкрити суть чисельних методів половинного ділення (дихотомії), Ньютона (дотичних), хорд (лінійної інтерполяції), а також простих ітерацій для розв'язання нелінійних алгебраїчних рівнянь?

5. Як перевірити збіжність ітераційних алгоритмів розв'язання нелінійних алгебраїчних рівнянь?

6. Чим відрізняються алгоритми половинного ділення (дихотомії), Ньютона (дотичних), хорд (лінійної інтерполяції), а також простих ітерацій?

7. За допомогою методу половинного ділення визначіть час польоту снаряду із точністю  $\varepsilon=0,00001$  за умови, що сила опору повітря руху снаряда пропорційна його швидкості ( $R=-kmv$ ). Рівняння руху снаряду подано функцією у параметричній формі (параметр  $t$  – час польоту снаряда) в проекції на вертикальну вісь ординат  $y$ :

$$y = \frac{1}{k^2} (g + kv_0 \sin \alpha) (1 - e^{-kt}) - \frac{g}{k} t,$$

де  $g=9,82 \text{ м/с}^2$  – прискорення вільного падіння. Вихідні дані по варіантах подано у таблиці 2.6.

Таблиця 2.6 – Вихідні дані до завдання

Варіант	Коефіцієнт опору повітря ( $k, \text{с}^{-1}$ )	Початкова швидкість ( $v_0, \text{м/с}$ )	Кут між вектором початкової швидкості і поверхнею горизонту ( $\alpha, \text{град}$ )
1	0,25	1200	30
2	0,33	1300	45
3	0,15	1450	50
4	0,42	1560	60

8. За допомогою методу Ньютона (дихотомії) визначіть дальність польоту снаряду із точністю  $\varepsilon=0,00001$  за умови, що сила опору повітря руху снаряда пропорційна його швидкості ( $R=-kmv$ ). Рівняння руху снаряду подано функцією у проекціях на горизонтальну й вертикальну осі  $x, y$ :

$$y = \frac{1}{k} \left( \frac{g + kv_0 \sin \alpha}{v_0 \cos \alpha} \right) x + \frac{g}{k^2} \ln \left( 1 - \frac{k}{v_0 \cos \alpha} x \right),$$

де  $g=9,82 \text{ м/с}^2$  – прискорення вільного падіння. Вихідні дані по варіантах подано у таблиці 2.7.

Таблиця 2.7 – Вихідні дані до завдання

Варіант	Коефіцієнт опору повітря ( $k, \text{с}^{-1}$ )	Початкова швидкість ( $v_0, \text{м/с}$ )	Кут між вектором початкової швидкості і поверхнею горизонту ( $\alpha, \text{град}$ )
5	0,25	1200	30
6	0,33	1300	45
7	0,15	1450	50
8	0,42	1560	60

9. За допомогою методу хорд (лінійної інтерполяції) визначить процентну ставку  $i$  дохідності облигацій із точністю  $\varepsilon=0,0001$ . Функція дохідності подана таким трансцендентним рівнянням:

$$(I + P) \cdot (1 + i)^{-n} + P \cdot (1 + i)^{-n+1} + A_n \cdot (1 + i) - (A_n + I) = 0,$$

де  $n = T - N$  – термін, що залишився до погашення облигації (в роках);  $I = P \cdot k$  – величина купонних платежів (ум. грош. од.). Вихідні дані по варіантах подано у таблиці 2.8.

Таблиця 2.8 – Вихідні дані до завдання

Варіант	Поточна вартість облигації ( $A_n$ , ум. грош. од.)	Номинальна вартість ( $P$ , ум. грош. од.)	Термін після випуску облигації ( $N$ , к-сть років)	Термін погашення ( $T$ , к-сть років)	Купонна процентна ставка ( $k$ , долі від од.)
9	1150	1000	1	10	0,12
10	1300	1200	2	15	0,14
11	1420	1300	3	16	0,16
12	1500	1500	4	20	0,18

10. За допомогою методу Ньютона (дотичних) визначить процентну ставку  $i$  із точністю  $\varepsilon=0,0001$ , на основі формули складних процентів:

$$S = \frac{Q}{(i/12)} \left[ \left( 1 + \frac{i}{12} \right)^N - 1 \right].$$

Вихідні дані по варіантах подано у таблиці 2.9.

Таблиця 2.9 – Вихідні дані до завдання

Варіант	Сума забезпечення грошового вкладу ( $S$ , ум. грош. од.)	Термін протягом якого, будуть вноситись рівномірні платежі ( $N$ , к-сть місяців)	Сума рівномірного щомісячного внеску ( $Q$ , ум. грош. од.)
13	$1,0 \cdot 10^6$	9	$1,0 \cdot 10^5$
14	$2,0 \cdot 10^6$	10	$1,5 \cdot 10^5$
15	$2,5 \cdot 10^6$	11	$2,0 \cdot 10^5$
16	$3,0 \cdot 10^6$	12	$3,2 \cdot 10^5$

11. У рівнянні Ландау-Гінзбурга-Девоншира за допомогою методу простих ітерацій визначить із точністю  $\varepsilon=0,0001$  значення поляризації від дії зовнішнього поля кристалів з фазовими переходами першого роду:

$$\alpha P + \beta P^3 - \gamma P^5 + E_m = 0.$$

Вихідні дані по варіантах подано у таблиці 2.10.

Таблиця 2.10 – Вихідні дані до завдання

Варіант	Нормовані термодинамічні параметри			Напруженість електромагнітного поля ( $E_m$ , В/м)
	$\alpha$	$\beta$	$\gamma$	
17	1,0	1,00	0,01	0,0
18	1,0	0,01	0,00	1,0
19	1,0	1,50	0,10	1,2
20	1,2	1,10	0,08	1,4

12. За допомогою методу хорд (лінійної інтерполяції) із точністю  $\varepsilon = 0,00001$  визначить критичну силу (втрата вертикальної рівноваги стержня), прикладену вздовж стержня, один кінець якого жорстко закріплений, а інший шарнірно закріплений і може переміщуватися тільки у вертикальному напрямку, яка визначається із рівняння:

$$\operatorname{tg}\left(\sqrt{\frac{PL}{EI}}\right) - \sqrt{\frac{PL}{EI}} = 0.$$

Вихідні дані по варіантах подано у таблиці 2.11.

Таблиця 2.11 – Вихідні дані до завдання

Варіант	Згинальна жорсткість стержня ( $EI$ , Н·м <sup>2</sup> )	Довжина стержня ( $L$ , м)
21	450	3
22	500	5
23	560	7
24	630	9

13. На прикладі відомої задачі космонавтики і методу простих ітерацій визначить ефективне співвідношення елементів мас ракети, щоб було досягнуто максимальне значення ККД ракети із рівняння:

$$\frac{2z}{1+z} - \ln(1+z) = 0,$$

де  $z$  – співвідношення маси палива для реактивного двигуна і маси корисного вантажу.

14. Чим відрізняються дійсні й комплексні корені рівняння?

15. Сформулюйте теореми, за допомогою яких визначається наявність комплексних коренів рівняння?

16. Розкрийте суть методу Ліна визначення комплексних коренів рівняння полінома.

17. Охарактеризуйте критерії збіжності, що використовуються за застосування методу Ліна під час розбиття полінома на множники?

18. Обчисліть комплексні корені рівняння полінома із точністю  $\varepsilon = 0,0001$ :

19. Обчисліть дійсні корені рівняння полінома із точністю  $\varepsilon = 0,0001$ :

$$x^5 + 5x^4 - 3x^3 + 16x^2 + 5x + 9 = 0.$$

20. Обчисліть дійсні й комплексні корені рівняння полінома із точністю  $\varepsilon = 0,0001$ :

$$x^4 - 2x^3 + 12x^2 + 4x - 3 = 0.$$

21. Розкрийте суть методу Ньютона для розв'язання систем нелінійних рівнянь.

22. Охарактеризуйте принцип розкладання функції в ряд Тейлора.

23. Як визначається матриця Якобі?

24. Як перевірити збіжність ітераційного алгоритму розв'язання систем нелінійних алгебраїчних рівнянь методом Ньютона?

25. Які існують варіанти методу Ньютона для розв'язання систем нелінійних алгебраїчних рівнянь?

26. Як визначається похибка за ітераційного обчислення методом Ньютона для розв'язання систем нелінійних алгебраїчних рівнянь?

27. Дайте означення поняття «норма» й охарактеризуйте усі їх види в матричному обчисленні.

28. Обчисліть корені системи нелінійних рівнянь із точністю  $\varepsilon = 0,001$ :

$$\begin{cases} 2x^2 - y^2 + 9z^2 = 1; \\ x^2 + 6y^2 - z^2 = 0; \\ x^2 + 2y - 3z^2 = 4. \end{cases}$$

29. Обчисліть корені системи нелінійних рівнянь із точністю  $\varepsilon = 0,001$  і початковим наближенням:

$$\begin{cases} x_1 + 3 \ln x_1 - x^2 = 0; \\ 2x_1^2 - x_1 x_2 - 5x_1 = 1, \end{cases} \quad x^{(0)} = \begin{bmatrix} 3,4 \\ 2,2 \end{bmatrix}.$$

30. Виконайте п'ять ітерацій методом Ньютона пошуку значень коренів для системи нелінійних рівнянь за заданого вектора початкового наближення:

$$\begin{cases} x_2 = 2x_1^4 - 1; \\ x_2 = -5x_1^2 + 3x_1, \end{cases} \quad \begin{bmatrix} x_1^{(0)} \\ x_2^{(0)} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

## РОЗДІЛ 3 ЗАДАЧІ ДИФЕРЕНЦІАЛЬНОГО ЧИСЛЕННЯ

Сучасні комп'ютерні системи широко використовуються в усіх сферах людської діяльності. Однією із основних сфер застосувань комп'ютерних систем є розв'язання задач, які виникають у науці і техніці, а саме побудова рішень математичних моделей, які описують різні фізичні явища. Ця сфера застосування називається науковим програмуванням. Нині навряд чи можна знайти яку-небудь область науки і техніки, де б не використовувались високопродуктивні обчислювальні комп'ютерні системи, а саме: розрахунок траєкторій супутників Землі, моделювання потоку повітря навколо літака з метою проектування його ефективної конструкції, дослідження міцнісних характеристик будівельних конструкцій, прогнозування кліматичних умов тощо. Математичні моделі усіх згаданих вище проблем, а також більшості інших наукових та інженерних задач являють собою системи диференціальних рівнянь. Залежно від числа незалежних змінних і, відповідно, типу вхідних у них похідних, рівняння поділяються на звичайні диференціальні, які містять одну незалежну змінну і похідні по ній, і рівняння в частинних похідних, що мають декілька незалежних змінних і похідні (частинні) по них. Звичайними диференціальними рівняннями (ЗДР) можна описати задачі руху системи взаємодіючих матеріальних точок, кінетики, електричних кіл, опору матеріалів тощо. Ряд важливих задач для рівнянь в частинних похідних також зводиться до задач для ЗДР. Наприклад, якщо багатовимірною задачею допускає розділення змінних (наприклад, задачі на знаходження власних коливань пружних балок і мембран простішої форми або визначення спектра власних значень енергії частинки у сферично-симетричному полі), або якщо її розв'язок залежить тільки від деякої комбінації змінних (автомодальні рішення). Таким чином, розв'язання ЗДР займає важливе місце серед прикладних задач фізики, хімії і техніки. Тому в цьому розділі розкриваються основні методи чисельного дослідження ЗДР.

Звичайним диференціальним рівнянням називається рівняння, що містить невідомі функції, незалежну змінну і похідні невідомих функцій (або їх диференціали).

Загальний вигляд ЗДР:

$$F(x, y, y', y'', \dots, y^{(n)}) = 0 \text{ або } F\left(x, y, \frac{dy}{dx}, \frac{d^2y}{dx^2}, \dots, \frac{d^ny}{dx^n}\right) = 0, \quad (3.1)$$

де  $y=y(x)$  – функція, яка визначається;  $y^{(n)} = \frac{d^ny}{dx^n}$  – похідна (диференціал)  $n$ -го порядку функції  $y(x)$  від аргументу  $x$ ;  $F$  – дійсна функція від власних аргументів, які також вважаються дійсними.

Диференціальні рівняння бувають лінійними та нелінійними. Лінійним рівнянням називається рівняння, в якому невідома функція та її похідні

знаходяться у першому ступені. Порядком диференціального рівняння називається порядок найвищої із похідних (або диференціалів), що входять у рівняння (3.1). У загальному випадку розв'язок (3.1) визначається внаслідок  $n$  послідовних інтегрувань, так що загальний розв'язок  $n$ -го порядку містить  $n$  довільних постійних:

$$y = y(x, C_1, C_2, \dots, C_n). \quad (3.2)$$

Якщо загальний розв'язок рівняння (3.1) отримано в неявному вигляді:

$$\Phi(x, y, C_1, C_2, \dots, C_n) = 0, \quad (3.3)$$

тоді він називається загальним інтегралом цього рівняння. Іншими словами, в тих випадках, коли рівність (3.3) можна розв'язати відносно шуканої функції  $y$ , його називають загальним розв'язком диференціального рівняння (3.1), а якщо вона залишається нерозв'язаною відносно  $y$  – загальним інтегралом. Надавши довільним сталим  $C_1, C_2, \dots, C_n$  конкретні числові значення, буде отримано із (3.3) частинний інтеграл.

Для того, щоб із загального розв'язку виділити одне, частинне, необхідно до диференціального рівняння додати деякі додаткові умови. Зазвичай використовуються початкові умови, які під час дослідження процесу, що розвивається в часі, є математичним записом початкового стану процесу.

Для системи  $n$  звичайних диференціальних рівнянь першого порядку:

$$\frac{dU(x)}{dx} = F(x, U), \quad (3.4)$$

де  $U = (y_1, y_2, \dots, y_n)^T$ ,  $F(x, U) = (f_1(x, U), f_2(x, U), \dots, f_n(x, U))^T$  – вектори, загальний розв'язок містить  $n$  похідних сталих  $U = \Phi(x, C_1, \dots, C_n)$ , і для виділення його частинних розв'язків необхідно також задати  $n$  додаткових умов, а саме стільки, скільки рівнянь містить система.

ЗДР знаходять все більш широке застосування в математичних моделях, що розробляються для моделювання процесів і явищ, які відбуваються в різних галузях техніки, науки і виробництва.

У сфері мікробіології відомою є задача визначення залежності зростання числа бактерій протягом часу за умови, що у сприятливих для розмноження умовах на початку знаходиться деяка кількість  $N_0$  бактерій. Також експериментально відомо, що швидкість розмноження бактерій пропорційна їх кількості.

Для розв'язання цієї задачі доцільно через  $N(t)$  позначити кількість розмножувальних бактерій у момент часу  $t$ ,  $N(0) = N_0$ . Відволікаючись від того, що чисельність може вимірюватись тільки цілими числами, будемо вважати, що  $N(t)$  змінюється в часі неперервно і диференційовано. Тоді швидкість розмноження є похідною від функції  $N(t)$ , тому вказаний у задачі

біологічний експериментальний закон дозволяє скласти диференціальне рівняння розмноження бактерій:

$$\frac{dN(t)}{dt} = kN(t), \quad k > 0. \quad (3.5)$$

Експериментальний коефіцієнт  $k$  залежить від виду бактерій і умов, в яких вони знаходяться. Задача звелась до чисто математичної задачі знаходження розв'язку  $N = N(t)$  рівняння (3.5), для якого  $N(0) = N_0$ . Оскільки  $N(t) > 0$ , то розділивши обидві частини рівняння (3.5) на  $N(t)$ , отримаємо  $\frac{d}{dt}(\ln N(t)) = k$ , звідки:

$$\ln N(t) = kt + C_1, \quad (3.6)$$

де  $C_1$  – довільна стала.

Позначивши  $C_1 = \ln C$  ( $C > 0$ ), із рівняння (3.6) знаходимо:

$$N(t) = Ce^{kt}. \quad (3.7)$$

Щоб із множини функцій (3.7) виділити ту, яка описує процес розмноження бактерій, можна скористатись умовою  $N(0) = N_0$  тоді із рівняння (3.7) маємо  $N_0 = C$ . Остаточоно:

$$N(t) = N_0 e^{kt}, \quad (3.8)$$

показує, що чисельність бактерій зростає за показниковим законом.

У балістиці актуальним є визначення залежності швидкості вертикального падіння тіла масою  $m$  залежно від часу. Відомо, що початкова висота падіння дорівнює  $h$ , а сила в'язкого тертя, що діє на тіло, пропорційна величині швидкості:  $F_{mp} = -\alpha v$ , де  $\alpha > 0$  – коефіцієнт тертя.

Під час розв'язання цієї задачі вважаємо, що  $v(t)$  – швидкість тіла в момент часу  $t$ . На тіло діє дві протилежно направлені сили: тяжіння  $F_m = mg$  і в'язкого тертя  $F_{mp} = -\alpha v$ .

У такому випадку диференціальне рівняння описує другий закон Ньютона:

$$ma = F = F_m + F_{mp} \Rightarrow m \frac{dv}{dt} = mg - \alpha v. \quad (3.9)$$

Розділивши обидві частини рівняння (3.9) на  $m$ , отримуємо диференціальне рівняння:

$$\frac{dv}{dt} = -\frac{\alpha}{m}v + g. \quad (3.10)$$

Розв'язком диференціального рівняння (3.10) буде такий вираз:

$$v(t) = \frac{mg}{\alpha} + Ce^{-\frac{\alpha}{m}t}. \quad (3.11)$$



Якщо тіло починає рух із нульовою швидкістю ( $v(0) = 0$ ), тоді  $C = -\frac{mg}{\alpha}$ :

$$v(t) = \frac{mg}{\alpha} \left( 1 - e^{-\frac{\alpha t}{m}} \right). \quad (3.12)$$

Під час вільного падіння без тертя  $\left( \frac{dv}{dt} = g \right)$  величина швидкості зростає лінійно:  $v(t) = gt$ . За наявності в'язкого тертя швидкість, зростаючи, прямує до постійної величини  $v = \frac{mg}{\alpha}$ .

Багато реальних процесів моделюються диференціальними рівняннями, що містять другу похідну невідомої функції. Такі диференціальні рівняння називаються рівняннями другого порядку. Прикладом задачі із тієї самої сфери балістики є задача визначення закону руху точки масою  $m$ , яка падає вертикально вниз, водночас нехтуючи опором повітря.

Для розв'язання такої задачі на вертикальній осі, вздовж якої падає точка, обирається точка відліку  $O$  і визначається додатний напрям – від точки  $O$  вниз. Положення точки визначається координатою  $y(t)$ , яка змінюється з часом  $t$ . Точка падає під дією сили тяжіння  $F_T = mg$ , тому, згідно з другим законом Ньютона,  $ma = F$ . Тоді  $m \frac{d^2 y}{dt^2} = mg$  або:

$$\frac{d^2 y}{dt^2} = g. \quad (3.13)$$

Інтегруючи двічі співвідношення (3.13), визначаємо:

$$\frac{dy}{dt} = gt + C_1, \quad y(t) = \frac{gt^2}{2} + C_1 t + C_2. \quad (3.14)$$

Формула (3.14) визначає закон руху матеріальної точки, але вона містить сталі інтегрування, в цьому випадку – дві. Знаючи початкове положення падаючої точки відносно точки  $O$  –  $y(0) = y_0$  і її початкову швидкість  $v(t) = v_0$ , із сукупності функцій (3.14) обирається одна, що описує рух точки.

Оскільки швидкість руху точки  $v(t) = \frac{dy}{dt}$ , то за вказаних початкових умов  $C_1 = v_0$ ,  $C_2 = y_0$ , шукана функція, яка описує закон руху точки:

$$y = \frac{gt^2}{2} + v_0 t + y_0. \quad (3.15)$$

Таким чином, було отримано відому формулу шляху, пройденого точкою за рівномірно прискореного руху.

У цьому розділі розглянуто розв'язання обчислювальних задач диференціального числення: розв'язання звичайних диференціальних

рівнянь і їх систем. Причому вважається, що користувач вже знайомий з головними відомостями розділів математичного аналізу.

### 3.1 Узагальнена постановка задачі для звичайних диференціальних рівнянь

Розрізняють три основних типи задач для ЗДР: задачі Коші, крайові задачі і задачі на власні значення.

Постановка задачі Коші для системи  $n$  ЗДР першого порядку в загальному випадку сформулюється таким чином. Знайти розв'язок диференціального рівняння:

$$\frac{dU(x)}{dx} = F(x, U) \text{ при } x > x_0, U(x_0) = U^0, \quad (3.16)$$

де  $x_0$  – початкове значення  $x$ ;  $U^0$  – початкове значення вектора  $U$  ( $U = (y_1, y_2, \dots, y_n)^T$ );  $F(x, U) = (f_1(x, U), f_2(x, U), \dots, f_n(x, U))^T$ , або в розгорнутому вигляді:

$$\frac{dy_i(x)}{dx} = f_i(x, y_1, y_2, \dots, y_n); x > x_0; y_i(x_0) = y_i^{(0)} \quad (i = 1, 2, \dots, n).$$

Для рівнянь першого –  $n$ -го порядків задача Коші формулюється таким чином:

знайти розв'язок диференціальних рівнянь:

а)  $\frac{dy}{dx} = f(x, y)$  для  $x > x_0$  за  $y(x_0) = C_0$ ;

б)  $\frac{d^2 y}{dx^2} = f(x, y, y')$  для  $x > x_0$  за  $y(x_0) = C_0, y'(x_0) = C_1$ ;

в)  $\frac{d^n y}{dx^n} = f(x, y, y', \dots, y^{(n)})$  для  $x > x_0$  за  $y(x_0) = C_0, y'(x_0) = C_1, \dots, y^{(n-1)}(x_0) = C_{n-1}$ ,

де  $C_1, C_2, \dots, C_n$  – деякі константи;  $y', y'', \dots, y^{(n)}$  – похідні функції  $y$ .

### 3.2 Чисельні методи розв'язання звичайних диференціальних рівнянь для задач типу Коші

Загалом методи розв'язання ЗДР умовно поділяються на точні, наближені й чисельні. До точних належать аналітичні методи, які дозволяють виразити розв'язок диференціального рівняння через

елементарні функції або подати його за допомогою квадратур від елементарних функцій. Знаходження точного, до того ж загального, розв'язку задачі (3.16) полегшує якісне дослідження цього розв'язку і подальшої дій з ним. Проте класи рівнянь, для яких розроблено методи отримання точних рішень, порівняно вузькі і охоплюють тільки малу частину задач, що виникають на практиці.

Наближеними називаються методи, в яких розв'язок буде отримано як межу  $y(x)$  деякої послідовності  $y_n(x)$ , причому  $y_n(x)$  виражаються через елементарні функції або за допомогою квадратур. Обмежуючись кінцевим числом  $n$ , буде отримано приблизний вираз для  $y(x)$ . До наближених методів відносяться: розкладання розв'язку в узагальнений степеневий ряд, метод Чаплигіна, метод Пікара, Канторовича та інші. Проте ці методи зручні в тому випадку, коли більшу частину проміжних операцій вдається виконати точно (наприклад, знайти явний вираз коефіцієнтів ряду). Це можна виконати тільки для порівняно простих задач (лінійних), що сильно звужує сферу застосування приблизних методів.

Чисельні методи – це алгоритми обчислення приблизних (а іноді й точних) значень потрібного розв'язку  $y(x)$  на деякій обраній сітці значень аргументу  $x_n$ . Розв'язок буде отримано у вигляді таблиці. Чисельні методи не дозволяють знайти загального розв'язку системи (3.16); вони можуть дати тільки деякий частинний розв'язок, наприклад, розв'язок задачі Коші (3.16). Проте ці методи можуть бути застосовані до широкого класу рівнянь і усіх типів задач для них.

Чисельні методи можуть бути застосовані тільки до коректно поставлених (або регульованих) задач. Проте потрібно відмітити, що для успішного застосування чисельних методів формальне виконання умов коректності може бути недостатнім. Необхідно, щоб задача була добре обумовленою, тобто малі зміни початкових умов приводили б до достатньо малої зміни інтегральних кривих. Якщо ця умова не виконується, тобто задача погано обумовлена (слабо стійка), тоді невеликі зміни початкових умов або еквівалентні цим змінам невеликі похибки чисельного методу можуть вносити сильні спотворення.

Наявні для розв'язання задач Коші чисельні методи класифікуються на два види:

1) однокрокові методи, в яких для знаходження наступної точки на кривій необхідна інформація лише про один попередній крок. Відомими однокроковими методами є методи Ейлера і Рунге-Кутта.

2) методи «прогнозування та корекції» (багатокрокові методи), в яких для пошуку наступної точки кривої необхідна інформація більш ніж про одну з попередніх точок. З метою отримання досить точного чисельного значення часто використовують ітерації. До таких методів належать методи Адамса, Мілна, Моултона тощо.

## Метод Ейлера

Метод Ейлера є найпростішим методом розв'язання задачі Коші (3.16) для  $x \in [a; b]$  та  $i = 1$ . З метою отримання розрахункових формул область неперервної зміни аргументу  $x$  може бути замінена множиною точок  $x_j = a + jh$ , які будуть називатись вузлами сітки:

$$h = \frac{b-a}{n}; \quad x_0 = a; \quad x_n = b,$$

де  $h$  – крок сітки.

Чисельне розв'язання задачі (3.16) являє собою таблицю значень:

$$x_j, y_j \quad (j = 0, 1, \dots, n),$$

де  $y_j$  – різницеве або чисельне значення розв'язку у вузлі  $x_j$ .

Рівняння (3.16) для  $x = x_j$  і  $n = 1$ , за означенням похідної:

$$\left. \frac{dy}{dx} \right|_{x=x_j} = \lim_{h \rightarrow 0} \frac{y(x_j + h) - y(x_j)}{h}. \quad (3.17)$$

Відкидаючи межу у (3.17), похідна  $\left. \frac{dy}{dx} \right|_{x=x_j}$  замінюється кінцево-різницеvim відношенням:

$$\left. \frac{dy}{dx} \right|_{x=x_j} \approx \frac{y(x_j + h) - y(x_j)}{h}. \quad (3.18)$$

Якщо підставити (3.18) в (3.16), отримаємо:

$$\frac{y(x_j + h) - y(x_j)}{h} \approx f(x_j, y(x_j)) \quad (j = 0, 1, \dots, N-1), \quad (3.19)$$

де  $y(x_j)$  – значення розв'язку  $y(x)$  задачі Коші у вузлі  $x_j$ . Позначивши через  $y_j$  чисельний розв'язок, який задовольняє різницеве рівняння, можна записати:

$$\frac{y_{j+1} - y_j}{h} = f(x_j, y_j) \quad (j = \overline{0, n-1}) \quad (3.20)$$

або

$$y_{j+1} = y_j + hf(x_j, y_j) \quad (j = \overline{0, n-1}). \quad (3.21)$$

Враховуючи початкову умову (3.16), за допомогою різницевого рівняння (3.21) можна послідовно визначити ряд ітераційних рівнянь:

$$y_1 = y_0 + hf(x_0, y_0) \quad (j=1);$$

$$y_2 = y_1 + hf(x_1, y_1) \quad (j=2);$$

$$\dots;$$

$$y_n = y_{n-1} + hf(x_{n-1}, y_{n-1}) \quad (j=n-1).$$

Графічною інтерпретацією наближеного розв'язку, отриманого за методом Ейлера, є ламана крива (рис. 3.1, а), що з'єднує послідовно точки  $M_0, M_1, \dots, M_n$  та називається ламаною Ейлера.

Для геометричної інтерпретації похибки, яка виникає у разі застосування методу Ейлера, необхідно розглянути точку  $x_n$ , в якій отримано чисельний розв'язок задачі  $y_n$ . Покладаючи  $y_n = y(x_n)$ , через цю точку  $(y_n, x_n)$  проводиться гіпотетична інтегральна крива  $y(x)$  і дотична до неї  $f(x_n, y(x_n))$  (рис. 3.1, б). Вона за взятих припущень дорівнює  $f(x_n, y_n)$ .

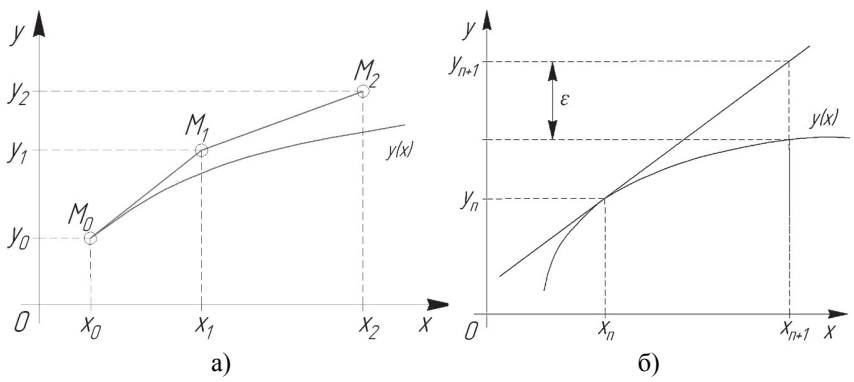


Рисунок 3.1 – Схема геометричної інтерпретації методу Ейлера чисельного розв'язання задачі Коші:  
 а) – схема апроксимації; б) – схема визначення похибки

Точка перетину цієї дотичної із перпендикуляром до осі  $x$ , яка проходить через точку  $x_{n+1}$ , дає наближене значення функції  $y_{n+1}$  в точці  $x_{n+1}$ . Водночас похибка розв'язку дорівнює  $\varepsilon = y_{n+1} - y(x_{n+1})$ . Відповідно, метод Ейлера – це лінійна екстраполяція функції  $y_n$  в точку  $x_{n+1}$  за значеннями функції та її похідної в точці  $x_n$ :  $y_{n+1} = y_n + f(x_n, y_n)(x_{n+1} - x_n)$ .

На основі розкладу в ряд Тейлора рівняння (3.21) визначено, що метод Ейлера має велику похибку, а саме похибку першого порядку  $O(h)$  і часто виявляється нестійким, оскільки мала похибка в початкових даних або через округлення в процесі обчислень збільшується із зростанням  $x$ .

З метою підвищення точності розв'язання задачі Коші для ЗДР використовують уточнений метод Ейлера. Цей метод базується на обчис-

ленні функції  $y(x_{n+1})$  у наступній точці  $x_{n+1}$  по значенню середньоарифметичної величини тангенсів кутів нахилу дотичної до інтегральної кривої  $y(x)$  в двох точках  $x_n$  і  $x_{n+1}$ . В цьому випадку припускається відомим розв'язок задачі  $y_n$  в точці  $x_n$ .

Метод складається із двох кроків:

1) за методом Ейлера (3.21) попередньо визначається наближене значення  $\bar{y}_{n+1}$  в точці  $x=x_n+h$  за формулою

$$\bar{y}_{n+1} = y_n + hf(x_n, y_n), \quad (3.22)$$

і в ній обчислюється функція  $f(x_{n+1}, \bar{y}_{n+1})$ ;

2) тангенси кутів нахилу дотичних у точках  $(x_n, y_n)$  і  $(x_{n+1}, \bar{y}_{n+1})$  додаються і визначається середнє арифметичне значення  $\Phi(x_n, y_n, h)$ :

$$\Phi(x_n, y_n, h) = \frac{1}{2} [f(x_n, y_n) + f(x_{n+1}, \bar{y}_{n+1})], \quad (3.23)$$

де  $f(x_{n+1}, \bar{y}_{n+1}) = f(x_n + h, y_n + hf(x_n, y_n))$ .

Після чого визначається кінцеве (уточнене) значення функції  $y_{n+1}$  в точці  $x_{n+1}$  за формулою  $y_{n+1} = y_n + h\Phi(x_n, y_n, h)$  або в розгорнутому вигляді:

$$y_{n+1} = y_n + \frac{h}{2} [f(x_n, y_n) + f(x_n + h, y_n + hf(x_n, y_n))]. \quad (3.24)$$

Таким чином, під час обчислення  $y_{n+1}$ , функцію  $f(x, y)$  доводиться обчислювати двічі в точках  $(x_n, y_n)$  і  $(x_n+h, y_n+h y'_n)$ .

За допомогою уточненого методу Ейлера можна виконувати контроль точності шляхом порівняння значення  $\bar{y}_{n+1}$  із формули (3.22) і  $y_{n+1}$  із формули (3.24), що дозволить на основі цього вибирати відповідне значення кроку  $h$  в кожному розрахунковому вузлі. Тобто, якщо значення  $|\bar{y}_{n+1} - y_{n+1}|$  порівнянне із похибками обчислень, тоді крок необхідно збільшити; в іншому випадку, якщо ця різниця достатньо велика (наприклад,  $|\bar{y}_{n+1} - y_{n+1}| > 0,01|\bar{y}_{n+1}|$ ), значення  $h$  необхідно зменшити. Використовуючи ці оцінки, можна побудувати алгоритм методу Ейлера із автоматичним вибором кроку (рис. 3.2).

З метою геометричної інтерпретації уточненого методу Ейлера в точці  $(x_n, y_n)$  (рис. 3.3) будується дотична  $L_1$  до гіпотетичної інтегральної кривої  $y(x)$ , і визначається попереднє значення  $\bar{y}_{n+1}$  в точці перетину цієї дотичної із перпендикуляром до осі  $x$ , що проходить через точку  $x_{n+1}$ . Після чого в точці  $(x_n+h, y_n+h y'_n)$  знову будується дотична  $L_2$  до кривої  $y(x)$ .

Далі визначається середньоарифметичне тангенсів кутів нахилу цих дотичних (крива  $L_3$ ) і проводиться через цю точку  $(x_n, y_n)$  лінія  $L_0$ , паралельна лінії  $L_3$ . Точка перетину цієї лінії із ординатою, що проходить через точку  $x_{n+1}$ , дає уточнене значення функції  $y_{n+1}$  в точці  $x_{n+1}$ .

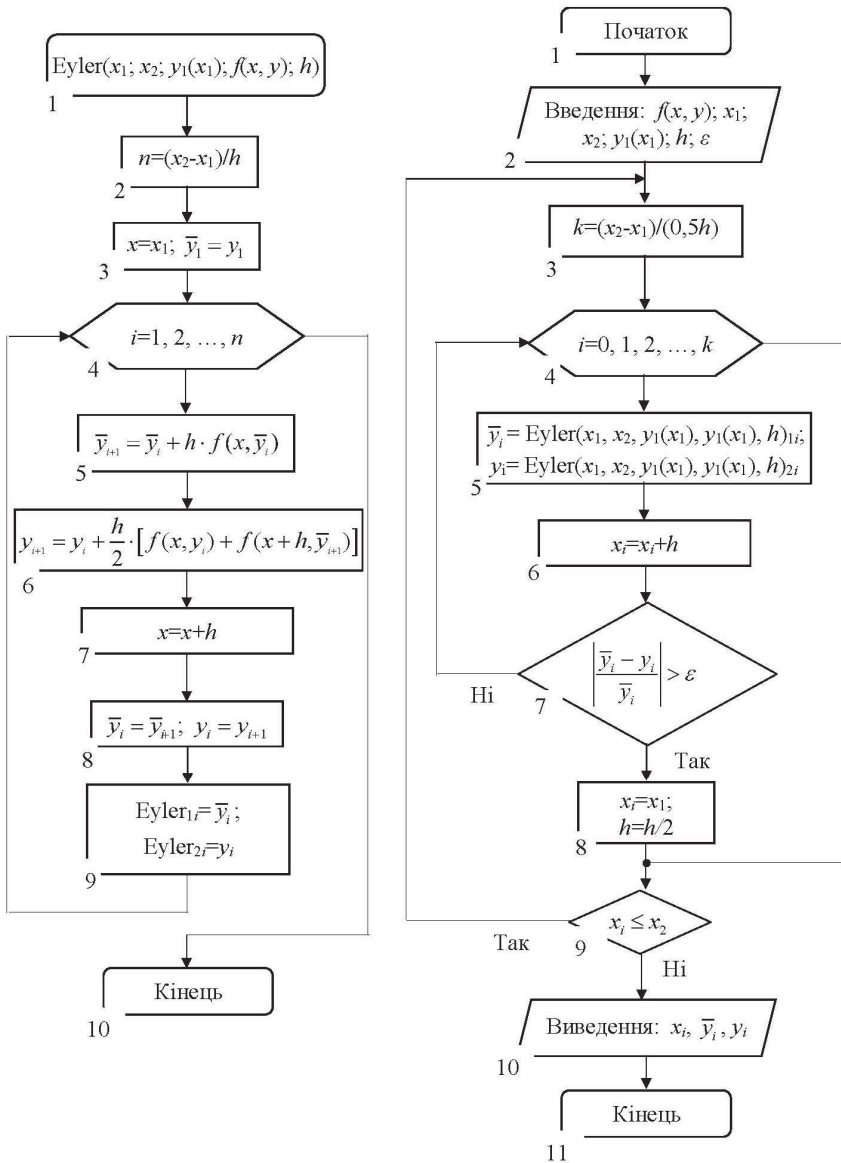


Рисунок 3.2 – Схема алгоритму метода Ейлера

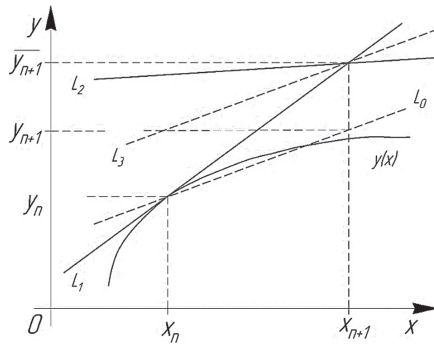


Рисунок 3.3 – Схема геометричної інтерпретації уточненого методу Ейлера

Оцінення похибки апроксимації формули (3.24) на основі розкладу в ряд Тейлора функцій  $y(x_n)$  і  $f(x_n + h, y_n + hf(x_n, y_n))$  показує, що цей метод має другий порядок апроксимації  $O(h^2)$ .

Метод Ейлера успішно застосовується для систем ЗДР. Для системи  $n$  ЗДР першого порядку метод Ейлера має такий вигляд:

$$\begin{cases} \frac{\overline{y}_{j+1} - \overline{y}_j}{h} = \overline{f}(x_j, \overline{y}_j) \quad (j = 0, 1, \dots, n-1); \\ \overline{y}(0) = \overline{y}_0; \quad x \in [0, 1], \end{cases} \quad (3.25)$$

де  $\overline{y}_j = (y_{1,j}, y_{2,j}, \dots, y_{n,j})^T$ .

Тоді метод Ейлера переписується для кожної компоненти вектора  $\overline{y}_j$ :

$$\begin{cases} \frac{y_{1,j+1} - y_{1,j}}{h} = f_{1,j}; \\ \frac{y_{2,j+1} - y_{2,j}}{h} = f_{2,j}; \\ \dots; \\ \frac{y_{n,j+1} - y_{n,j}}{h} = f_{n,j} \quad (j = 0, N-1); \\ y_1(0) = y_{10}, y_2(0) = y_{20}, \dots, y_n(0) = y_{n0}. \end{cases}$$

Для оцінення загальної похибки чисельного розв'язання задачі Коші для систем ЗДР може бути використаний метод Рунге на основі залежності, яка порівнює результати обчислення значень  $\overline{y}_n$  і  $\tilde{y}_n$  в точці  $x_n$  із різними кроками  $\tilde{h}$  та  $\tilde{h}/2$ , відповідно похибка зрізу:



$$R_n^h = (\tilde{y}_n - \bar{y}_n) / (1 - 2^{-p}), \quad (3.26)$$

де  $p$  – порядок апроксимації використовуваної різницевої схеми, зокрема  $p = 1$  – метод Ейлера.

Алгоритми методу Ейлера для систем ЗДР із автоматичним вибором кроку наведено на рисунку 3.4.

**Приклад 3.1.** Розв’язати чисельними методами Ейлера задачу Коші для ЗДР першого порядку:

$$\frac{dy}{dx} = y + x; \quad x \in [0; 1,0],$$

яка задовольняє початкову умову за  $x_0 = 0, y_0 = 1,0$  і відносної похибки обчислення  $\varepsilon = 10,0\%$ .

*Розв’язання:*

Для розв’язання цієї задачі відрізок  $[0; 1,0]$  можна розділити на десять частин точками  $x_0 = 0; 0,1; 0,2; \dots, 1,0$ . Відповідно  $h = 0,1$ . Значення  $y_1, y_2, \dots, y_n$  будуть визначатись методом Ейлера за формулою (3.21), тоді:

$$y_1 = y_0 + hf(x_0, y_0) = 1 + 0,1 \cdot (0 + 1) = 1,10;$$

$$y_2 = y_1 + hf(x_1, y_1) = 1,1 + 0,1 \cdot (1,1 + 0,1) = 1,21;$$

.....

Також значення  $y_1, y_2, \dots, y_n$  можуть бути визначені уточненим методом Ейлера за формулою (3.24), тоді:

$$\begin{aligned} \bar{y}_1 &= y_0 + \frac{h}{2} [f(x_0, y_0) + f(x_0 + h, y_0 + hf(x_0, y_0))] = \\ &= 1,0 + \frac{0,1}{2} [(0 + 1,0) + (0 + 0,1 + 1,0 + 0,1 \cdot (0 + 1,0))] = 1,11000; \end{aligned}$$

$$\begin{aligned} \bar{y}_2 &= \bar{y}_1 + \frac{h}{2} [f(x_1, \bar{y}_1) + f(x_1 + h, \bar{y}_1 + hf(x_1, \bar{y}_1))] = \\ &= 1,11 + \frac{0,1}{2} [(0,1 + 1,11) + (0,1 + 0,1 + 1,11 + 0,1 \cdot (0,1 + 1,11))] = 1,24205; \end{aligned}$$

.....

Також для порівняльного аналізу точності розв’язання звичайним і уточненим методом Ейлера було отримано аналітичний розв’язок ЗДР, а саме:

$$\tilde{y} = 2e^x - x - 1. \quad (3.27)$$

У процесі розв’язання складено таблицю 3.1 результатів обчислення.

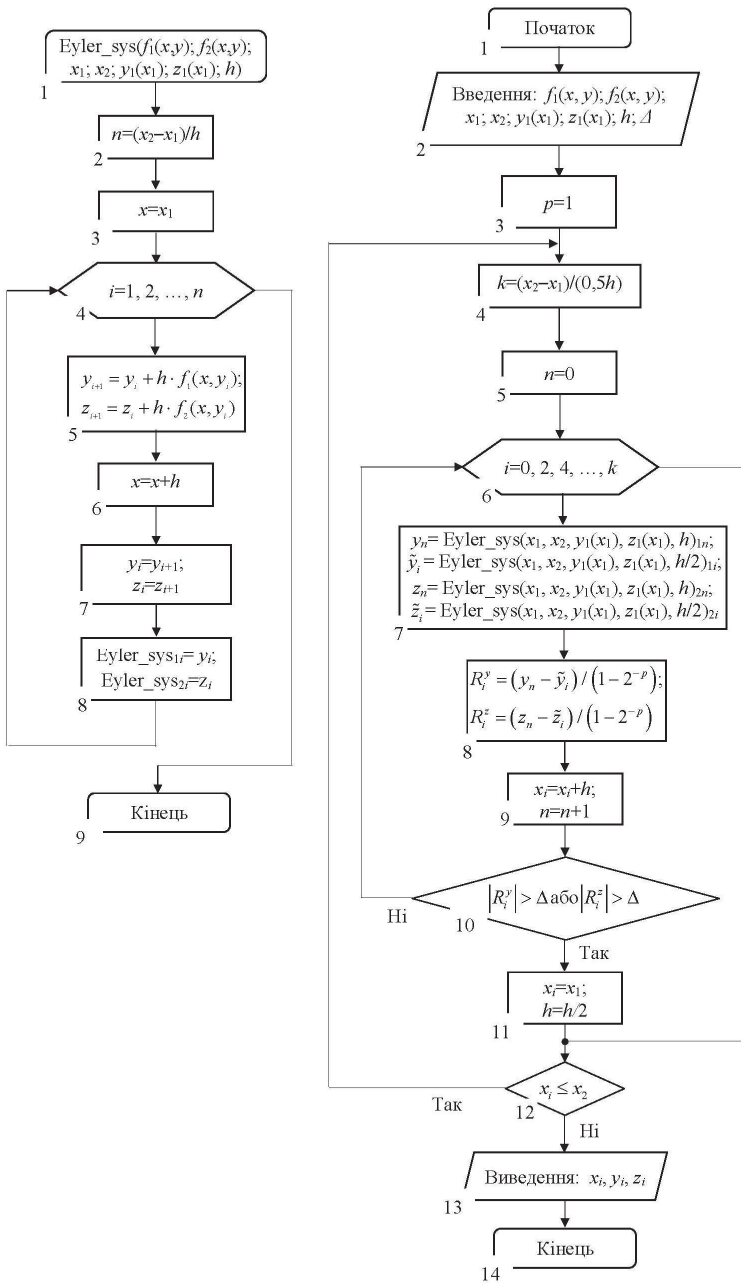


Рисунок 3.4 – Схема алгоритму методу Ейлера для систем ЗДР

Таблиця 3.1 – Результати обчислення розв’язку диференціального рівняння

$n$	$x_n$	$y_n(x_n)$	$\bar{y}_n(x_n)$	$\tilde{y}_n(x_n)$
0	0,0	1,0000	1,00000	1,00000
1	0,1	1,1000	1,11000	1,11034
2	0,2	1,2200	1,24205	1,24281
3	0,3	1,3620	1,39847	1,39972
4	0,4	1,5240	1,58180	1,58365
5	0,5	1,7164	1,79489	1,79744
6	0,6	1,9380	2,04086	2,04424
7	0,7	2,1918	2,32315	2,32751
8	0,8	2,4730	2,64558	2,65108
9	0,9	2,8003	3,01236	3,01921
10	1,0	3,1703	3,42816	3,43656

Із отриманих результатів, наведених у таблиці 3.1, видно, що похибки, допущені під час визначення розв’язку, зростають до кінця таблиці. Зокрема, в точках  $x_5 = 0,5$  і  $x_{10} = 1,0$  відносна похибка:

– шляхом порівняння між значеннями аналітичного розв’язку і звичайним методом Ейлера

$$\varepsilon_{x_5} = \left| \frac{\tilde{y}_{x_5} - y_{x_5}}{\tilde{y}_{x_5}} \right| \cdot 100\% = \left| \frac{1,79744 - 1,71640}{1,79744} \right| \cdot 100\% = 4,51\%,$$

$$\varepsilon_{x_{10}} = \left| \frac{\tilde{y}_{x_{10}} - y_{x_{10}}}{\tilde{y}_{x_{10}}} \right| \cdot 100\% = \left| \frac{3,43656 - 3,17030}{3,43656} \right| \cdot 100\% = 7,75\%;$$

– шляхом порівняння між значеннями аналітичного розв’язку і уточненого метода Ейлера

$$\bar{\varepsilon}_{x_5} = \left| \frac{\tilde{y}_{x_5} - \bar{y}_{x_5}}{\tilde{y}_{x_5}} \right| \cdot 100\% = \left| \frac{1,79744 - 1,79489}{1,79744} \right| \cdot 100\% = 0,14\%,$$

$$\bar{\varepsilon}_{x_{10}} = \left| \frac{\tilde{y}_{x_{10}} - \bar{y}_{x_{10}}}{\tilde{y}_{x_{10}}} \right| \cdot 100\% = \left| \frac{3,43656 - 3,42816}{3,43656} \right| \cdot 100\% = 0,24\%.$$

Чисельні результати обчислення розв’язку диференціального рівняння, які зведено в таблиці 3.1, зручно подати на графіку (рис. 3.5).

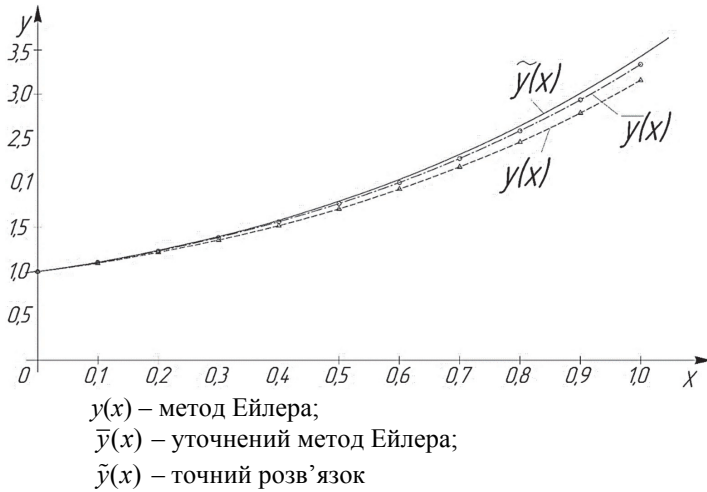


Рисунок 3.5 – Діаграма результатів чисельного розв'язку задачі Коші ЗДР

Також можна відмітити досить високу точність уточненого метода Ейлера в процесі розв'язання задачі Коші ЗДР, яка порівняно із звичайним методом Ейлера приблизно в десять разів вища.

Розглянемо реалізацію методу Ейлера мовою програмування PYTHON:

```

%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
#введення кроку інтегрування
h_0=float(input())
#введення початкового значення аргументу інтегрування (загального проміжку
# інтегрування)
a=int(input()); x_mtr=np.array(a)
#введення початкового значення аргументу функції диф. рівняння
y_0=int(input()); y_mtr=np.array(y_0)
#введення кінцевого значення загального проміжку інтегрування
b=int(input())
#введення значення відносної похибки обчислення
e=float(input())/100
#задання функції диф. рівняння
def f(x,y):
    return x+y
#обчислення ітераційних формул
def euler(a,b,y_0,h):
    x=a; y=y_0; y_plus=y_0
    x_mas=[x]; y_mas=[y]; y_masplus=[y]
    x_mtr=np.array(a)
    y_mtr=np.array(y_0)
  
```

```

while x<=b:
    #ітераційна формула методу Ейлера
    y=y+(h*f(x,y))
    x=x+h
    #ітераційна формула уточненого методу Ейлера
    y_plus=y_plus+((h/2)*(f(x-h,y_plus)+f(x,y_plus+(h*f(x,y_plus))))))
    x_mas.append(x)
    y_mas.append(y)
    x_mtr=np.append(x_mtr, x)
    y_masplus.append(y_plus)
return x_mas, y_mas, y_masplus, x_mtr
#контроль точності обчислення і коригування кроку обчислень у вузлі
h=h_0; x=a
while x<=b:
    for k in range(0,len(eyler(a,b,y_0,h)[0])):
        x=x+h
        if abs(eyler(a,b,y_0,h)[1][k]-
eyler(a,b,y_0,h)[2][k])>0.01*abs(eyler(a,b,y_0,h)[1][k]):
            h=h/2; x=a; break
#будуємо графіки рішень диф. рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
plt.plot(eyler(a,b,y_0,h)[0], eyler(a,b,y_0,h)[1])
plt.plot(eyler(a,b,y_0,h)[3], (2*(np.exp(eyler(a,b,y_0,h)[3]))) -
eyler(a,b,y_0,h)[3]-1)
plt.plot(eyler(a,b,y_0,h)[0], eyler(a,b,y_0,h)[2])
plt.legend(['Метод Ейлера', 'f(x)=2*exp(x)-x-1', 'Уточнений метод Ейлера'],
loc=1)
plt.grid(True)
plt.xlim([0, 1])
plt.ylim([1, 4])
plt.show().

```

**Приклад 3.2.** Відомо, що під час знаходження розв'язків диференціальних рівнянь Лапласа і Гельмгольца в циліндричних і сферичних координатах, а також під час розв'язування задач поширення хвиль, статичних потенціалів тощо виникає диференціальне рівняння Бесселя:

$$y'' + \frac{1}{x}y' + y = 0; \quad x \in [0, 1, 0], \quad (3.28)$$

знайти розв'язок з початковими умовами  $x_0 = 1, 0$ ;  $y_0 = 0, 765$ ;  $y'_0 = -0, 440$  і похибкою зрізу  $R^h = 0, 005$ .

*Розв'язання:*

Для розв'язання цієї задачі відрізок  $[1, 0; 2, 0]$  можна розділити на десять частин точками  $x_0 = 1, 0; 1, 1; 1, 2; \dots, 2, 0$ . Відповідно  $h = 0, 1$ .

Для застосування чисельних методів Ейлера необхідно диференціальне рівняння другого порядку (3.28) звести до системи двох рівнянь першого порядку з двома невідомими функціями. Це може бути досягнуто підстановкою  $y' = z$ . Оскільки  $y'' = z'$ , то рівняння (3.28) може бути записано у вигляді системи:

$$\begin{cases} y' = z; \\ z' = -\frac{z}{x} - y; \\ x_0 = 1, 0; y_0 = 0, 765; z_0 = -0, 440. \end{cases} \quad (3.29)$$

Значення  $y_1, y_2, \dots, y_n$  і  $z_1, z_2, \dots, z_n$  будуть визначатись методом Ейлера за формулою (3.22), а саме:

$$\begin{cases} \frac{y_1 - y_0}{h} = z_0; \\ \frac{z_1 - z_0}{h} = -\frac{z_0}{x_0} - y_0; \end{cases} \Rightarrow \begin{cases} y_1 = y_0 + h z_0; \\ z_1 = z_0 + h \left( -\frac{z_0}{x_0} - y_0 \right); \end{cases}$$

$$\begin{cases} \frac{y_2 - y_1}{h} = z_1; \\ \frac{z_2 - z_1}{h} = -\frac{z_1}{x_1} - y_1; \end{cases} \Rightarrow \begin{cases} y_2 = y_1 + h z_1; \\ z_2 = z_1 + h \left( -\frac{z_1}{x_1} - y_1 \right); \end{cases}$$

.....;

$$\begin{cases} \frac{y_{n+1} - y_n}{h} = z_n; \\ \frac{z_{n+1} - z_n}{h} = -\frac{z_n}{x_n} - y_n; \end{cases} \Rightarrow \begin{cases} y_{n+1} = y_n + h z_n; \\ z_{n+1} = z_n + h \left( -\frac{z_n}{x_n} - y_n \right); \end{cases} \quad (3.30)$$

$$x_0 = 1, 0; y_0 = 0, 765; z_0 = -0, 440.$$

Також для порівняльного аналізу точності розв'язання звичайним і уточненим методом Ейлера було отримано аналітичний розв'язок диференціального рівняння Бесселя (3.28) у вигляді розкладу у ряд Тейлора, а саме, функція Бесселя першого роду нульового порядку:

$$\tilde{y}(x) = J_0(x) = \sum_{k=0}^{\infty} \frac{(-1)^k}{k!(k+1)!} \left( \frac{x}{2} \right)^{2k}, \quad (3.31)$$

де  $k$  – ціле число.

Також похідна від функції Бесселя першого роду нульового порядку дорівнює від'ємній функції Бесселя першого порядку:

$$\tilde{y}'(x) = J_0'(x) = -J_1(x). \quad (3.32)$$

Значення функцій Бесселя (3.31) і (3.32) визначаються за допомогою готових табличних даних, які наведено у таблиці 3.2 для  $x \in [1, 0; 2, 0]$ . Для визначення похибки обчислення методом Рунге (3.26) були визначені значення  $\bar{y}_n(x_n)$  із кроком  $h/2=0,05$ . У процесі розв'язання було складено таблицю 3.2 результатів обчислення.

Таблиця 3.2 – Результати обчислення розв’язку диференціального рівняння

$n$	$x_n$	$y_n(x)$	$\bar{y}_n(x_n)$	$y'_n(x_n)$	$\tilde{y}_n$	$\tilde{y}'_n(x_n)$
0	1,0	0,76500	0,76500	-0,44000	0,7652	-0,4400
1	1,1	0,72100	0,72019	-0,47250	0,7196	-0,4709
2	1,2	0,67375	0,67229	-0,50165	0,6711	-0,4983
3	1,3	0,62359	0,62166	-0,52722	0,6201	-0,5220
4	1,4	0,57086	0,56866	-0,54902	0,5669	-0,5419
5	1,5	0,51596	0,51367	-0,56689	0,5118	-0,5579
6	1,6	0,45927	0,45709	-0,58069	0,4554	-0,5699
7	1,7	0,40120	0,39933	-0,59033	0,3980	-0,5778
8	1,8	0,34217	0,34080	-0,59572	0,3400	-0,5815
9	1,9	0,28260	0,28192	-0,59684	0,2818	-0,5812
10	2,0	0,22291	0,22311	-0,59369	0,2239	-0,5767

Із отриманих результатів, наведених у таблиці 3.2, видно, що похибки, допущені під час визначення розв’язку, зростають до кінця таблиці для значень першої похідної, а саме  $y'_n(x_n)$  і  $\tilde{y}'_n(x_n)$ . Зокрема, в точках  $x_5 = 1,5$  і  $x_{10} = 2,0$ :

– відносна похибка шляхом порівняння між значеннями аналітичного розв’язку і методом Ейлера

$$\varepsilon_{x_5} = \left| \frac{\tilde{y}_{x_5} - y_{x_5}}{\tilde{y}_{x_5}} \right| \cdot 100\% = \left| \frac{0,51596 - 0,51180}{0,51596} \right| \cdot 100\% = 0,81\%,$$

$$\varepsilon_{x_{10}} = \left| \frac{\tilde{y}_{x_{10}} - y_{x_{10}}}{\tilde{y}_{x_{10}}} \right| \cdot 100\% = \left| \frac{0,22390 - 0,22291}{0,22390} \right| \cdot 100\% = 0,44\%;$$

– похибка зрізу методом Рунге (див. 3.26)

$$R_{x_5}^h = (y_{x_5} - \bar{y}_{x_5}) / (1 - 2^{-p}) = (0,51596 - 0,51367) / (1 - 2^{-1}) = 0,0046;$$

$$R_{x_{10}}^h = (y_{x_{10}} - \bar{y}_{x_{10}}) / (1 - 2^{-p}) = (0,22291 - 0,22311) / (1 - 2^{-1}) = -0,0004.$$

Чисельні результати обчислення розв’язку диференціального рівняння (3.28), подані в таблиці 3.2, зручно зобразити на графіку (рис. 3.6).

Із отриманих результатів, наведених у таблиці 3.2 і рисунку 3.6, видно, що абсолютна і відносна похибки, допущені під час визначення розв’язку, зростають у середині розглянутого проміжку  $x \in [1,0; 2,0]$ .

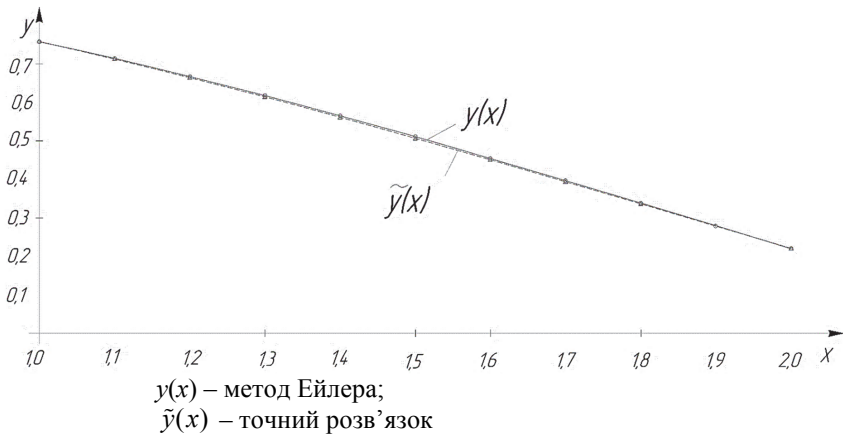


Рисунок 3.6 – Діаграма результатів чисельного розв'язку задачі Коші ЗДР вищого порядку

Розглянемо реалізацію методу Ейлера мовою програмування PYTHON:

```

%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
#введення кроку інтегрування
h_0=float(input())
#введення початкового значення аргументу інтегрування (загального проміжку
  інтегрування) x0
a=int(input()); x_mtr=np.array(a)
#введення початкового значення аргументу функції диф. рівняння y0
y_0=float(input()); y_mtr=np.array(y_0)
#введення початкового значення аргументу функції диф. рівняння z0
z_0=float(input()); z_mtr=np.array(z_0)
#введення кінцевого значення загального проміжку інтегрування
b=int(input())
#введення допустимої похибки обчислення
e_0=float(input())
#задання функцій системи диф. рівняння
def f1(z):
    return z
def f2(x,y,z):
    return (-z/x)-y
#обчислення ітераційних формул
def euler(a,b,y_0,z_0,h):
    x=a; y=y_0; z=z_0
    x_mas=[x]; y_mas=[y]; z_mas=[z]
    x_mtr=np.array(a)
    y_mtr=np.array(y_0)
    z_mtr=np.array(z_0)
    while x<b:
        #ітераційна формула методу Ейлера
        y_buf=y
        z_buf=z
  
```



```

y=y+(h*f1(z_buf))
z=z+(h*f2(x,y_buf,z_buf))
x=x+h
x_mas.append(x)
y_mas.append(y)
z_mas.append(z)
return x_mas, y_mas, z_mas
y_Bessel=[0.7652, 0.7196, 0.6711, 0.6201, 0.5669,
           0.5118, 0.4554, 0.3980, 0.34, 0.2818, 0.2239]
#визначення похибки обчислень методом Рунге і коригування кроку
# інтегрування
p=1; x=a; h=h_0
while x<=b:
    n=0
    for k in range(0,len(eyler(a,b,y_0,z_0,h/2)[1]),2):
        Rh_y=(eyler(a,b,y_0,z_0,h)[1][n]-
              eyler(a,b,y_0,z_0,h/2)[1][k])/(1-(1/2**p))
        Rh_z=(eyler(a,b,y_0,z_0,h)[2][n]-
              eyler(a,b,y_0,z_0,h/2)[2][k])/(1-(1/2**p))
        #print(abs(Rh_y))
        #print(abs(Rh_z))
        n=n+1
        x=x+h
        if abs(Rh_y)>e_0 or abs(Rh_z)>e_0:
            h=h/2; x=a
            break
#будуємо графіки рішень диф. рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
plt.plot(eyler(a,b,y_0,z_0,h)[0], eyler(a,b,y_0,z_0,h)[1])
plt.plot(eyler(a,b,y_0,z_0,h_0)[0], y_Bessel)
plt.legend(['Метод Ейлера', 'Точний розв'язок на основі обчислювальних
таблиць'], loc=1)
plt.grid(True)
plt.xlim([1, 2])
plt.ylim([0.2, 0.8])
plt.show()

```

## Метод Рунге-Кутта

Для побудови різницевої схеми інтегрування чисельного розв'язання задачі Коші ЗДР необхідно використати розкладання рівняння (3.16) для  $i = 1$  в ряд Тейлора:

$$y(x_{k+1}) = y(x_k) + y'(x_k)h + y''(x_k)\frac{h^2}{2!} + \dots + y^{(n)}(x_k)\frac{h^n}{n!}. \quad (3.33)$$

Для виразу (3.33) К. Рунге запропонував (згодом В. Кутта розвинув цю ідею методу) для різниці  $\Delta y(h) = y(x_{k+1}) - y(x_k)$  виконувати пошук лінійних апроксимаційних комбінацій такого вигляду:

$$\Delta y(h) = y(x_{k+1}) - y(x_k) = p_{r1}K_1(h) + p_{r2}K_2(h) + \dots + p_{rm}K_r(h), \quad (3.34)$$

де  $p_{rm}$  ( $m = 1, 2, \dots, r$ ) – деякі постійні коефіцієнти;  $r$  – порядок точності чисельного розв'язання диференціального рівняння, а  $K_r(h)$  – функції, які обчислюються за формулами:

$$\begin{cases} K_1(h) = f(x_k, y_k); \\ K_2(h) = f(x_k + \frac{h}{2}, y_k + \frac{h}{2}K_1); \\ K_3(h) = f(x_k + \frac{h}{2}, y_k + \frac{h}{2}K_2); \\ K_4(h) = f(x_k + h, y_k + hK_3). \end{cases} \quad (3.35)$$

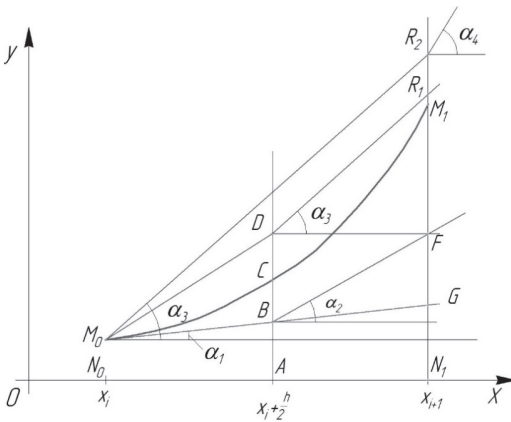
Тоді рівняння (3.34) на основі функцій (3.35) переписується у такому вигляді:

$$y_{k+1} = y_k + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4). \quad (3.36)$$

Ітераційне рівняння (3.36) є формулою Рунге-Кутта чисельного розв'язання ЗДР із похибкою обчислення четвертого порядку  $O(h^4)$ , що також отримала найбільше поширення у практичних розрахунках.

Підвищення порядку точності чисельних однокрокових методів Рунге-Кутта призводить до швидкого зростання трудомісткості обчислень, оскільки на одному кроці багаторазово доводиться обчислювати значення функції  $f(x, y(x))$  для різних значень аргументів. Тому на практиці переважно застосовують схему обчислення метода Рунге-Кутта четвертого роду (3.36).

Складові  $K_1, K_2, K_3, K_4$  схеми обчислення (3.36) методу Рунге-Кутта четвертого порядку має просту геометричну інтерпретацію (рис. 3.7). Нехай крива  $M_0CM_1$  являє собою розв'язок задачі Коші (3.16) ЗДР. Точка  $C$  цієї кривої лежить на прямій, перпендикулярній до осі  $Ox$  і ділить відрізок  $N_0N_1$  навпіл,  $B$  і  $G$  – точки перетину дотичної, проведеної до кривої в точці  $M_0$  із



ординатами  $AC$  і  $N_1M_1$ . Тоді число  $K_1$  із точністю до множника  $h$  є кутовим коефіцієнтом ( $\alpha_1$ ) дотичної в точці  $M_0$  до інтегральної кривої  $M_0CM_1$ , тобто  $K_1 = f(x_k, y_k)$ .

Точка  $B$  має координати  $x = x_i + h/2$  і  $y = y_i + K_1 h/2$ , відповідно, число  $K_2$  із точністю до множника  $h$  є кутовим коефіцієнтом ( $\alpha_2$ ) дотичної, проведеної до інтегральної кривої в точці  $B$  ( $BF$  – відрізок дотичної).

Рисунок 3.7 – Схема геометричної інтерпретації методу Рунге-Кутта

Через точку  $M_0$  проводиться пряма, паралельна прямій  $BF$ . Точка  $D$  має координати  $x = x_i + h/2$ ,  $y = y_i + K_1/2$  і число  $K_3$  із точністю до множника  $h$  є кутовим коефіцієнтом ( $\alpha_3$ ) дотичної, проведеної до інтегральної кривої в точці  $D$ , ( $DR_1$  – відрізок цієї дотичної). Тоді, через точку  $M_0$  проводиться пряма  $DR_1$ , яка перетинає продовження  $M_1N_1$  в точці  $R_2(x_i+h, y_i+K_3)$ . Отже,  $K_4$  із точністю до множника  $h$  є кутовим коефіцієнтом ( $\alpha_4$ ) дотичної, проведеної до інтегральної кривої в точці  $R_2$ .

Алгоритм методу Рунге-Кутта із автоматичним вибором кроку зображено на рисунку 3.8.

Для оцінення загальної похибки чисельного розв'язання задачі Коші для ЗДР методом Рунге-Кутта, і вибору кроку інтегрування на кожному ітераційному кроці обчислення було запропоновано метод Колатца, а саме: якщо в процесі розрахунків значення  $R = \left| \frac{K_2 - K_3}{K_1 - K_2} \right|$  перевищує декілька сотих, тоді крок необхідно зменшити (див. рис. 3.8).

**Приклад 3.3.** Розв'язати чисельним методом Рунге-Кутта четвертого порядку задачу Коші для ЗДР першого порядку:

$$\frac{dy}{dx} = y + x; \quad x \in [0; 1,0], \quad (3.37)$$

яка задовольняє початкову умову за  $x_0 = 0$  та  $y_0 = 1,0$ .

*Розв'язання:*

Для розв'язання цієї задачі відрізок  $[0; 1,0]$  можна розділити на десять частин точками  $x_0 = 0; 0,1; 0,2; \dots, 1,0$ . Відповідно  $h = 0,1$ . Значення  $y_1, y_2, \dots, y_n$  будуть визначатись методом Рунге-Кутта четвертого порядку за формулою (3.36), тоді:

$$\left\{ \begin{array}{l} K_1^{(0)} = y_0 + x_0; \\ K_2^{(0)} = y_0 + \frac{h}{2} K_1^{(0)} + x_0 + \frac{h}{2}; \\ K_3^{(0)} = x_0 + \frac{h}{2} + y_0 + \frac{h}{2} K_2^{(0)}; \\ K_4^{(0)} = x_0 + h + y_0 + h K_3^{(0)}; \\ y_1 = y_0 + \frac{h}{6} (K_1^{(0)} + 2K_2^{(0)} + 2K_3^{(0)} + K_4^{(0)}); \end{array} \right. \Rightarrow \left\{ \begin{array}{l} K_1^{(0)} = 1 + 0 = 1,0; \\ K_2^{(0)} = 1,0 + \frac{0,1}{2} \cdot 1,0 + 0 + \frac{0,1}{2} = 1,10000; \\ K_3^{(0)} = 0 + \frac{0,1}{2} + 1,0 + \frac{0,1}{2} \cdot 1,1 = 1,10500; \\ K_4^{(0)} = 0 + 0,1 + 1,0 + 0,1 \cdot 1,105 = 1,21050; \\ y_1 = 1 + \frac{0,1}{6} \left( 1 + 2 \cdot 1,1 + \right. \\ \left. + 2 \cdot 1,105 + 1,2105 \right) = 1,11034; \end{array} \right.$$

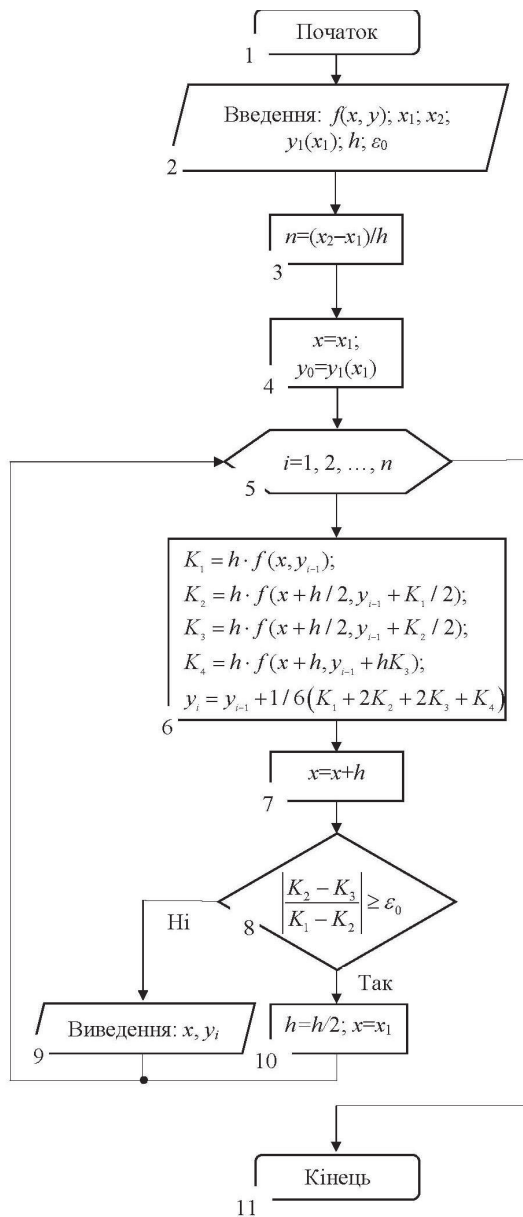


Рисунок 3.8 – Схема алгоритму методу Рунге-Кутта четвертого порядку

$$\begin{cases} K_1^{(1)} = y_1 + x_1; \\ K_2^{(1)} = y_1 + \frac{h}{2}K_1^{(1)} + x_1 + \frac{h}{2}; \\ K_3^{(1)} = x_1 + \frac{h}{2} + y_1 + \frac{h}{2}K_2^{(1)}; \\ K_4^{(1)} = x_1 + h + y_1 + hK_3^{(1)}; \\ y_2 = y_1 + \frac{h}{6}(K_1^{(1)} + 2K_2^{(1)} + 2K_3^{(1)} + K_4^{(1)}); \end{cases}$$

$$\begin{cases} K_1^{(1)} = 1,11034 + 0,1 = 1,2100; \\ K_2^{(1)} = 1,11034 + \frac{0,1}{2}1,210 + 0,1 + \frac{0,1}{2} = 1,3210; \\ K_3^{(1)} = 0,1 + \frac{0,1}{2} + 1,11034 + \frac{0,1}{2}1,321 = 1,3270; \\ K_4^{(1)} = 0,1 + 0,1 + 1,11034 + 0,1 \cdot 1,327 = 1,4430; \\ y_2 = 1,11034 + \frac{0,1}{6}(1,210 + 2 \cdot 1,321 + \\ + 2 \cdot 1,327 + 1,443) = 1,2428; \end{cases}$$

.....

Також для порівняльного аналізу точності розв'язання методом Рунге-Кутта четвертого порядку було отримано аналітичний розв'язок ЗДР, а саме:

$$\tilde{y} = 2e^x - x - 1. \tag{3.38}$$

У процесі розв'язання було складено таблицю 3.3 результатів обчислення.

Із отриманих результатів, наведених у таблиці 3.3, видно високий ступінь точності визначенні розв'язку, зокрема, в точках  $x_5 = 0,5$  і  $x_{10} = 1,0$ :

– відносна похибка шляхом порівняння між значеннями аналітичного розв'язку і методом Рунге-Кутта четвертого порядку

$$\Delta_{x_5} = \left| \frac{\tilde{y}_{x_5} - y_{x_5}}{\tilde{y}_{x_5}} \right| \cdot 100\% = \left| \frac{1,79744 - 1,79743}{1,79744} \right| \cdot 100\% = 5,56 \cdot 10^{-4}\%,$$

$$\Delta_{x_{10}} = \left| \frac{\tilde{y}_{x_{10}} - y_{x_{10}}}{\tilde{y}_{x_{10}}} \right| \cdot 100\% = \left| \frac{3,43656 - 3,43655}{3,43656} \right| \cdot 100\% = 2,90 \cdot 10^{-4}\%;$$

– значення показника похибки за методом Колатца не перевищує декілька сотих

$$R_{x_5} = \frac{\left| \frac{K_2^{x_5} - K_3^{x_5}}{K_1^{x_5} - K_2^{x_5}} \right|}{\left| \frac{2,1328240 - 2,140283}{1,9836422 - 2,132824} \right|} = 0,005,$$

$$R_{x_{10}} = \frac{\left| \frac{K_2^{x_{10}} - K_3^{x_{10}}}{K_1^{x_{10}} - K_2^{x_{10}}} \right|}{\left| \frac{4,1651520 - 4,177450}{3,9191925 - 4,165152} \right|} = 0,005.$$

Також можна відмітити відносно високу точність методу Рунге-Кутта четвертого порядку порівняно із звичайним і уточненим методами Ейлера під час розв'язання задачі Коші ЗДР.

Таблиця 3.3 – Результати обчислення розв'язку диференціального рівняння

$n$	$x_n$	$K_1^{(n)}$	$K_2^{(n)}$	$K_3^{(n)}$	$K_4^{(n)}$	$y_n(x_n)$	$\tilde{y}_n(x_n)$
0	0,0	0,0000000	0,0000000	0,0000000	0,0000000	1,000000	1,000000
1	0,1	1,0000000	1,1000000	1,1050000	1,2105000	1,11034	1,11034
2	0,2	1,2100000	1,3210000	1,3270000	1,4430000	1,24280	1,24281
3	0,3	1,4428000	1,564940	1,571047	1,699905	1,39971	1,39972
4	0,4	1,6997113	1,834697	1,841446	1,983856	1,58364	1,58365
5	0,5	1,9836422	2,132824	2,140283	2,297671	1,79743	1,79744
6	0,6	2,2974343	2,462306	2,470550	2,644489	2,04423	2,04424
7	0,7	2,6442283	2,826440	2,835550	3,027783	2,32750	2,32751
8	0,8	3,0274948	3,228870	3,238938	3,451389	2,65107	2,65108
9	0,9	3,4510698	3,673623	3,684751	3,919545	3,01919	3,01921
10	1,0	3,9191925	4,165152	4,177450	4,436937	3,43655	3,43656

Розглянемо реалізацію методу Рунге-Кутта мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
#введення кроку інтегрування
h=float(input())
#введення початкового значення аргументу інтегрування (загального проміжку
інтегрування)
a=int(input()); x_mtr=np.array(a)
#введення початкового значення аргументу функції диф. рівняння
y_0=int(input()); y_mtr=np.array(y_0)
#введення кінцевого значення загального проміжку інтегрування
```

```

b=int(input())
#введення допустимої похибки обчислення за методом Колатца
e_0=float(input())
#задання функції диф. рівняння
def f(x,y):
    return x+y
#обчислення ітераційних формул
x_mas=[a]; y_mas=[y_0]
x_mtr=np.array(a)
x=a; y=y_0
while x<b:
    #ітераційна формула методу Рунге-Кутта
    K1=f(x,y); K2=f(x+(h/2),y+(h/2*K1))
    K3=f(x+(h/2),y+(h/2*K2))
    K4=f(x+(h/2),y+(h*K3))
    y=y+(h/6*(K1+(2*K2)+(2*K3)+K4))
    x=x+h
    x_mas.append(x)
    y_mas.append(y)
    x_mtr=np.append(x_mtr, x)
#визначення похибки обчислення за методом Колатца і коригування кроку
# інтегрування
    if abs(K2-K3)/abs(K1-K2)>=e_0:
        h=h/2; x=a; del x_mas; del x_mtr; del y_mas
        x_mas=[a]; x_mtr=np.array(a); y_mas=[y_0]; y=y_0
#будуємо графіки рішень диф. рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
plt.plot(x_mas, y_mas)
plt.plot(x_mtr, (2*(e**x_mtr))-x_mtr-1)
plt.legend(['Метод Рунге-Кутта', 'f(x)=2*exp(x)-x-1'],loc=1)
plt.grid(True)
plt.xlim([0, 1])
plt.ylim([1, 4])
plt.show()

```

Усі методи Рунге-Кутта узагальнюються на системи ЗДР. Нехай дано систему диференціальних рівнянь:

$$\frac{d\bar{y}}{dx} = \bar{f}(x, \bar{y}); \quad x \in [a; b], \quad (3.39)$$

із початковими умовами –  $\bar{y}(a) = \bar{y}_0$ , де  $\bar{y} = (y_1, y_2, \dots, y_n)^T$ ;  
 $\bar{f} = (f_1, f_2, \dots, f_n)^T$ ;  $\bar{y}_0 = (y_{10}, y_{20}, \dots, y_{n0})^T$ .

Обирається  $h > 0$  і будується рівномірна сітка:

$$\bar{\omega}_n = \left\{ x_n \mid x_n = a + nh; \quad n = \overline{0, N}; \quad N = \frac{b-a}{h} \right\}.$$

Ставиться задача визначення значення наближеного розв'язку  $\bar{y}_n = \bar{y}(x_n)$  ( $n = \overline{1, N}$ ) за формулами  $\bar{y}_{n+1} = \bar{y}_n + \Delta \bar{y}_n$  ( $n = \overline{1, N-1}$ ), де  $\Delta \bar{y}_n$

обчислюється, наприклад, за такою формулою:

$$\left\{ \begin{array}{l} \bar{y}_{n+1} = \bar{y}_n + \frac{h}{6} (\bar{K}_1^{(n)} + 2\bar{K}_2^{(n)} + 2\bar{K}_3^{(n)} + \bar{K}_4^{(n)}); \\ \bar{K}_1^{(n)} = \bar{f}(x_n, \bar{y}_n); \\ \bar{K}_2^{(n)} = \bar{f}\left(x_n + \frac{h}{2}, \bar{y}_n + \frac{h}{2} \bar{K}_1^{(n)}\right); \\ \bar{K}_3^{(n)} = \bar{f}\left(x_n + \frac{h}{2}, \bar{y}_n + \frac{h}{2} \bar{K}_2^{(n)}\right); \\ \bar{K}_4^{(n)} = \bar{f}(x_n + h, \bar{y}_n + h \bar{K}_3^{(n)}); \\ n = 0, \bar{N} - 1. \end{array} \right. \quad (3.40)$$

Отже, знаючи  $\bar{y}_0$ , за формулами (3.40) обчислюється  $\bar{y}_1$ . Беручи  $(x_1, \bar{y}_1)$  за вихідні дані і повторюючи цей самий процес, визначається  $\bar{y}_2$  і т. д. Аналогічно будь-яка обчислювальна схема метода Рунге-Кутта для одного рівняння переноситься на систему рівнянь вигляду (3.39). У прикладі 3.4 розглядається застосування чисельних методів Рунге-Кутта для ЗДР вищих порядків, що також додатково дозволяє визначити особливості чисельного розв'язання систем ЗДР.

Для оцінення загальної похибки чисельного розв'язання задачі Коші для систем ЗДР може бути використаний метод Рунге на основі залежності (3.26), яка порівнює результати обчислення значень  $\bar{y}_n$  і  $\tilde{y}_n$  в точці  $x_n$  із різними кроками  $\bar{h}_n$  та  $\tilde{h}_n$ , відповідно:

$$R_n^h = (\tilde{y}_n - \bar{y}_n) / (1 - 2^{-p}), \quad (3.41)$$

де  $p = 4$  – порядок апроксимації використовуваної різницевої схеми методом Рунге-Кутта четвертого порядку.

Алгоритм методу Рунге-Кутта для розв'язання систем ЗДР із визначенням похибки обчислення і автоматичним вибором кроку подано на рисунку 3.9.

**Приклад 3.4.** Методом Рунге-Кутта отримати чисельний розв'язок рівняння коливальних маятника в середовищі, яке створює опір руху:

$$\frac{d^2\theta}{dt^2} + 0,2 \frac{d\theta}{dt} + 10 \sin \theta = 0; \quad t \in [0, 1,0] \quad (3.42)$$

за початкових умов  $\theta(0) = 0,3$ ;  $\frac{d\theta}{dt}(0) = 0$  і похибки зрізу  $R^h = 0,001$ .



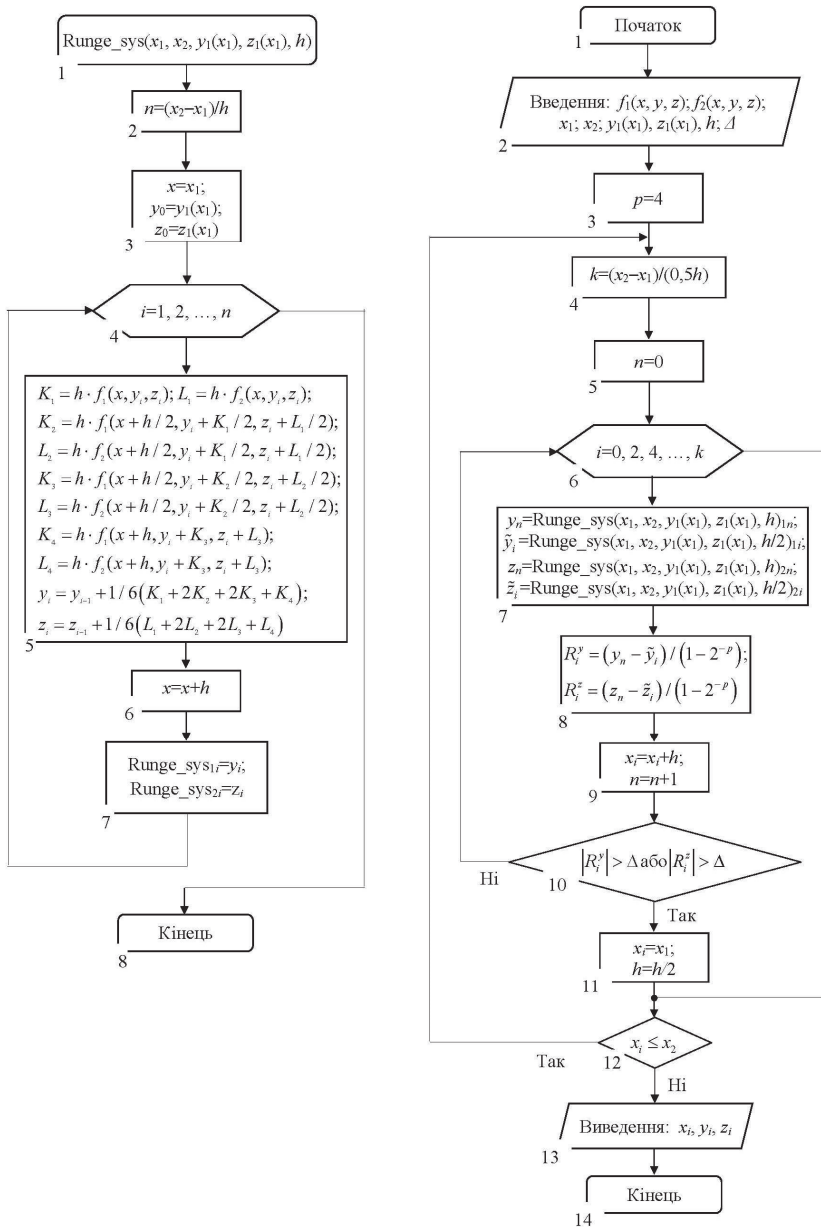


Рисунок 3.9 – Схема алгоритму метода Рунге-Кутта четвертого порядку для систем ЗДР

У рівнянні (3.42)  $\theta(t)$  – функція кута відхилення маятника, яка залежить від часу  $t$ .

*Розв'язання:*

Для розв'язання цієї задачі відрізок  $[0; 1,0]$  можна розділити на десять частин точками  $x_0 = 0; 0,1; 0,2; \dots, 1,0$ . Відповідно  $h = 0,1$ . Для розв'язання задачі вводиться заміна  $z = \frac{d\theta}{dt}$ . Тоді рівняння (3.42) із початковими умовами можуть бути подані у вигляді системи ЗДР:

$$\begin{cases} \frac{d\theta}{dt} = z; \\ \frac{dz}{dt} = -0,2z - 10\sin\theta; \\ \theta(0) = 0,3; \\ z(0) = 0. \end{cases} \quad (3.43)$$

Також для порівняльного аналізу точності розв'язання методом Рунге-Кутта четвертого порядку було отримано аналітичний розв'язок ЗДР типу (3.42) із припущення, що  $\theta \rightarrow 0$ , а саме:

$$\tilde{\theta}(t) = 0,3e^{-0,1t} [\cos(3,1607 \cdot t) + 0,03164 \cdot \sin(3,1607 \cdot t)]. \quad (3.44)$$

Значення  $\theta_1, \theta_2, \dots, \theta_n$  і  $z_1, z_2, \dots, z_n$  будуть визначатись методом Рунге-Кутта четвертого порядку за формулою (3.40), а саме:

$$\begin{cases} K_{1z}^{(0)} = f(\theta_0, z_0, t_0) = -0,2z_0 - 10\sin\theta_0; \\ K_{2z}^{(0)} = f\left(\theta_0 + \frac{h}{2}K_{1\theta}^{(0)}, z_0 + \frac{h}{2}K_{1z}^{(0)}, t_0 + \frac{h}{2}\right) = -0,2\left(z_0 + \frac{h}{2}K_{1z}^{(0)}\right) - 10\sin\left(\theta_0 + \frac{h}{2}K_{1\theta}^{(0)}\right); \\ K_{3z}^{(0)} = f\left(\theta_0 + \frac{h}{2}K_{2\theta}^{(0)}, z_0 + \frac{h}{2}K_{2z}^{(0)}, t_0 + \frac{h}{2}\right) = -0,2\left(z_0 + \frac{h}{2}K_{2z}^{(0)}\right) - 10\sin\left(\theta_0 + \frac{h}{2}K_{2\theta}^{(0)}\right); \\ K_{4z}^{(0)} = f\left(\theta_0 + hK_{3\theta}^{(0)}, z_0 + hK_{3z}^{(0)}, t_0 + h\right) = -0,2\left(z_0 + hK_{3z}^{(0)}\right) - 10\sin\left(\theta_0 + hK_{3\theta}^{(0)}\right); \\ z_1 = z_0 + \frac{h}{6}\left(K_{1z}^{(0)} + 2K_{2z}^{(0)} + 2K_{3z}^{(0)} + K_{4z}^{(0)}\right); \end{cases}$$

$$\left\{ \begin{aligned} K_{1z}^{(0)} &= -0,2 \cdot 0 - 10 \cdot \sin 0,3 = -2,95520; \\ K_{2z}^{(0)} &= -0,2 \cdot \left[ 0 + \frac{0,1}{2}(-2,9552) \right] - 10 \cdot \sin \left[ 0,3 + \frac{0,1}{2} \cdot 0 \right] = -2,92565; \\ K_{3z}^{(0)} &= -0,2 \cdot \left[ 0 + \frac{0,1}{2}(-2,92565) \right] - 10 \cdot \sin \left[ 0,3 + \frac{0,1}{2}(-0,14776) \right] = -2,85529; \\ K_{4z}^{(0)} &= -0,2 \cdot \left[ 0 + 0,1(-2,85529) \right] - 10 \cdot \sin \left[ 0,3 + 0,1(-0,28553) \right] = -2,75804; \\ z_1 &= 0 - \frac{0,1}{6}(2,95520 + 2 \cdot 2,92565 + 2 \cdot 2,85529 + 2,75804) = -0,28792; \end{aligned} \right.$$

$$\left\{ \begin{aligned} K_{1\theta}^{(0)} &= \psi(\theta_0, z_0, t_0) = z_0; \\ K_{2\theta}^{(0)} &= \psi \left[ \theta_0 + \frac{h}{2} K_{1\theta}^{(0)}, z_0 + \frac{h}{2} K_{1z}^{(0)}, t_0 + \frac{h}{2} \right] = z_0 + \frac{h}{2} K_{1z}^{(0)}; \\ K_{3\theta}^{(0)} &= \psi \left[ \theta_0 + \frac{h}{2} K_{2\theta}^{(0)}, z_0 + \frac{h}{2} K_{2z}^{(0)}, t_0 + \frac{h}{2} \right] = z_0 + \frac{h}{2} K_{2z}^{(0)}; \\ K_{4\theta}^{(0)} &= \psi \left[ \theta_0 + h K_{3\theta}^{(0)}, z_0 + h K_{3z}^{(0)}, t_0 + h \right] = z_0 + h K_{3z}^{(0)}; \\ \theta_1 &= \theta_0 + \frac{h}{6} (K_{1\theta}^{(0)} + 2K_{2\theta}^{(0)} + 2K_{3\theta}^{(0)} + K_{4\theta}^{(0)}); \end{aligned} \right.$$

$$\left\{ \begin{aligned} K_{1\theta}^{(0)} &= z_0 = 0,00000; \\ K_{2\theta}^{(0)} &= 0 + \frac{0,1}{2}(-2,95520) = -0,14776; \\ K_{3\theta}^{(0)} &= 0 + \frac{0,1}{2}(-2,92565) = -0,14628; \\ K_{4\theta}^{(0)} &= 0 + 0,1 \cdot (-2,85529) = -0,28553; \\ \theta_1 &= 0,3 - \frac{0,1}{6}(0 + 2 \cdot 0,14776 + 2 \cdot 0,14628 + 0,28553) = 0,28554; \end{aligned} \right.$$

.....

У процесі розв'язання було складено таблицю 3.4 результатів обчислення чисельним методом Рунге-Кутта четвертого порядку і аналітичним розв'язком (3.43), які графічно інтерпретуються на рисунку 3.10. Також для визначення похибки обчислення методом Рунге (3.41) було визначено значення  $\bar{\theta}_n(x_n)$  із кроком  $h/2 = 0,05$ .

Із таблиці 3.4 видно, що значення кута відхилення  $\theta_n(t_n)$  маятника в середовищі, яке чинить опір, з часом убуває, що відповідає фізиці аналізованого процесу.

Таблиця 3.4 – Результати обчислення розв’язку диференціального рівняння

$n$	$t_n, \text{с}$	$\theta_n(t_n), \text{град}$	$\bar{\theta}_n(x_n), \text{град}$	$\frac{d\theta}{dt}(t_n), \frac{\text{град}}{\text{с}}$	$\tilde{\theta}(t_n), \text{град}$
0	0,0	0,30000	0,30000	0,00000	0,30000
1	0,1	0,28544	0,28544	-0,28792	0,28522
2	0,2	0,24352	0,24351	-0,54295	0,24274
3	0,3	0,17876	0,17874	-0,74113	0,17726
4	0,4	0,09783	0,09780	-0,86376	0,09567
5	0,5	0,00892	0,00889	-0,89960	0,00630
6	0,6	-0,07910	-0,07914	-0,84648	-0,08191
7	0,7	-0,15763	-0,15767	-0,71161	-0,16034
8	0,8	-0,21919	-0,21923	-0,51039	-0,22147
9	0,9	-0,25819	-0,25821	-0,26420	-0,25964
10	1,0	-0,27135	-0,27136	0,00225	-0,27157

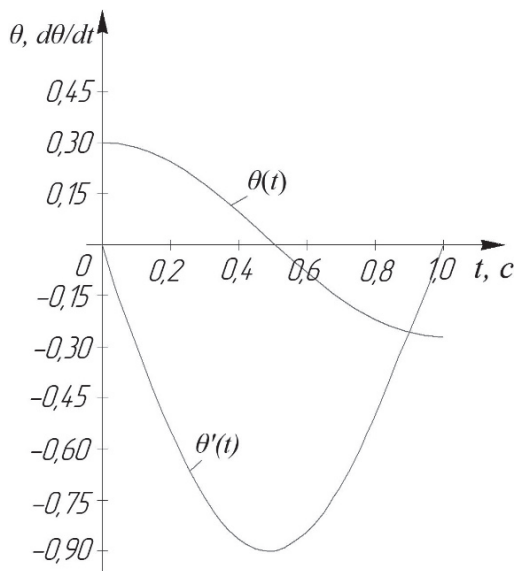


Рисунок 3.10 – Діаграма результатів чисельного розв’язку задачі Коші ЗДР вищого порядку методом Рунге-Кутта четвертого порядку

Із отриманих результатів, наведених у таблиці 3.4 видно високу ступінь точності визначенні розв'язку, зокрема, в точках  $t_3 = 0,3$  с і  $t_{10} = 1,0$  с:

– відносна похибка шляхом порівняння між значеннями аналітичного розв'язку і методом Рунге-Кутти четвертого порядку

$$\varepsilon_{t_3} = \left| \frac{\tilde{\theta}_{t_3} - \theta_{t_3}}{\tilde{\theta}_{t_3}} \right| \cdot 100\% = \left| \frac{0,17876 - 0,17726}{0,17726} \right| \cdot 100\% = 0,850\%,$$

$$\varepsilon_{t_{10}} = \left| \frac{\tilde{\theta}_{t_{10}} - \theta_{t_{10}}}{\tilde{\theta}_{t_{10}}} \right| \cdot 100\% = \left| \frac{-0,27157 - (-0,27135)}{-0,27157} \right| \cdot 100\% = 0,081\%;$$

– похибка зрізу методом Рунге (див. 3.41)

$$R_{x_3}^h = (\theta_{x_3} - \bar{\theta}_{x_3}) / (1 - 2^{-p}) = (0,17876 - 0,17874) / (1 - 2^{-4}) = 2,13 \cdot 10^{-5},$$

$$R_{x_{10}}^h = (\theta_{x_{10}} - \bar{\theta}_{x_{10}}) / (1 - 2^{-p}) = (-0,27135 - (-0,27136)) / (1 - 2^{-4}) = 1,07 \cdot 10^{-5}.$$

Розглянемо реалізацію методу Рунге-Кутта четвертого порядку для систем диференціальних рівнянь мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
#введення кроку інтегрування
h_0=float(input())
#введення початкового значення аргументу інтегрування (загального проміжку
# інтегрування) t0
a=int(input()); t_mtr=np.array(a)
#введення початкового значення аргументу функції диф. рівняння q0
q_0=float(input()); q_mtr=np.array(q_0)
#введення початкового значення аргументу функції диф. рівняння z0
z_0=float(input()); z_mtr=np.array(z_0)
#введення кінцевого значення загального проміжку інтегрування
b=int(input())
#введення допустимої похибки обчислення
e_0=float(input())
#задання функцій системи диф. рівняння
def f1(z):
    return z
def f2(q,z):
    return (-0.2*z)-(10*math.sin(q))
#обчислення ітераційних формул
def runge(a,b,q_0,z_0,h_0):
    t=a; q=q_0; z=z_0
    t_mas=[t]; q_mas=[q]; z_mas=[z]
    t_mtr=np.array(t)
    q_mtr=np.array(q_0)
    z_mtr=np.array(z_0)
    h=h_0
    while t<=b:
        #ітераційні формули методу Рунге-Кутта
        k1_z=f2(q,z); k1_q=f1(z)
        k2_z=f2(q+(h/2*k1_q),z+(h/2*k1_z)); k2_q=f1(z+(h/2*k1_z))
        k3_z=f2(q+(h/2*k2_q),z+(h/2*k2_z)); k3_q=f1(z+(h/2*k2_z))
        k4_z=f2(q+(h*k3_q),z+(h*k3_z)); k4_q=f1(z+(h*k3_z))
```

```

z=z+(h/6*(k1_z+(2*k2_z)+(2*k3_z)+k4_z))
q=q+(h/6*(k1_q+(2*k2_q)+(2*k3_q)+k4_q))
t=t+h
q_mas.append(q)
z_mas.append(z)
t_mas.append(t)
t_mtr=np.append(t_mtr, t)
return t_mas, q_mas, z_mas, t_mtr
#визначення похибки обчислень методом Рунге і коригування кроку
# інтегрування
p=4; x=a
while x<=b:
    n=0
    for k in range(0,len(runge(a,b,q_0,z_0,h_0/2)[1]),2):
        Rh_q=(runge(a,b,q_0,z_0,h_0)[1][n]-
                runge(a,b,q_0,z_0,h_0/2)[1][k])/(1-(1/2**p))
        Rh_z=(runge(a,b,q_0,z_0,h_0)[2][n]-
                runge(a,b,q_0,z_0,h_0/2)[2][k])/(1-(1/2**p))
        #print(Rh_q)
        #print(Rh_z)
        n=n+1
        x=x+h_0
        if abs(Rh_q>e_0) and abs(Rh_z>e_0):
            h_0=h_0/2; x=a
            break
#будуємо графіки рішень диф. рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('t',fontsize=15, color='blue')
plt.ylabel('psi, dPsi(t)/dt',fontsize=15, color='blue')
plt.plot(runge(a,b,q_0,z_0,h_0)[0], runge(a,b,q_0,z_0,h_0)[1])
plt.plot(runge(a,b,q_0,z_0,h_0)[0], runge(a,b,q_0,z_0,h_0)[2])
plt.plot(runge(a,b,q_0,z_0,h_0)[3], 0.3*(e**(-0.1*runge(a,b,q_0,z_0,h_0)[3]))*
        (np.cos(3.1607*runge(a,b,q_0,z_0,h_0)[3])+
        0.03164*np.sin(3.1607*runge(a,b,q_0,z_0,h_0)[3])))
plt.legend(['Метод Рунге-Кутта для Psi(t)', 'Метод Рунге-Кутта для
dPsi(t)/dt', 'Точний розв'язок'], loc=1)
plt.grid(True)
plt.xlim([0, 1])
plt.ylim([-0.95, 0.45])
plt.show().

```

Потрібно відзначити у таблиці 3.4 на інших інтервалах часу  $t_n$  значну розбіжність між результатами чисельного і аналітичного розрахунків, що саме обумовлено отримання аналітичного розрахунку (3.43) із припущення, що  $\theta \rightarrow 0$ .

Вищенаведені розглянуті чисельні однокрокові методи розв'язання задачі Коші для ЗДР можуть бути узагальнені такими характеристиками:

1) для отримання інформації в новій точці необхідні дані тільки однієї попередньої точки;

2) в основі усіх однокрокових методів покладено розкладення функції в ряд Тейлора, в якому зберігаються члени, що містять крок  $h$  в степені до  $n$  включно. Ціле число називається порядком методу, а похибка на кроці має порядок  $m+1$ ;

3) однокрокові методи не потребують обчислення похідних, тому що обчислюється тільки функція, але може бути необхідне її значення в декількох проміжкових точках;

4) існує можливість зміни величини кроку обчислення.

## Метод Адамса

В однокрокових методах розв'язання задачі Коші для одного ЗДР:

$$\frac{du}{dx} = f(x, u); \quad x > x_0; \quad u(x_0) = u_0, \quad (3.45)$$

яка вважається коректно поставленою, значення  $u_{n+1}$  залежить тільки від інформації про розв'язання в попередній точці сітки  $x_n$  ( $n = 0, 1, 2, \dots$ ).

Для підвищення точності може бути використана інформація про розв'язання в декількох попередніх точках сітки  $x_n, x_{n-1}, x_{n-2}, \dots$ . Більше того, доцільно використовувати інформацію із забіганням уперед за точку  $x_{n+1}$ .

У багатокрокових, як і однокрокових, методах доцільно використовувати постійний крок такої розрахункової сітки:

$$\overline{\omega}_h = \left\{ x_m \mid x_m = a + mh; m = \overline{0, N}; N = \frac{b-a}{h} \right\}. \quad (3.46)$$

Вводяться сіткові функції  $y_n = y(x_n)$ ,  $f_n = f(x_n, y_n)$ ,  $u_n = u(x_n)$ , визначені на сітці  $\overline{\omega}_h$  (3.46). Лінійним  $n$ -кроковим різницевим методом називається система різницевих рівнянь:

$$\frac{a_0 y_n + a_1 y_{n-1} + a_2 y_{n-2} + \dots + a_m y_{n-m}}{h} = b_0 f_n + b_1 f_{n-1} + \dots + b_m f_{n-m}, \quad (3.47)$$

або в компактній формі  $\sum_{k=0}^m \frac{a_k y_{n-k}}{h} = \sum_{k=0}^m b_k f_{n-k}$ , яка визначена для  $n = m, (m+1), \dots$ , де  $a_k, b_k$  – числові коефіцієнти, що не залежать від  $n$  ( $k = 0, 1, \dots, m$ ), причому  $a_0 \neq 0$ . Рівняння (3.47) – це рекурентне співвідношення із визначення нового значення  $y_n = y(x_n)$  через знайдені раніше значення  $y_{n-1}, y_{n-2}, \dots, y_{n-m}$ .

Частинним випадком багатокрокових методів (3.47), є наявність умови коли похідна  $u'(x)$  апроксимується тільки за двома точками  $x_n$  та  $x_{n-1}$ , тобто коефіцієнти  $a_k$  набувають значень:  $a_0 = -a_1 = 1$ ;  $a_k = 0$  ( $k = 2, 3, \dots, m$ ). У такому випадку має місце використання методу Адамса, який загалом має вигляд:

$$\frac{y_n - y_{n-1}}{h} = \sum_{k=0}^m b_k f_{n-k}. \quad (3.48)$$

де за  $b_0 = 0$  методи називаються явними, для яких порядок апроксимації дорівнює  $m$ , а за  $b_0 \neq 0$  – неявними, для яких порядок апроксимації дорівнює  $m+1$ .

Для явних  $m$ -крокових методів Адамса для кожного значення  $m$  визначено коефіцієнти методу найвищого порядку апроксимації рівнянь (3.48). Зокрема, за  $m=4$  буде отримано явний метод четвертого порядку апроксимації:

$$\frac{y_n - y_{n-1}}{h} = \frac{1}{24}(55f_{n-1} - 59f_{n-2} + 37f_{n-3} - 9f_{n-4}) \quad (3.49)$$

із локальною похибкою  $\rho = \frac{251}{720}h^5 u^{(5)}(x)$ .

Формула (3.49) відома як метод Адамса-Башфорта четвертого порядку. Коефіцієнти найвищого порядку апроксимації рівняння (3.48) цього методу можуть бути отримані й вище, використовуючи інформацію про більшу кількість попередніх точок, а це також дозволяє отримати метод Адамса-Башфорта високого порядку. Проте точність обчислень зі збільшенням порядку зростає нелінійно (чим більше відстань попередньої точки від поточної, тим слабше вона впливає на точність).

Багатокрокові методи породжують проблему, яка виникає у разі використання однокрокових методів. Оскільки в багатокрокових методах використовується інформація про раніше отримані точки, то, на відміну від однокрокових методів, вони не мають властивості «самостартування». Тому, перш ніж застосовувати багатокроковий метод, доводиться обчислювати вихідні дані за допомогою однокрокових методів, наприклад методи Ейлера або Рунге-Кутта.

У неявних  $m$ -крокових методів Адамса для кожного значення  $m$  визначено коефіцієнти методу найвищого порядку апроксимації рівнянь (3.48), які дорівнюють  $(m+1)$ . В такому разі виникає наступний клас неявних обчислювальних методів, які називаються методами Адамса-Моултона другого, третього й четвертого порядків, відповідно. Зокрема, для  $m = 3$  буде отримано метод четвертого порядку апроксимації  $O(h^4)$ :

$$\frac{y_n - y_{n-1}}{h} = \frac{1}{24}(9f_n + 19f_{n-1} - 5f_{n-2} + f_{n-3}) \quad (3.50)$$

із локальною похибкою  $\rho = -\frac{19}{720}h^5 u^{(5)}(x)$ .

У формулі (3.50) значення  $f_n$  невідомо, оскільки для обчислення  $f(x_n, y_n) = f_n$  необхідно мати три значення  $y_n$ , які є невідомими. Відповідно методи Адамса-Моултона визначають значення  $y_n$  неявно, хоча методи Адамса-Башфорта називаються явними, оскільки процес визначення значення  $y_n$  не потребує розв'язування жодних рівнянь. Тому на практиці під час розв'язування ЗДР використовується спільна явна і неявна формули, що призводить до застосування методу «прогнозування і корекції» (об'єднання методів Адамса четвертого порядку):



$$\begin{cases} y_n^* = y_{n-1} + \frac{h}{24}(55f_{n-1} - 59f_{n-2} + 37f_{n-3} - 9f_{n-4}); \\ f_n^* = f(x_n, y_n^*); \\ y_n = y_{n-1} + \frac{h}{24}(9f_n^* + 19f_{n-1} - 5f_{n-2} + f_{n-3}). \end{cases} \quad (3.51)$$

Загалом метод «прогнозу та корекції» є явним. Спочатку за формулою Адамса-Башфорта (3.49) обчислюється значення  $y_n^*$  в (3.51), що є «прогнозом» для  $y_n$ . Після чого значення  $y_n^*$  використовується для обчислення наближеного значення  $f_n^*$  в (3.51), яке також використовується у формулі Адамса-Моултона (3.50). Таким чином, формула Адамса-Моултона «коригує» наближення, яке визначає формула Адамса-Башфорта (3.49). Оскільки помилки зрізу ряду для формул прогнозування  $y_n^*$  Адамса-Башфорта (3.49) дорівнює  $\rho = \frac{251}{720}h^5 u^{(5)}(x)$ , а для коригування  $y_n$  Адамса-

Моултона (3.50) дорівнює  $\rho = -\frac{19}{720}h^5 u^{(5)}(x)$ , це дозволяє додатково зменшити похибку обчислення на 3%. Рівняння корекції точніші, ніж формули прогнозування, що дозволяють загалом підвищити точність обчислення ЗДР, незважаючи на виникнення додаткових обчислень. Також для досягнення найбільшої точності обчислення, процес корекції в методах «прогнозування та корекції» може бути повторено декілька разів на тому самому ітераційному кроці для отримання значення із певною заданою точністю за допомогою абсолютної похибки  $|y_n^* - y_n^{(i)}| \leq \Delta$ , де  $y_n^{(i)}$  – поточне значення розв'язку методом Адамса-Моултона (3.50) на певному ітераційному кроці загального методу методах «прогнозування та корекції».

Алгоритм методу Адамса подано на рисунку 3.11.

Метод Адамса («прогнозування та корекції») успішно узагальнюється на задачі Коші систем ЗДР. Нехай дано систему диференціальних рівнянь:

$$\frac{d\bar{y}}{dx} = \bar{f}(x, \bar{y}); \quad x \in [a; b], \quad (3.52)$$

із початковими умовами –  $\bar{y}(a) = \bar{y}_0$ , де  $\bar{y} = (y_1, y_2, \dots, y_n)^T$ ,  $\bar{f} = (f_1, f_2, \dots, f_n)^T$ ,  $\bar{y}_0 = (y_{10}, y_{20}, \dots, y_{n0})^T$ .

Вибирається  $h > 0$  і будується рівномірна сітка:

$$\bar{\omega}_n = \left\{ x_n \mid x_n = a + nh; \quad n = 0, N; \quad N = \frac{b-a}{h} \right\}.$$

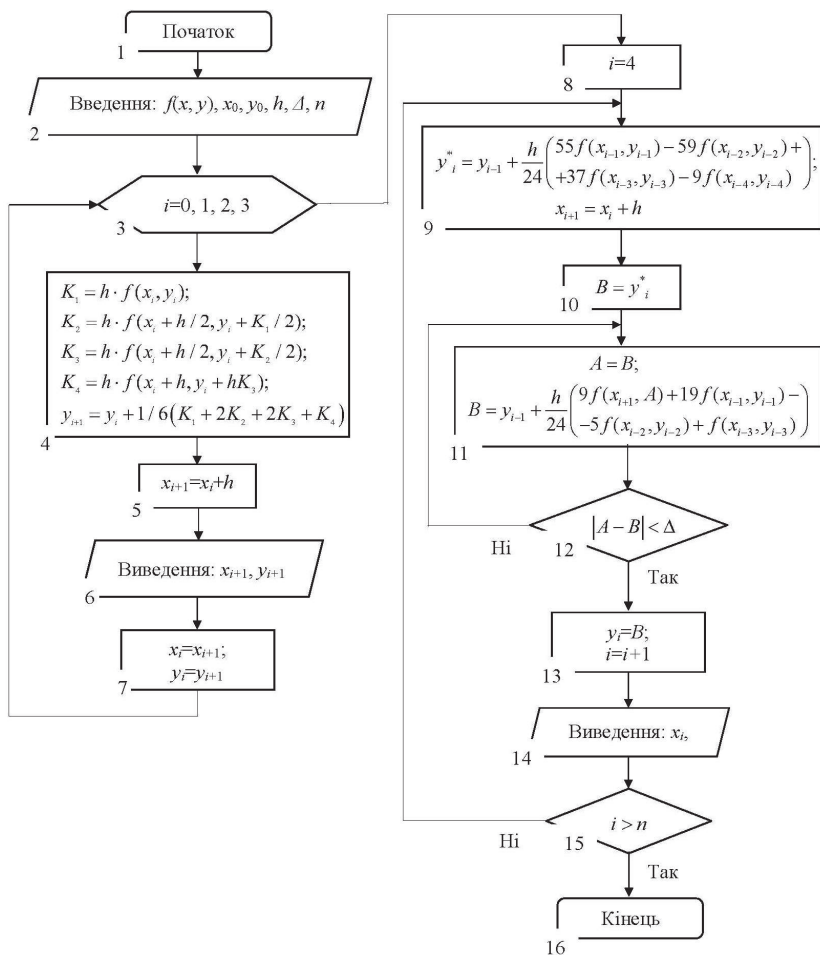


Рисунок 3.11 – Схема алгоритму методу Адамса

Ставиться задача визначення значення наближеного розв'язку  $\overline{y}_n = \overline{y}(x_n)$  ( $n = \overline{1, N}$ ) за такими формулами (узагальнення формули (3.51) на випадок диференціальних рівнянь):

$$\begin{cases} \overline{y}_m^* = \overline{y}_{m-1} + \frac{h}{24} (55\overline{f}_{m-1} - 59\overline{f}_{m-2} + 37\overline{f}_{m-3} - 9\overline{f}_{m-4}); \\ \overline{f}_m^* = \overline{f}(x_m, \overline{y}_m^*); \\ \overline{y}_m = \overline{y}_{m-1} + \frac{h}{24} (9\overline{f}_m^* + 19\overline{f}_{m-1} - 5\overline{f}_{m-2} + \overline{f}_{m-3}) \quad (m = \overline{1, N-1}). \end{cases} \quad (3.53)$$

Отже, отримавши значення  $\bar{y}_0, \bar{y}_1, \bar{y}_2, \bar{y}_3$  та  $\bar{f}_0, \bar{f}_1, \bar{f}_2, \bar{f}_3$  за формулами (3.40), обчислюються значення  $\bar{y}_4$  і  $\bar{f}_4$  за формулами (3.53). Беручи  $(x_4, \bar{y}_4)$  та  $\bar{f}(x_4, \bar{y}_4)$  за вихідні дані і повторюючи цей самий процес, визначається  $\bar{y}_5$  і т. д.

Алгоритм методу Адамса для розв'язання систем ЗДР подано на рисунку 3.12.

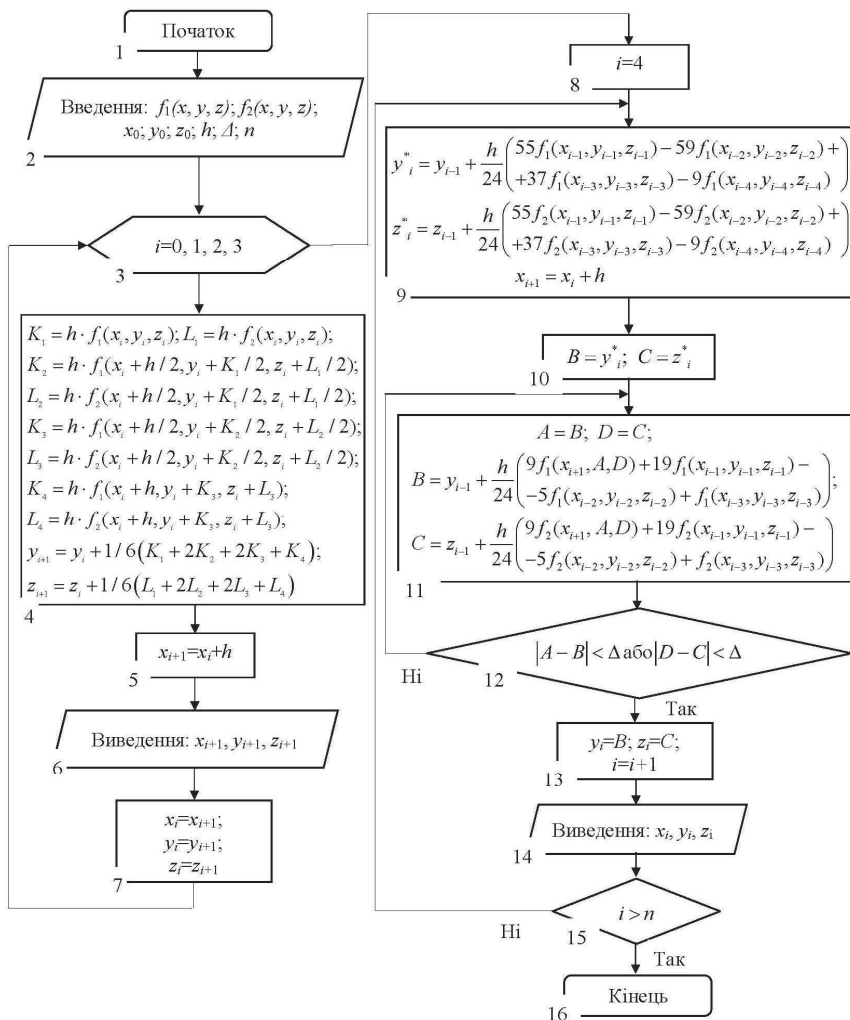


Рисунок 3.12 – Схема алгоритму методу Адамса для систем ЗДР

**Приклад 3.5.** Розв'язати чисельним методом «прогнозування та корекції» на основі ітераційних формул Адамса задачу Коші для ЗДР першого порядку:

$$\frac{dy}{dx} = y + x; \quad x \in [0; 1,0], \quad (3.54)$$

яка задовольняє початкову умову за  $x_0 = 0, y_0 = 1,0$  і абсолютна похибка на етапі коригування  $\Delta = 10^{-4}$ .

*Розв'язання:*

Для розв'язання цієї задачі відрізок  $[0; 1,0]$  можна розділити на десять частин точками  $x_0 = 0; 0,1; 0,2; \dots, 1,0$ . Відповідно  $h = 0,1$ . Для початку визначаються значення  $\bar{y}_0, \bar{y}_1, \bar{y}_2, \bar{y}_3$  методом Рунге-Кутта четвертого порядку в прикладі 3.3 (див. табл. 3.3). Також для порівняння було отримано значення  $\tilde{y}_1, \tilde{y}_2, \tilde{y}_3, \tilde{y}_4$  аналітичного розв'язку ЗДР (3.54) і зведено в таблицю 3.5. Застосування чисельного методу «прогнозування та корекції» починається із отримання розв'язків ЗДР явним методом Адамса-Башфорта (3.49) різницевої схеми (3.51). Для початку вибирається «початковий відрізок» із таблиці 3.5, а саме:  $x_0 = 0,0; x_1 = 0,1; x_2 = 0,2; x_3 = 0,3; \bar{y}_0 = 1,0; \bar{y}_1 = 1,11034; \bar{y}_2 = 1,24280; \bar{y}_3 = 1,39971$ .

Тоді виконується розрахунок «прогнозування» (явний метод Адамса – Башфорта (3.49)):

$$\left\{ \begin{aligned} \bar{y}_4^* &= \bar{y}_3(x_3) + \frac{h}{24} (55f(x_3, \bar{y}_3) - 59f(x_2, \bar{y}_2) + 37f(x_1, \bar{y}_1) - 9f(x_0, \bar{y}_0)); \\ \bar{f}_4^* &= f(x_4, \bar{y}_4^*) = x_4 + \bar{y}_4^*(x_4); \\ \bar{y}_4^* &= 1,39971 + \frac{0,1}{24} (55(0,3 + 1,39971) - 59(0,2 + 1,24280) + 37(0,1 + 1,11034) - 9(1,0)) = 1,58365; \\ \bar{f}_4^* &= f(x_4, \bar{y}_4^*) = 0,4 + 1,58365 = 1,98363; \\ \bar{y}_5^* &= \bar{y}_4(x_4) + \frac{h}{24} (55f(x_4, \bar{y}_4^*) - 59f(x_3, \bar{y}_3) + 37f(x_2, \bar{y}_2) - 9f(x_1, \bar{y}_1)); \\ \bar{f}_5^* &= f(x_5, \bar{y}_5^*) = x_5 + \bar{y}_5^*(x_5); \\ \bar{y}_5^* &= 1,58364 + \frac{0,1}{24} (55(0,4 + 1,58365) - 59(0,3 + 1,39971) + 37(0,2 + 1,24280) - 9(0,1 + 1,11034)) = 1,79741; \\ \bar{f}_5^* &= f(x_5, \bar{y}_5^*) = 0,5 + 1,79741 = 2,29741; \end{aligned} \right.$$

Для корекції результатів, отриманих за явним методом Адамса – Башфорта (3.49) застосовується неявний метод Адамса–Моултона (3.50) різницевої схеми (3.51). Для початку обирається «початковий відрізок» із таблиці 3.5, а саме:  $x_0 = 0,0$ ;  $x_1 = 0,1$ ;  $x_2 = 0,2$ ;  $x_3 = 0,3$ ;  $\bar{y}_0 = 1,0$ ;  $\bar{y}_1 = 1,11034$ ;  $\bar{y}_2 = 1,24280$ ;  $\bar{y}_3 = 1,39971$ . Тоді виконується розрахунок «корекція» (неявний метод Адамса–Моултона (3.50)):

$$\begin{aligned} \overline{\overline{y}}_4(x_4) &= \overline{y}_3(x_3) + \frac{h}{24} \left( 9\overline{\overline{f}}^*(x_4, \overline{\overline{y}}_4^*) + 19f(x_3, \overline{y}_3) - 5f(x_2, \overline{y}_2) + f(x_2, \overline{y}_2) \right) = \\ &= 1,39971 + \frac{0,1}{24} (9 \cdot 1,98363 + 19 \cdot 1,39972 - 5 \cdot 1,24280 + 1,11034) = 1,58364; \end{aligned}$$

$$\begin{aligned} \overline{\overline{y}}_5(x_5) &= \overline{\overline{y}}_4(x_4) + \frac{h}{24} \left( 9\overline{\overline{f}}^*(x_5, \overline{\overline{y}}_5^*) + 19\overline{\overline{f}}^*(x_4, \overline{\overline{y}}_4) - 5f(x_3, \overline{y}_3) + f(x_2, \overline{y}_2) \right) = \\ &= 1,39971 + \frac{0,1}{24} (9 \cdot 2,29741 + 19 \cdot 1,98363 - 5 \cdot 1,39972 + 1,24280) = 1,79743; \end{aligned}$$

.....

Також для досягнення найбільшої точності обчислення, процес корекції (неявний метод Адамса–Моултона (3.50)) може бути повторено декілька разів на тому самому ітераційному кроці для отримання значення із певною заданою точністю  $\Delta$  (абсолютна похибка), що і було реалізовано мовою програмування PYTHON (див. рис. 3.12):

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
#введення кроку інтегрування
h=float(input())
#введення початкового значення аргументу інтегрування (загального проміжку
# інтегрування)
a=int(input()); x_mtr=np.array(a)
#введення початкового значення аргументу функції диф. рівняння
y_0=int(input()); y_mtr=np.array(y_0)
#введення кінцевого значення загального проміжку інтегрування
b=int(input())
#введення значення точності обчислення формулою корекції методу Адамса
e=float(input())
#задання функції диф. рівняння
def f(x,y):
    return x+y
#визначення перших трьох значень розв'язку диференціального рівняння
x=a; y=y_0
x_mas=[x]; y_mas=[y]
x_mtr=np.array(a)
y_mtr=np.array(y_0)
while x<3*h:
    #ітераційна формула методу Рунге-кутта
    k1=f(x,y); k2=f(x+(h/2),y+(h/2*k1))
    k3=f(x+(h/2),y+(h/2*k2))
    k4=f(x+(h/2),y+(h*k3))
    y=y+(h/6*(k1+(2*k2)+(2*k3)+k4))
    x=x+h
```

```

x_mas.append(x)
y_mas.append(y)
x_mtr=np.append(x_mtr, x)
i=4
while x<=b-h:
#ітераційна формула прогнозування методу Адамса
y=y_mas[i-1]+(h/24*(55*f(x_mas[i-1],y_mas[i-1])-59*f(x_mas[i-2],y_mas[i-2])+
37*f(x_mas[i-3],y_mas[i-3])-9*f(x_mas[i-4],y_mas[i-4])))
x=x+h
f_tran=y
s_tran=y+1
#ітераційна формула корекції методу Адамса
while abs(s_tran-f_tran)<=e:
s_tran=f_tran
f_tran=y_mas[i-1]+(h/24*(9*f(x,s_tran)+19*f(x_mas[i-1],y_mas[i-1])-
5*f(x_mas[i-2],y_mas[i-2]))+f(x_mas[i-3],
y_mas[i-3])))
i=i+1
x_mas.append(x)
y_mas.append(f_tran)
x_mtr=np.append(x_mtr, x)
x_mtr=np.append(x_mtr, x)
#будуємо графіки рішень диф. рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
plt.plot(x_mas, y_mas)
plt.plot(x_mtr, (2*(e**x_mtr))-x_mtr-1)
plt.legend(['Метод Адамса','f(x)=2*exp(x)-x-1'],loc=1)
plt.grid(True)
plt.xlim([0, 1])
plt.ylim([1, 4])
plt.show()

```

Із отриманих результатів, наведених у таблиці 3.5, видно високий ступінь абсолютної і відносної похибки, допущених в процесі визначення розв'язку, зокрема, в точках  $x_5 = 0,5$  і  $x_{10} = 1,0$ :

– відносна похибка шляхом порівняння між значеннями аналітичного розв'язку і методом Адамса

$$\varepsilon_{x_5} = \left| \frac{\tilde{y}_{x_5} - \bar{y}_{x_5}}{\tilde{y}_{x_5}} \right| \cdot 100\% = \left| \frac{1,79744 - 1,79743}{1,79744} \right| \cdot 100\% = 5,56 \cdot 10^{-4} \%,$$

$$\varepsilon_{x_{10}} = \left| \frac{\tilde{y}_{x_{10}} - \bar{y}_{x_{10}}}{\tilde{y}_{x_{10}}} \right| \cdot 100\% = \left| \frac{3,43656 - 3,43652}{3,43656} \right| \cdot 100\% = 1,16 \cdot 10^{-3} \%;$$

– абсолютна похибка шляхом порівняння між значеннями «прогнозування»  $\bar{y}_n^*(x_n)$  (Адамса-Башфорта) і «коригування»  $\bar{y}_n(x_n)$  (Адамса-Моултона)

$$\Delta_{x_5} = \left| \bar{y}_{x_5} - \bar{y}_{x_5}^* \right| = |1,79743 - 1,79741| = 2,0 \cdot 10^{-5},$$

$$\Delta_{x_{10}} = \left| \bar{y}_{x_{10}} - \bar{y}_{x_{10}}^* \right| = |3,43652 - 3,43644| = 8,0 \cdot 10^{-5}.$$

Таблиця 3.5 – Результати обчислення розв’язку диференціального рівняння

$n$	$x_n$	$\bar{y}_n(x_n)$	$\tilde{y}_n(x_n)$	$\overline{\overline{y}}_n^*(x_n)$	$\overline{\overline{y}}_n(x_n)$
0	0,0	1,00000	1,00000	1,00000	1,00000
1	0,1	1,11034	1,11034	1,11034	1,11034
2	0,2	1,24280	1,24281	1,24280	1,24280
3	0,3	1,39971	1,39972	1,39971	1,39971
4	0,4	1,58364	1,58365	1,58363	1,58364
5	0,5	1,79743	1,79744	1,79741	1,79743
6	0,6	2,04423	2,04424	2,04410	2,04423
7	0,7	2,32750	2,32751	2,32745	2,32749
8	0,8	2,65107	2,65108	2,65100	2,65106
9	0,9	3,01919	3,01921	3,01911	3,01918
10	1,0	3,43655	3,43656	3,43644	3,43652

Також в самому чисельному методі «прогнозування та корекція» можна відслідкувати підвищення точності обчислення за використання процесу «коригування» (методи Адамса-Моултона) на основі попередньо отриманих результатів чисельного розв’язання ЗДР на базі ітераційного рівняння «прогнозування» (метод Адамса-Бошфорта) задачі Коші ЗДР.

Оскільки для розв’язання задачі Коші методом «прогнозування та корекції» необхідними є дані перших початкових значень розв’язку системи ЗДР, тому за основу, для прикладу застосування методу «прогнозування та корекції», буде вибрано дані розв’язку методом Рунге-Кутта четвертого порядку, розглянутого в прикладі 3.4.

**Приклад 3.6.** Розв’язати чисельним методом «прогнозування та корекції» на основі ітераційних формул Адамса задачу коливань маятника в середовищі, яке створює опір руху:

$$\frac{d^2\theta}{dt^2} + 0,2 \frac{d\theta}{dt} + 10 \sin \theta = 0; t \in [0; 1,0], \quad (3.55)$$

за початкових умов  $\theta(0) = 0,3$ ;  $\frac{d\theta}{dt}(0) = 0$  і абсолютної похибки на етапі коригування  $\Delta = 10^{-3}$ . У рівнянні (2.60)  $\theta(t)$  – функція кута відхилення маятника, яка залежить від часу  $t$ .

*Розв’язання:*

Для розв’язання цієї задачі відрізок  $[0; 1,0]$  можна розділити на десять частин точками  $x_0=0; 0,1; 0,2; \dots, 1,0$ . Відповідно  $h = 0,1$ . Для розв’язання задачі вводиться заміна  $z = \frac{d\theta}{dt}$ . Тоді рівняння (3.55) із початковими умовами може бути записано у вигляді системи ЗДР:

$$\begin{cases} \frac{d\theta}{dt} = z; \\ \frac{dz}{dt} = -0,2z - 10\sin\theta; \\ \theta(0) = 0,3; \quad z(0) = 0. \end{cases} \quad (3.56)$$

Для початку визначаються значення  $\bar{\theta}_0, \bar{\theta}_1, \bar{\theta}_2, \bar{\theta}_3$  і  $\bar{z}_0, \bar{z}_1, \bar{z}_2, \bar{z}_3$  методом Рунге-Кутта четвертого порядку, які визначено в прикладі 3.4 і наведено в таблиці 3.4. Також для порівняння було отримано значення  $\tilde{\theta}_0, \tilde{\theta}_1, \tilde{\theta}_2, \tilde{\theta}_3$  аналітичного розв'язку ЗДР (3.43) і наведено в таблиці 3.4. Застосування чисельного методу «прогнозування та корекції» починається із отримання розв'язків ЗДР явними методами Адамса-Башфорта різницевої схеми (3.53). Для початку вибирається «початковий відрізок» із таблиці 3.6, а саме:  $t_0=0; t_1=0,1; t_2=0,2; t_3=0,3; \bar{\theta}_0=0,3; \bar{\theta}_1=0,28544; \bar{\theta}_2=0,24352; \bar{\theta}_3=0,17876; \bar{z}_0=0; \bar{z}_1=-0,28792; \bar{z}_2=-0,54295; \bar{z}_3=-0,74113$ . Після чого виконується розрахунок «прогнозування» (явний метод Адамса – Башфорта різницевої схеми (3.53)):

$$\begin{cases} \bar{\theta}_4^* = \bar{\theta}_3(t_3) + \frac{h}{24} (55f(t_3, \bar{z}_3, \bar{\theta}_3) - 59f(t_2, \bar{z}_2, \bar{\theta}_2) + 37f(t_1, \bar{z}_1, \bar{\theta}_1) - 9f(t_0, \bar{z}_0, \bar{\theta}_0)); \\ \bar{z}_4^* = \bar{z}_3(t_3) + \frac{h}{24} (55\xi(t_3, \bar{z}_3, \bar{\theta}_3) - 59\xi(t_2, \bar{z}_2, \bar{\theta}_2) + 37\xi(t_1, \bar{z}_1, \bar{\theta}_1) - 9\xi(t_0, \bar{z}_0, \bar{\theta}_0)); \\ \bar{f}_4^* = f(t_4, \bar{z}_4^*, \bar{\theta}_4^*) = \bar{z}_4^*(t_4); \\ \bar{\xi}_4^* = \xi(t_4, \bar{z}_4^*, \bar{\theta}_4^*) = -0,2\bar{z}_4^* - 10\sin(\bar{\theta}_4^*); \\ \bar{\theta}_4^* = 0,17876 + \frac{0,1}{24} (55(-0,74113) - 59(-0,54295) + 37(-0,28792) - 9 \cdot 0) = 0,09801; \\ \bar{z}_4^* = -0,74113 + \frac{0,1}{24} (55 \cdot (-0,2 \cdot (-0,74113) - 10\sin(0,17876)) - 59 \cdot (-0,2 \cdot (-0,54295) - 10\sin(0,24352)) + 37 \cdot (-0,2 \cdot (-0,28792) - 10\sin(0,28544)) - 9 \cdot (-0,2 \cdot 0 - 10\sin(0,3))) = -0,86299; \\ \bar{f}_4^* = \bar{z}_4^*(t_4) = -0,86299; \\ \bar{\xi}_4^* = -0,2\bar{z}_4^* - 10\sin(\bar{\theta}_4^*) = -0,2(-0,86299) - 10\sin(0,09801) = -0,80589; \end{cases}$$



Для корекції результатів, отриманих за явним методом Адамса – Башфорта, застосовується метод (неявний) Адамса-Моултона різницевої схеми (3.53). Для початку обирається «початковий відрізок» із таблиці 3.6, а саме:  $t_0=0$ ;  $t_1=0,1$ ;  $t_2=0,2$ ;  $t_3=0,3$ ;  $\bar{\theta}_0=0,3$ ;  $\bar{\theta}_1=0,28544$ ;  $\bar{\theta}_2=0,24352$ ;  $\bar{\theta}_3=0,17876$ ;  $\bar{z}_0=0$ ;  $\bar{z}_1=-0,28792$ ;  $\bar{z}_2=-0,54295$ ;  $\bar{z}_3=-0,74113$ . Тоді виконується розрахунок «корекція» (неявний метод Адамса-Моултона) різницевої схеми (3.53):

$$\begin{cases} \bar{\theta}_4(t_4) = \bar{\theta}_3(t_3) + \frac{h}{24} \left( 9\bar{f}^*(t_4, \bar{z}_4, \bar{\theta}_4) + 19f(t_3, \bar{z}_3, \bar{\theta}_3) - 5f(t_2, \bar{z}_2, \bar{\theta}_2) + f(t_1, \bar{z}_1, \bar{\theta}_1) \right); \\ \bar{z}_4(t_4) = \bar{z}_3(t_3) + \frac{h}{24} \left( 9\bar{\zeta}^*(t_4, \bar{z}_4, \bar{\theta}_4) + 19\zeta(t_3, \bar{z}_3, \bar{\theta}_3) - 5\zeta(t_2, \bar{z}_2, \bar{\theta}_2) + \zeta(t_1, \bar{z}_1, \bar{\theta}_1) \right); \end{cases}$$

$$\begin{cases} \bar{\theta}_4(t_4) = 0,17876 + \frac{0,1}{24} \left( 9 \cdot (0,86299) + 19 \cdot (-0,74113) - \right. \\ \left. -5 \cdot (-0,54295) + (-0,28792) \right) = 0,097840; \\ \bar{z}_4^*(t_4) = -0,74113 + \frac{0,1}{24} \left( 9 \cdot (-0,80589) + \right. \\ \left. + 19 \cdot (-0,2 \cdot (-0,74113) - 10 \sin(0,17876)) - \right. \\ \left. -5 \cdot (-0,2 \cdot (-0,54295) - 10 \sin(0,24352)) + \right. \\ \left. + (-0,2 \cdot (-0,28792) - 10 \sin(0,28544)) \right) = -0,86390; \end{cases}$$

Також для досягнення найбільшої точності обчислення, процес корекції (неявний метод Адамса-Моултона) різницевої схеми (3.53) може бути повторено декілька разів на тому самому ітераційному кроці для отримання значення із певною заданою точністю  $\Delta$  (абсолютна похибка), що і було реалізовано мовою програмування PYTHON (див. рис. 3.12):

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
#введення кроку інтегрування
h=float(input())
#введення початкового значення аргументу інтегрування (загального проміжку
# інтегрування) t0
a=int(input()); t_mtr=np.array(a)
#введення початкового значення аргументу функції диф. рівняння q0
q_0=float(input()); q_mtr=np.array(q_0)
#введення початкового значення аргументу функції диф. рівняння z0
z_0=int(input()); z_mtr=np.array(z_0)
#введення кінцевого значення загального проміжку інтегрування
b=int(input())
#введення значення точності обчислення формулою корекції методу Адамса
e=float(input())
#задання функції системи диф. рівняння
def f1(z):
    return z
def f2(q,z):
```

```

    return (-0.2*z)-(10*math.sin(q))
#визначення перших трьох значень розв'язку диференціального рівняння
t=a; q=q_0; z=z_0
t_mas=[t]; q_mas=[q]; z_mas=[z]
t_mtr=np.array(t)
q_mtr=np.array(q_0)
z_mtr=np.array(z_0)
while t<3*h:
    #ітераційні формули методу Рунге-Кутта
    K1_Z=f2(q, z); K1_Q=f1(z)
    K2_Z=f2(q+(h/2*K1_Q), z+(h/2*K1_Z)); K2_Q=f1(z+(h/2*K1_Z))
    K3_Z=f2(q+(h/2*K2_Q), z+(h/2*K2_Z)); K3_Q=f1(z+(h/2*K2_Z))
    K4_Z=f2(q+(h*K3_Q), z+(h*K3_Z)); K4_Q=f1(z+(h*K3_Z))
    z=z+(h/6*(K1_Z+(2*K2_Z)+(2*K3_Z)+K4_Z))
    q=q+(h/6*(K1_Q+(2*K2_Q)+(2*K3_Q)+K4_Q))
    t=t+h
    t_mas.append(t)
    q_mas.append(q)
    z_mas.append(z)
    t_mtr=np.append(t_mtr, t)
i=4
while t<=b-h:
    #ітераційна формула прогнозування методу Адамса
    q=q_mas[i-1]+(h/24*(55*f1(z_mas[i-1])-59*f1(z_mas[i-2]))+
    37*f1(z_mas[i-3])-9*f1(z_mas[i-4])))
    z=z_mas[i-1]+(h/24*(55*f2(q_mas[i-1], z_mas[i-1])-59*f2(q_mas[i-2],
    z_mas[i-2]))+
    37*f2(q_mas[i-3], z_mas[i-3])-9*f2(q_mas[i-4],
    z_mas[i-4])))
    t=t+h
    b_tran=q; c_tran=z
    a_tran=q+1; d_tran=z+1
    #ітераційна формула корекції методу Адамса
    while abs(b_tran-a_tran)<=e or abs(c_tran-d_tran)<=e:
        a_tran=b_tran; d_tran=c_tran
        b_tran=q_mas[i-1]+(h/24*(9*f1(d_tran)+19*f1(z_mas[i-1])-
        5*f1(z_mas[i-2])+f1(z_mas[i-3])))
        c_tran=z_mas[i-1]+(h/24*(9*f2(a_tran, d_tran)+
        19*f2(q_mas[i-1], z_mas[i-1])-
        5*f2(q_mas[i-2], z_mas[i-2])+
        f2(q_mas[i-3], z_mas[i-3])))
    i=i+1
    t_mas.append(t)
    q_mas.append(b_tran)
    z_mas.append(c_tran)
    t_mtr=np.append(t_mtr, t)
#будуємо графіки рішень диф. рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('t', fontsize=15, color='blue')
plt.ylabel('psi, dPsi(t)/dt', fontsize=15, color='blue')
plt.plot(t_mas, q_mas)
plt.plot(t_mas, z_mas)
plt.plot(t_mtr, 0.3*(np.exp(-0.1*t_mtr))*
(np.cos(3.1607*t_mtr)+(0.03164*np.sin(3.1607*t_mtr))))
plt.legend(['Метод Рунге-Кутта для Psi(t)', 'Метод Рунге-Кутта для
dPsi(t)/dt', 'Точний розв'язок'], loc=1)
plt.grid(True)
plt.xlim([0, 1])
plt.ylim([-0.95, 0.45])
plt.show()

```

Із отриманих результатів, наведених у таблиці 3.6, видно високу точність обчислень відносно результатів аналітичного розрахунку ( $\hat{\theta}(t_n)$ ) (див. табл. 3.4), зокрема, в точках  $t_4 = 0,4$  і  $t_{10} = 1,0$ :

– відносна похибка шляхом порівняння між значеннями аналітичного розв'язку і методом Адамса

$$\varepsilon_{t_4} = \left| \frac{\tilde{\theta}_{t_4} - \bar{\theta}_{t_4}}{\bar{\theta}_{t_4}} \right| \cdot 100\% = \left| \frac{0,09567 - 0,09783}{0,09567} \right| \cdot 100\% = 2,26\%,$$

$$\varepsilon_{t_{10}} = \left| \frac{\tilde{\theta}_{t_{10}} - \bar{\theta}_{t_{10}}}{\bar{\theta}_{t_{10}}} \right| \cdot 100\% = \left| \frac{(-0,27157) - (-0,27163)}{-0,27157} \right| \cdot 100\% = 0,02\% ;$$

– абсолютна похибка шляхом порівняння між значеннями «прогнозування»  $\bar{\theta}_n^*(t_n)$  (Адамса-Башфорта) і «коригування»  $\bar{\theta}_n(x_n)$  (Адамса-Моултона)

$$\Delta_{t_4} = \left| \bar{\theta}_{t_4} - \bar{\theta}_{t_4}^* \right| = |0,09783 - 0,09800| = 1,7 \cdot 10^{-4},$$

$$\Delta_{t_{10}} = \left| \bar{\theta}_{t_{10}} - \bar{\theta}_{t_{10}}^* \right| = |(-0,27157) - (-0,27163)| = 6,0 \cdot 10^{-5}.$$

Таблиця 3.6 – Результати обчислення розв’язку диференціального рівняння

$n$	$t_n, \text{с}$	$\bar{\theta}^*(t_n),$ град	$\frac{d\bar{\theta}^*}{dt}(t_n),$ град/с	$\bar{\theta}(t_n),$ град	$\frac{d\bar{\theta}}{dt}(t_n),$ град/с	$\tilde{\theta}(t_n),$ град
0	0,0	0,30000	0,00000	0,300000	0,000000	0,30000
1	0,1	0,28544	-0,28792	0,285440	-0,287920	0,28522
2	0,2	0,24352	-0,54295	0,243520	-0,542950	0,24274
3	0,3	0,17876	-0,74113	0,178760	-0,741130	0,17726
4	0,4	0,09800	-0,86298	0,097830	-0,86390	0,09567
5	0,5	0,00914	-0,89904	0,008901	-0,89987	0,00630
6	0,6	-0,07884	-0,84631	-0,07916	-0,84684	-0,08191
7	0,7	-0,15740	-0,71199	-0,15775	-0,71194	-0,16034
8	0,8	-0,21906	-0,51117	-0,21939	-0,51060	-0,22147
9	0,9	-0,25819	-0,26504	-0,25845	-0,26418	-0,25964
10	1,0	-0,27147	0,00166	-0,27163	0,00256	-0,27157

Потрібно відмітити у таблиці 3.6 на інших інтервалах часу  $t_n$  значну розбіжність між результатами чисельного й аналітичного (див. табл. 3.4) розрахунків, що також обумовлено отриманням аналітичного розрахунку (3.43) із припущення, що  $\theta \rightarrow 0$ . Проте результати чисельного обчислення «прогнозування та корекції» показують близькі значення із результатами обчислення методом Рунге-Кутта четвертого порядку в таблиці 3.4.

Порівняно з однокроковими методами, методи «прогнозування та корекції» мають низку особливостей:

1. Для реалізації методів «прогнозування та корекції» необхідно мати інформацію про кілька попередніх точок. Тому вони не належать до «самостартуючих» методів і для початку розв’язання необхідно використовувати певний однокроковий метод. Тому у процесі розв’язування диференціальних рівнянь не можна змінювати крок інтегрування.

2. Однокрокові методи та методи «прогнозування та корекції» забезпечують приблизно однакову точність результатів, проте другі, на відміну від перших, дозволяють легко оцінити похибку на кроці.

3. Застосовуючи метод Рунге-Кутта четвертого порядку на кожному кроці необхідно обчислювати чотири значення функції, тоді як для забезпечення збіжності в методі «прогнозування та корекції» того самого порядку точності достатньо двох значень функції. Тому методи «прогнозування та корекції» потребують майже вдвічі менше машинного часу, ніж методи Рунге-Кутта порівнянної точності.

### 3.3 Чисельні методи розв’язання звичайних диференціальних рівнянь для крайових задач

Під час постановки крайової задачі необхідно знайти розв’язок диференціального рівняння  $n$ -го порядку  $y^{(n)} = f(x, y, y', y'', \dots, y^{(n-1)})$  на відрізку  $a \leq x \leq b$  за крайових умов:

$$\begin{aligned} y(a) &= A_0, & y(b) &= B_0; \\ y'(a) &= A_1, & y'(b) &= B_1; \\ & \dots; \\ y^{(k)}(a) &= A_k, & y^{(m)}(b) &= B_m; \\ & \dots; \\ y^{(i)}(a) &= A_i, & y^{(j)}(b) &= B_j, \end{aligned}$$

де  $A_k, B_m$  – деякі константи ( $k=0, 1, 2, \dots, i; m=0, 1, 2, \dots, j$ ). Загальна кількість додаткових умов на кінцях відрізка  $[a; b]$  має дорівнювати порядку диференціального рівняння  $i+j+2=n$ .

У випадку системи диференціальних рівнянь має виконуватись те саме правило. Причому додаткові умови не можуть бути зосереджені на одному будь-якому кінці відрізка. Наприклад, для диференціальних рівнянь другого і третього порядків задача сформулюється так.

Розв’язати крайову задачу для рівнянь:

– другого порядку  $y'' = f(x, y, y')$  на відрізку  $a \leq x \leq b$  за крайових умов  $y(a) = A, y(b) = B$ ;

– третього порядку  $y''' = f(x, y, y', y'')$  на відрізку  $a \leq x \leq b$  за крайових умов  $y(a) = A$ ,  $y'(a) = C$ ,  $y(b) = B$  або за крайових умов  $y(a) = A$ ,  $y(b) = B$ ,  $y'(b) = D$ .

Константи  $A, B, C, D$  відповідають значенням вказаних функцій в точках  $a$  і  $b$ .

Конкретні крайові умови вибираються із фізичної постановки задачі.

Від крайових задач задача Коші відрізняється тим, що область, в якій має бути визначений розв'язок, заздалегідь не вказується. Проте задача Коші може розглядатись як одна із крайових задач. Задача Коші зазвичай виникає під час аналізування процесів, що визначаються диференціальним законом еволюції та початковим станом (математичним виразом яких і є рівняння та початкова умова). Більше того, існують методи, які дозволяють пошук розв'язку граничної задачі привести до пошуку розв'язків ряду задач Коші для відповідного диференціального рівняння. Одним із таких відомих чисельних методів розв'язку ЗДР для крайових задач є метод «стрілянини».

### Метод «стрілянини»

Метод «стрілянини» зводить розв'язання крайової задачі для ЗДР до розв'язання ітераційної послідовності задач Коші. Нехай необхідно розв'язати таку крайову задачу типу:

$$y''(x) = f(x, y, y'), \quad x \in [a; b]; \quad (3.57)$$

$$y(a) = A; \quad (3.58)$$

$$y(b) = B. \quad (3.59)$$

Замість крайової задачі (3.57)–(3.59) розглядається така задача Коші:

$$y''(x) = f(x, y, y'), \quad x \in [a; b]; \quad (3.60)$$

$$y(a) = A; \quad (3.61)$$

$$y'(a) = tg \alpha, \quad \alpha: y(b, \alpha) = B, \quad (3.62)$$

в якій інтегральна крива  $y(x, \alpha)$  залежить не тільки від змінної  $x$ , але й від параметра  $\alpha$ , що називається кутом пристрілювання (кут між дотичною до кривої розв'язку ЗДР і віссю абсцис). Він обирається із умови рівності значення інтегральної кривої на правій межі  $y(b, \alpha)$  значенню  $B$  із попередньо заданою точністю  $\varepsilon$  (рис. 3.13):

$$|y(b, \alpha) - B| \leq \varepsilon. \quad (3.63)$$

Кут пристрілювання, що задовольняє нерівність (3.63), буде позначатись через  $\alpha^*$ . Інтегральна крива, отримана із розв'язку задачі Коші (3.60)–(3.62) із кутом, близьким до цього значення, згідно з нерівністю (3.63) і буде розв'язком задачі (3.57)–(3.59) з точністю  $\varepsilon$ .

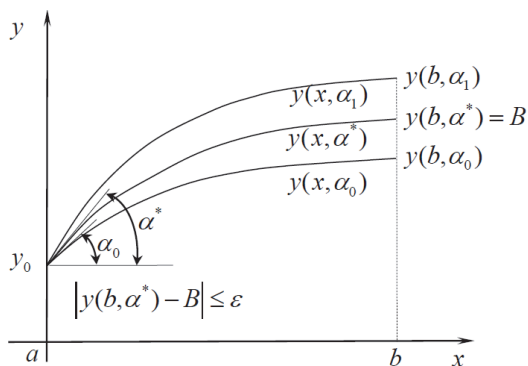


Рисунок 3.13 – Схема геометричної інтерпретації методу стрілянини

Таким чином, для реалізації методу «стрілянини» на початку вибирається початкове значення кута  $\alpha_0$  із умови  $\operatorname{tg}\alpha_0 = \frac{B-A}{b-a}$ . Із цим значенням  $\alpha_0$  одним із відомих чисельних методів, розглянутих вище, розв'язується задача Коші (3.60)–(3.62) із отриманням  $y(x, \alpha_0)$  і  $y(b, \alpha_0)$ ; якщо водночас виконується умова (3.63), тоді крайова задача (3.57)–(3.59) розв'язана із точністю  $\varepsilon$ .

В іншому випадку можуть бути такі два варіанти:

а) якщо  $y(b, \alpha_0) > B$ , тоді кут пристрілювання будь-яким способом зменшується і розв'язується задача Коші (3.60)–(3.62) тим самим чисельним методом до тих пір, поки не виконається умова  $y(b, \alpha_0) < B$ ;

б) якщо  $y(b, \alpha_0) < B$ , тоді кут пристрілювання будь-яким способом збільшується і розв'язується задача Коші до тих пір, поки не виконається умова  $y(b, \alpha_0) > B$ .

Таким чином, кут пристрілювання знаходиться в середині інтервалу  $\alpha \in [\alpha_0; \alpha_1]$ , після чого істинне значення  $\alpha^*$  кута пристрілювання визначається методом половинного ділення, а саме: наступне значення кута визначається за допомогою ітераційної формули  $\alpha_{k+1} = (\alpha_{k-1} + \alpha_k)/2$ ; визначається значення ординат  $y(x, \alpha_{k+1})$  і  $y(b, \alpha_{k+1})$  у відповідних точках після чого аналізується нерівність  $|y(b, \alpha) - B| \leq \varepsilon$ ; якщо вона виконується, тоді  $\alpha^* = (\alpha_{k-1} + \alpha_k)/2$  і  $y(x^*, \alpha_{k+1})$  – істинна інтегральна крива; якщо нерівність не виконується, тоді ітераційний процес повторюється спочатку.

Метод половинного ділення дуже повільно сходиться і доводиться вирішувати значну кількість задач Коші для різних кутів пристрілювання  $\alpha_k$ . Для прискорення збіжності ітераційного процесу використовується ітераційна формула Ньютона:

$$\alpha_{k+1} = \alpha_k + \frac{B - y(b, \alpha_k)}{y(b, \alpha_k) - y(b, \alpha_{k-1})} (\alpha_k - \alpha_{k-1}). \quad (3.64)$$

Для визначення попереднього значення кута пристрілювання  $\alpha_{k-1}$  в рівнянні (3.64) використовується проміжне значення, визначене на першому ітераційному кроці для  $k = 1$ , а саме:

$$\alpha_1 = \alpha_0 + \frac{B - y(b, \alpha_0)}{y(b, \alpha_0 + \delta) - y(b, \alpha_0)} \delta, \quad (3.65)$$

де  $\delta$  – мале значення приросту кута ( $\delta = 10^\circ \dots 20^\circ$ ).

Алгоритм методу «стрілянини» для розв’язання ЗДР крайових задач подано на рисунку 3.14.

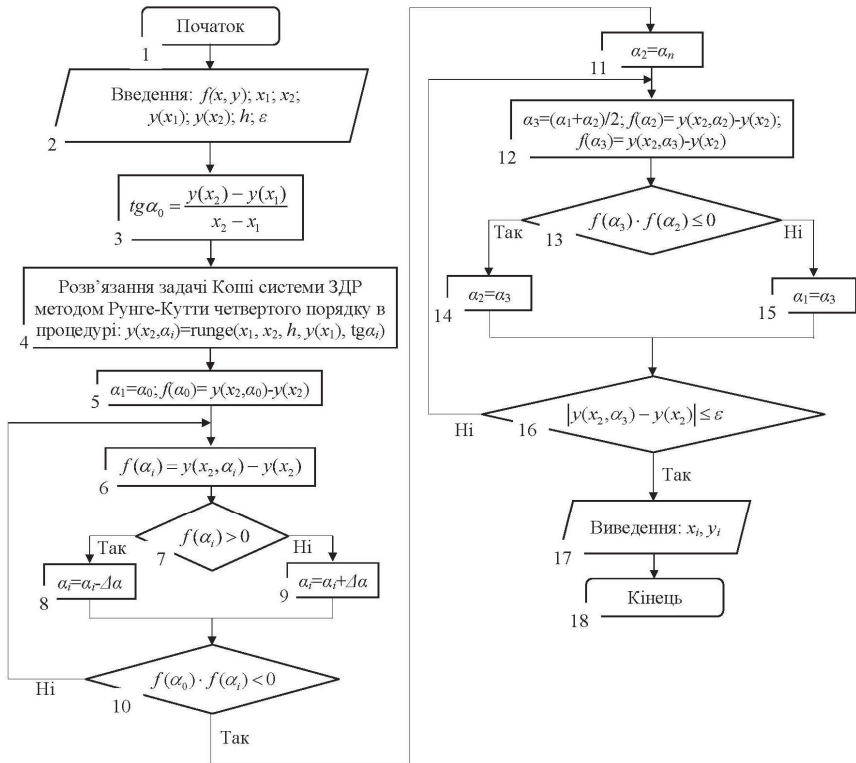


Рисунок 3.14 – Схема алгоритму метода «стрілянини»

**Приклад 3.7.** Розв'язати чисельним методом «стрілянини» крайову задачу для ЗДР другого порядку з точністю  $\varepsilon = 0,01$ :

$$y'' = \frac{1}{x}y' + x^2, \quad (3.66)$$

яка задовольняє початкові умови  $y(1, 0) = 0$  і  $y(2, 0) = 1$ .

*Розв'язання:*

Крайова задача (3.66) зводиться до задачі Коші:

$$\begin{cases} y'' = \frac{1}{x}y' + x^2; \\ y(1) = 0; \\ y'(1) = tg\alpha^*; \\ y(2, \alpha^*) = 1; \quad (\alpha = \alpha^*). \end{cases} \quad (3.67)$$

Рівняння  $f(\alpha) = y(2, \alpha) - y(2, \alpha^*) = y(2, \alpha) - 1$ , як нелінійне рівняння відносно кута пристрілювання  $\alpha$  може бути розв'язане одним із розглянутих чисельних ітераційних методів із паралельним розв'язанням задачі Коші на кожній ітерації.

Нехай початковий кут пристрілювання  $\alpha_0$  визначається із співвідношення:

$$tg\alpha_0 = \frac{B-A}{b-a} = \frac{1-0}{2-1} = 1; \quad \alpha_0 = 45^\circ.$$

Тоді задача Коші для ЗДР другого порядку (3.67) і відповідна задача Коші для нормальної системи ЗДР другого порядку на першій ітерації буде мати вигляд:

$$\begin{cases} y'' = \frac{1}{x}y' + x^2; \\ y(1, 0) = 0; \\ y'(1, 0) = tg\alpha_0 = 1; \end{cases} \rightarrow \begin{cases} y' = z; \\ z' = \frac{1}{x}z + x^2; \\ y(1, 0) = 0; \\ z(1, 0) = 1. \end{cases} \quad (3.68)$$

Розв'язуючи систему (3.68) з кроком  $h = 0,25$  чисельним методом Рунге-Кутта четвертого порядку, отримаємо:  $y(2, 45^\circ) = y_4 = 2,625$  (табл. 3.7). Оскільки  $f^1(45^\circ) = y(2, 45^\circ) - 1 = 2,625 - 1,0 = 0,625 > 0$ , то необхідно зменшити кут пристрілювання  $\alpha$  (наприклад  $\Delta\delta = 44^\circ$ ) таким чином, щоб значення  $f(\alpha)$  на наступній ітерації було менше нуля. Можна взяти  $\alpha_1 = \alpha_0 - \Delta\delta = 45^\circ - 44^\circ = 1^\circ$ , і тоді задача Коші на другій ітерації матиме вигляд:



$$\begin{cases} y'' = \frac{1}{x}y' + x^2; \\ y(1,0) = 0; \\ y'(1,0) = \operatorname{tg}\alpha_1 = 0,0175; \end{cases} \rightarrow \begin{cases} y' = z; \\ z' = \frac{1}{x}z + x^2; \\ y(1,0) = 0; \\ z(1,0) = 0,0175. \end{cases} \quad (3.69)$$

Розв'язуючи знову систему (3.69) з кроком  $h = 0,25$  чисельним методом Рунге-Кутта четвертого порядку, буде отримано:  $y(2, 1^\circ) = y_4 = 1,151$  (див. табл. 3.7). Оскільки  $f^2(1, 0^\circ) = y(2, 1^\circ) - 1 = 1,151 - 1,0 = 0,151 > 0$ , то необхідно зменшувати кут пристрілювання  $\alpha$  доти, доки значення  $f(\alpha)$  на наступній ітерації не буде меншим нуля. Тому, беручи  $\alpha_2 = \alpha_1 - \Delta\delta = 1^\circ - 44^\circ = -43^\circ$ , розв'язок задачі Коші на третій ітерації буде  $y(2, -43^\circ) = y_4 = -0,274$  (див. табл. 3.7), тобто  $f^3(1^\circ) = y(2, 1^\circ) - 1 = 1,151 - 1,0 = 0,151 > 0$ .

Відповідно, кут  $\alpha^*$  знаходиться усередині інтервалу:  $\alpha_1 = 1^\circ < \alpha^* < \alpha_2 = -43^\circ$ .

На четвертій ітерації кут пристрілювання вибирається на основі ітераційної формули  $\alpha_3 = (\alpha_1 + \alpha_2)/2 = (1^\circ + (-43^\circ))/2 = -21^\circ$ . Тоді задача Коші на четвертій ітерації буде мати вигляд:

$$\begin{cases} y'' = \frac{1}{x}y' + x^2; \\ y(1) = 0; \\ y'(1) = \operatorname{tg}\alpha_3 = -0,384; \end{cases} \rightarrow \begin{cases} y' = z; \\ z' = \frac{1}{x}z + x^2; \\ y(1) = 0; \\ z(1) = -0,384. \end{cases} \quad (3.70)$$

Розв'язуючи систему (3.70) з кроком  $h = 0,25$  чисельним методом Рунге-Кутта четвертого порядку, отримуємо:  $y(2; -21^\circ) = y_4 = 0,549$  (див. табл. 3.7). Тоді  $f^4(-21^\circ) = y(2; -21^\circ) - 1 = 0,549 - 1,0 = -0,451 < 0$ , а також  $|f^4(-21^\circ)| = 0,451 > \varepsilon = 0,01$ .

Порівняння значень функції  $f(\alpha)$  на четвертій ( $f^4(-21^\circ) < 0$ ) і на третій ітераціях ( $f^3(1^\circ) > 0$ ) означає, що  $-21^\circ < \alpha^* < 1^\circ$ .

Продовжуючи за аналогією розв'язання поставленої задачі Коші на наступних ітераціях ( $\alpha_4 = (\alpha_3 + \alpha_2)/2 = (1^\circ + (-21^\circ))/2 = -10^\circ$ ,  $-10^\circ < \alpha^* < 1^\circ$ ;  $\alpha_5 = (\alpha_4 + \alpha_3)/2 = (1^\circ + (-10^\circ))/2 = -4,5^\circ$ ) чисельним методом Рунге-Кутта четвертого порядку з кроком  $h = 0,25$ , результати зводяться у таблицю 3.7. На останньому, шостому, ітераційному кроці  $y(2, -4,5^\circ) = 1,007$ ,  $f^5(-4,5^\circ) = y(2, -4,5^\circ) - 1 = 1,007 - 1,0 = 0,007 > 0$ , а  $|f^5(-21^\circ) = 0,007 < \varepsilon = 0,01$ .

Таким чином, кут пристрілювання вирається  $\alpha^* = -4,5^\circ$ . На цьому куті інтегральна крива має значення  $y_i$  із рядка таблиці 3.7, що відповідає шостій ітерації.

Також можна відмітити відносно високу точність методу для крайових задач ЗДР порівняно із точним розв'язком  $\tilde{y}(x) = \frac{1}{8}x^4 - \frac{7}{24}x^2 + \frac{1}{6}$ , результати якого наведено у таблиці 3.7.

Таблиця 3.7 – Результати обчислення розв’язку диференціального рівняння крайової задачі

$i$	0	1	2	3	4
$x_i$	1,0	1,25	1,50	1,75	2,0
$\alpha_0=45^0$	1-а ітерація				
$y_i$	0,0	0,321	0,820	1,563	2,625
$\alpha_0=1^0$	2-а ітерація				
$y_i$	0,0	0,044	0,206	0,550	1,151
$\alpha_0=-43^0$	3-а ітерація				
$y_i$	0,0	-0,223	-0,388	-0,430	-0,274
$\alpha_0=-21^0$	4-а ітерація				
$y_i$	0,0	-0,068	-0,045	0,136	0,549
$\alpha_0=-10^0$	5-а ітерація				
$y_i$	0,0	-0,010	0,085	0,350	0,860
$\alpha_0=-4,5^0$	6-а ітерація				
$y_i$	0,0	0,017	0,146	0,451	1,007
$y_i$	0,0	0,016	0,143	0,446	1,000

Розглянемо реалізацію методу стрілянини для ЗДР другого порядку крайової задачі мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
import random
import math
#введення кроку інтегрування
h=float(input())
#введення початкового значення аргументу інтегрування (загального проміжку
# інтегрування) x1
x_a=int(input()); x_mtr=np.array(x_a)
#введення початкового значення аргументу функції диф. рівняння y(x1)
A=float(input())
#введення початкового значення аргументу функції диф. рівняння y(x2)
B=float(input())
#введення кінцевого значення загального проміжку інтегрування x2
x_b=int(input())
#введення значення похибки обчислення
e=float(input())
#задання функцій диф. рівняння 2-го порядку у вигляді системи диф. рівнянь
def f1(z):
    return z
def f2(x,z):
    return (z/x)+(x**2)
#задання функції обчислення диф. рівняння 2-го порядку методом Рунге-Кутта
# 4-го порядку,
def runge(x_a,x_b,h,y_0,z_0):
```

```

#обчислення ітераційних формул
x=x_a; y=y_0; z=math.tan(z_0*math.pi/180)
y_mas=[]; x_mas=[]; z_mas=[];
del y_mas
del z_mas
del x_mas
x_mas=[x]; y_mas=[y]; z_mas=[z]
x_mtr=np.array(x)
y_mtr=np.array(y_0)
z_mtr=np.array(z_0)
while x<x_b:
    #ітераційні формули методу Рунге-Кутта
    K1_Z=f2(x,z); K1_Y=f1(z)
    K2_Z=f2(x+(h/2),z+(h/2*K1_Z)); K2_Y=f1(z+(h/2*K1_Z))
    K3_Z=f2(x+(h/2),z+(h/2*K2_Z)); K3_Y=f1(z+(h/2*K2_Z))
    K4_Z=f2(x+h,z+(h*K3_Z)); K4_Y=f1(z+(h*K3_Z))
    z=z+(h/6*(K1_Z+(2*K2_Z)+(2*K3_Z)+K4_Z))
    y=y+(h/6*(K1_Y+(2*K2_Y)+(2*K3_Y)+K4_Y))
    x=x+h
    x_mas.append(x)
    y_mas.append(y)
    z_mas.append(z)
    x_mtr=np.append(x_mtr, x)
    return y_mas, z_mas, x_mas, x_mtr
#визначення даних початкових двох кутів пристрілювання
t_an=(B-A)/(x_b-x_a)
ang=math.atan(t_an)*180/math.pi
ang_sec=ang
while (runge(x_a,x_b,h,A,ang)[0][len(runge(x_a,x_b,h,A,ang)[0])-1]-B)*
(runge(x_a,x_b,h,A,ang_sec)[0][len(runge(x_a,x_b,h,A,ang_sec)[0])-1]-B)>0:
    ang_frst=ang_sec
    if (runge(x_a,x_b,h,A,ang_sec)[0][len(runge(x_a,x_b,h,A,ang_sec)[0])-1]-
    B)>0:
        ang_sec=ang_sec-44
    else:
        ang_sec=ang_sec+44
#визначення методом половинного ділення точного значення кута
    пристрілювання
ang_new=ang_frst
while abs(runge(x_a,x_b,h,A,ang_new)[0][len(runge(x_a,x_b,h,A,ang_new)[0])
-1]-B)>=e:
    ang_new=(ang_frst+ang_sec)/2
    if (runge(x_a,x_b,h,A,ang_new)[0][len(runge(x_a,x_b,h,A,ang_new)[0])-1]-
    B)*(runge(x_a,x_b,h,A,ang_frst)[0][len(runge(x_a,x_b,h,A,ang_frst)
    [0])-1]-B)<=0:
        ang_sec=ang_new
    else:
        ang_frst=ang_new
print('кут пристрілювання: alpha=',ang_new,'; тангенс кута пристрілювання:
    tg(alpha)=' ,math.tan(ang_new*math.pi/180))
print('Розв'язок ЗДР крайової задачі для yі=', runge(x_a,x_b,h,A,ang_new)[0])
#будуємо графіки рішень диф. рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('t',fontSize=15, color='blue')
plt.ylabel('psi, dPsi(t)/dt',fontSize=15, color='blue')
plt.plot(runge(x_a,x_b,h,A,ang_new)[2], runge(x_a,x_b,h,A,ang_new)[0])
plt.plot(runge(x_a,x_b,h,A,ang_new)[2], runge(x_a,x_b,h,A,ang_new)[1])
plt.plot(runge(x_a,x_b,h,A,ang_new)[3], (runge(x_a,x_b,h,A,ang_new)[3]**4
/8)-((7*runge(x_a,x_b,h,A,ang_new)[3]**2)/24)+(1/6))
plt.legend(['Метод стрілянини для y(x)', 'Метод стрілянини для y` (x)',
    'Точний розв'язок'], loc=1)
plt.grid(True)
plt.xlim([1, 2])
plt.ylim([0, 3])
plt.show()

```

## Метод кінцевих різниць

Нехай необхідно розв'язати таку крайову задачу типу:

$$y''(x) + p(x)y' + q(x)y = f(x), \quad x \in [a; b]; \quad (3.71)$$

$$y(a) = A, \quad y(b) = B. \quad (3.72)$$

де  $p(x)$ ,  $q(x)$ ,  $f(x)$  – відомі неперервні значення на відрізку  $[a; b]$  функції;  $A$  і  $B$  – задані постійні значення.

Одним із найбільш ефективних і популярних чисельних засобів розв'язання ЗДР та диференціальних рівнянь у частинних похідних є апарат різницевого методів.

У його основі лежить подання незалежного аргументу на відрізку  $[a; b]$  у вигляді дискретної множини точок  $x_i$  ( $i=0, 1, \dots, n$ );  $x_0=a$ ,  $x_n=b$ , яка називається сіткою.

Найбільше поширення отримала рівномірна сітка (3.46) із кроком  $x_i - x_{i-1} = h$  (рис. 3.15). В цьому випадку замість безперервної функції  $f(x)$  розглядається сіткова функція  $y_i = f(x_i)$ .

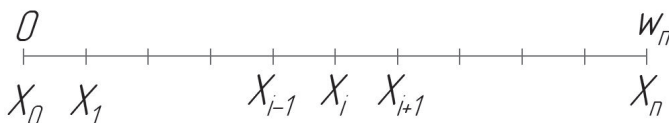


Рисунок 3.15 – Схема одновимірної розрахункової сітки

Сіткову функцію можна розглядати як функцію цілочисельного аргументу

$$y(i) = y_i \quad (i=0, \pm 1, \pm 2, \dots).$$

Для  $y_i$  можна ввести операції, які є дискретним (різницевим) аналогом операцій диференціювання та інтегрування.

Аналогом першої похідної є різниці першого порядку:

$$\Delta y_i = y_{i+1} - y_i \quad \text{— права різниця;}$$

$$\nabla y_i = y_i - y_{i-1} \quad \text{— ліва різниця;}$$

$$\delta y_i = \frac{1}{2}(\Delta y_i + \nabla y_i) = \frac{1}{2}(y_{i+1} - y_{i-1}) \quad \text{— центральна різниця.}$$

Потрібно зауважити, що  $\Delta y_i = \nabla y_{i+1}$ .

Далі записуються різниці другого порядку:

$$\Delta^2 y_i = \Delta(\Delta y_i) = \Delta(y_{i+1} - y_i) = y_{i+2} - 2y_{i+1} + y_i = \Delta \nabla y_{i+1};$$

$$\Delta \nabla y_i = \Delta(y_i - y_{i-1}) = (y_{i+1} - y_i) - (y_i - y_{i-1}) = y_{i+1} - 2y_i + y_{i-1}.$$

Аналогічно визначається різниця  $m$ -го порядку:

$$\Delta^m y_i = \Delta(\Delta^{m-1} y_i).$$

Очевидно, що:

$$\sum_{j=k}^i \Delta y_j = y_{i+1} - y_k; \quad \sum_{j=k}^i \Delta y_j = y_i - y_{k-1}.$$

На множині розрахункової сітки (див. рис. 3.15), що називається шаблоном, неперервний диференціальний оператор  $L_y$  замінюється різницеvim  $L_{hy}$ . Різницеві схеми будують шляхом заміни похідних у диференціальних рівняннях на різницеві відношення. Загальна формула для апроксимації похідних у деякій точці  $x_i$  має такий вигляд:

$$\frac{d^n y(x_i)}{dx^n} = \frac{1}{h^n} \sum_{s=-m}^l a_s y(x_i + sh) + O(h^p),$$

де коефіцієнти  $a_s$  ( $s=-m, -m+1, \dots, l$ ) підбираються таким чином, щоб досягти необхідного порядку апроксимації. Границя суми  $m$  і  $l$  підпорядковується умові  $m+l \geq n+p-1$ .

Дуже часто на практиці використовують різницеві відношення для апроксимації похідної із першим порядком за  $h$  на трьох вузлах сітки:

– права схема

$$L_h^+ y = \frac{dy(x_i)}{dx} = \frac{y(x_i + h) - y(x_i)}{h} + O(h) = y_x^+;$$

– – ліва схема

$$L_h^- y = \frac{dy(x_i)}{dx} = \frac{y(x_i) - y(x_i - h)}{h} + O(h) = y_x^-;$$

– – центральна схема

$$L_h^0 y = \frac{dy(x_i)}{dx} = \frac{y(x_i + h) - y(x_i - h)}{2h} + O(h) = y_x^0;$$

– центральна схема із другим порядком точності за  $h$

$$\begin{aligned} L_{hy} &= \frac{d^2 y(x_i)}{dx^2} = \frac{y(x_i + h) - 2y(x_i) + y(x_i - h)}{h^2} + O(h^2) = \\ &= \frac{y_x^+(x_i) - y_x^-(x_i)}{h} = \frac{y_x^-(x_i + h) - y_x^-(x_i)}{h} = y_{xx}^-(x). \end{aligned}$$

Під час отримання різницевої схеми важливу роль відіграє вимога, щоб різницева схема якнайкраще відображала основні властивості вихідного диференціального рівняння. Оцінення точності різницевої схеми зводиться до вивчення похибки апроксимації і стійкості.

У методі кінцевих різниць ЗДР розв'язання крайової задачі (3.71) і (3.72) зводиться до системи кінцево-різницевих рівнянь. Для цього основний відрізок  $[a; b]$  розбивається на  $n$  рівних частин довжиною  $h$  (крок), де  $h = (b-a)/n$  (рис. 3.16). Тобто область неперервної зміни аргументу  $[a; b]$  замінюється дискретною множиною точок, яка називається вузлами  $x_i$  ( $i = \overline{0, n}$ ).

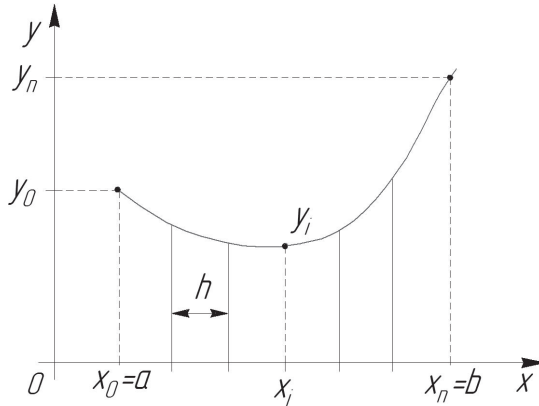


Рисунок 3.16 – Схема розрахункової сітки методу кінцевих різниць

Точки розбиття мають абсциси:

$$x_i = x_0 + ih \quad (i = 0, 1, 2, \dots, n), \quad x_0 = a, \quad x_n = b.$$

Значення в точках поділу  $x_i$  шуканої функції  $y = y(x)$  та її похідних  $y' = y'(x)$ ,  $y'' = y''(x)$  позначимо через  $y_i = y(x_i)$ ,  $y'_i = y'(x_i)$ ,  $y''_i = y''(x_i)$ .

Також вводяться позначення:

$$p_i = p(x_i), \quad q_i = q(x_i), \quad f_i = f(x_i).$$

Замінюючи похідні симетричними кінцево-різницевиими відношеннями, для внутрішніх точок  $x_i$  відрізка  $[a; b]$ :

$$y'_i = \frac{y_{i+1} - y_{i-1}}{2h}; \quad (3.73)$$

$$y''_i = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2}. \quad (3.74)$$

Підставляючи (3.73), (3.74) у вихідне рівняння (3.71) за  $x = x_i$  ( $i = \overline{1, n-1}$ ), отримуємо систему різницевих рівнянь:



Алгоритм методу кінцевих різниць для розв'язання ЗДР крайових задач подано на рисунку 3.17.

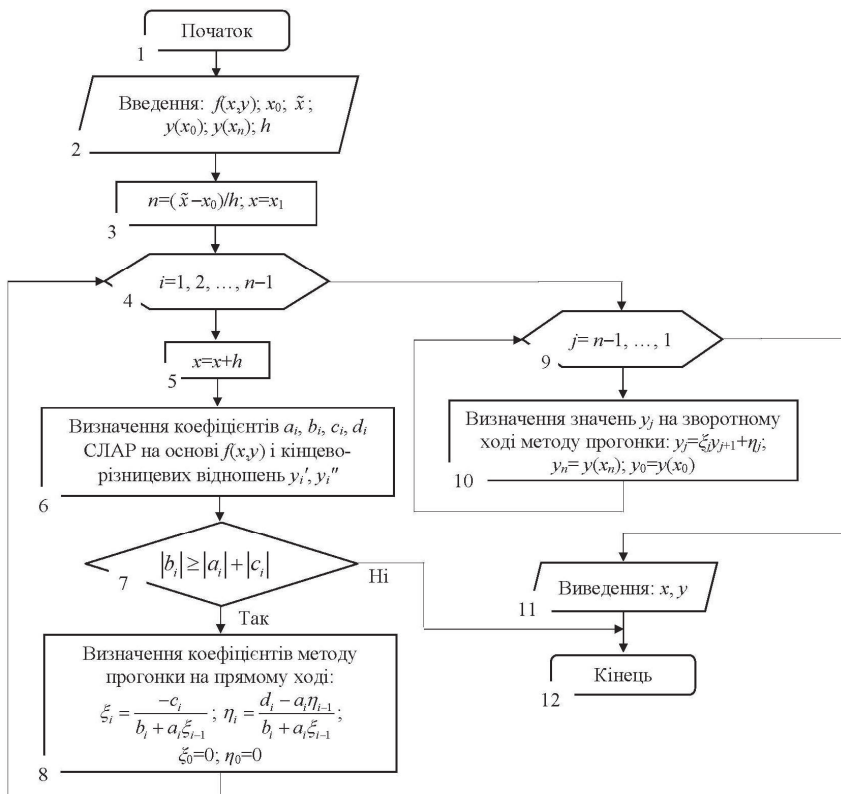


Рисунок 3.17 – Схема алгоритму методу кінцевих різниць

**Приклад 3.8.** Розв'язати чисельним методом кінцевих різниць крайову задачу для ЗДР другого порядку:

$$x^2 y'' + x^2 y' = 1, \quad (3.79)$$

яка задовольняє початкові умови за  $y(1, 0) = 0$ ;  $y(1, 4) = 0,0566$ .

*Розв'язання:*

Використовуючи формули (3.73), (3.74) замінюємо вихідне рівняння (3.79) системою кінцево-різницевих рівнянь:



$$x_i^2 \left( \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} \right) + x_i \left( \frac{y_{i+1} - y_{i-1}}{2h} \right) = 1.$$

Внаслідок зведення вільних членів, отримуємо:

$$y_{i-1}(2x_i^2 - hx_i) - 4x_i^2 y_i + y_{i+1}(2x_i^2 + hx_i) = 2h^2, \quad (3.80)$$

де  $a_i = 2x_i^2 - hx_i$ ,  $b_i = -4x_i^2$ ,  $c_i = 2x_i^2 + hx_i$ ,  $d_i = 2h^2$ .

Обираючи крок  $h = 0.1$ , буде отримано три внутрішніх вузли:  
 $x_i = 0, 1, i+1 \quad (i=1, 2, 3)$  де  $a_1 = 2x_1^2 - hx_1 = 2 \cdot 1,1^2 - 0,1 \cdot 1,1 = 2,31$ ;  
 $b_1 = -4x_1^2 = -4 \cdot 1,1^2 = -4,84$ ;  $c_1 = 2x_1^2 + hx_1 = 2 \cdot 1,1^2 + 0,1 \cdot 1,1 = 2,53$ ;  
 $d_1 = 2h^2 = 2 \cdot 0,1^2 = 0,02$ .

Аналогічно визначаються коефіцієнти для  $a_j, b_j, c_j, d_j \quad (j=2, 3)$ .

Написавши рівняння (3.80) для кожного із цих вузлів, отримаємо таку систему рівнянь:

$$\begin{cases} a_1 y_0 - b_1 y_1 + c_1 y_2 = d_1; \\ a_2 y_1 - b_2 y_2 + c_2 y_3 = d_2; \\ a_3 y_2 - b_3 y_3 + c_3 y_4 = d_3; \end{cases} \rightarrow \begin{cases} 2,31y_0 - 4,84y_1 + 2,53y_2 = 0,02; \\ 2,76y_1 - 5,76y_2 + 3,00y_3 = 0,02; \\ 3,25y_2 - 6,76y_3 + 3,51y_4 = 0,02. \end{cases} \quad (3.81)$$

Оскільки в отриманій тридіагональній матриці системи рівнянь (3.81) виконується умова переваги діагональних елементів ( $|b_i| \geq |a_i| + |c_i|$ ), то може бути використано метод прогонки.

Прогоночні коефіцієнти на прямому ході:

$$\xi_1 = \frac{-c_1}{b_1} = -2,53 / (-4,84) = 0,52273; \quad \eta_1 = \frac{d_1}{b_1} = 0,02 / (-4,84) = -0,00413;$$

$$\xi_2 = \frac{-c_2}{b_2 + a_2 \xi_1} = \frac{-3}{-5,76 + 2,76 \cdot 0,52273} = 0,69488;$$

$$\eta_2 = \frac{d_2 - a_2 \eta_1}{b_2 + a_2 \xi_1} = \frac{0,02 - 2,76 \cdot (-0,00413)}{-5,76 + 2,76 \cdot 0,52273} = -0,00727;$$

$$\xi_3 = \frac{-c_3}{b_3 + a_3 \xi_2} = \frac{-3,51}{6,76 + 3,25 \cdot 0,69488} = 0,77971;$$

$$\eta_3 = \frac{d_3 - a_3 \eta_2}{b_3 + a_3 \xi_2} = \frac{-0,02 - 3,25 \cdot (-0,00727)}{-6,76 + 3,25 \cdot 0,69488} = -0,0969.$$

На зворотному ході значення  $y_i$  визначаються за допомогою виразів  $y_i = \xi_i y_{i+1} + \eta_i$ :

$$\begin{cases} i=3: & y_3 = \xi_3 y_4 + \eta_3 = \eta_3 = 0,03446; \\ i=2: & y_2 = \xi_2 y_3 + \eta_2 = 0,69488 \cdot 0,03446 - 0,00727 = 0,01668; \\ i=1: & y_1 = \xi_1 y_2 + \eta_1 = 0,52273 \cdot 0,01668 - 0,00413 = 0,00459. \end{cases}$$

Також можна відмітити відносно високу точність методу кінцевих різниць для крайових задач ЗДР порівняно із точним розв'язком  $\tilde{y}(x) = 0,5 \ln^2 x$  у відповідних точках:

$$y(x_1) = y(1,1) = 0,0047; \quad y(x_2) = y(1,2) = 0,0166; \quad y(x_3) = y(1,3) = 0,0344.$$

Зокрема, в точках  $x_1 = 1,1$  і  $x_3 = 1,3$ :

– відносна похибка шляхом порівняння між значеннями аналітичного розв'язку і методом кінцевих різниць

$$\varepsilon_{x_1} = \left| \frac{\tilde{y}_{x_1} - y_{x_1}}{\tilde{y}_{x_1}} \right| \cdot 100\% = \left| \frac{0,00459 - 0,00470}{0,00459} \right| \cdot 100\% = 2,40\%,$$

$$\varepsilon_{x_3} = \left| \frac{\tilde{y}_{x_3} - y_{x_3}}{\tilde{y}_{x_3}} \right| \cdot 100\% = \left| \frac{0,03446 - 0,03440}{0,03446} \right| \cdot 100\% = 0,17\%.$$

Розглянемо реалізацію методу кінцевих різниць для ЗДР другого порядку крайової задачі мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
#введення кроку інтегрування
h=float(input())
#введення початкового значення аргументу інтегрування (загального проміжку
# інтегрування) x1
x_a=int(input()); x_mtr=np.array(x_a)
#введення кінцевого значення загального проміжку інтегрування x2
x_b=float(input())
#введення початкового значення аргументу функції диф. рівняння y(x1)
A=float(input())
#введення початкового значення аргументу функції диф. рівняння y(x2)
B=float(input())
#визначення коефіцієнтів системи різницевих рівнянь
n=round((x_b-x_a)/h)
x,a,b,c,d=[[],[0],[0],[0],[0]]
sig_ma,tet_ta=[0],[0]
x_tr=x_a
for i in range(0,n):
    x.append(x_tr)
    x_tr=x_tr+h
    x_mtr=np.append(x_mtr, x_tr)
for i in range(1,n):
    a.append(2*(x[i]**2)-(h*x[i]))
    b.append(-4*x[i]**2)
    c.append(2*(x[i]**2)+(h*x[i]))
    d.append(2*h**2)
```

```

#перевірка виконання умови переваги по модулю діагональних елементів
# матриці
if abs(b[i])<abs(a[i])+abs(c[i]):
    print('Система кінцево-різницевих рівнянь немає розв'язку')
    break
#визначення прогоночних коефіцієнтів на прямому ході
sig_ma.append(-c[i]/(b[i]+(a[i]*sig_ma[i-1])))
tet_ta.append((d[i]-(a[i]*tet_ta[i-1]))/(b[i]+(a[i]*sig_ma[i-1])))
#визначення розв'язку диф. рівняння на зворотньому ході
y=[0]*(n+1)
y[0]=A; y[n]=B
for i in range(n-1,0,-1):
    y[i]=(sig_ma[i]*y[i+1])+tet_ta[i]
x.append(x_b)
print('x(i)=',x)
print('y(i)=',y)
#будуємо графіки рішень диф. рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
plt.plot(x, y)
plt.plot(x_mtr, (0.5*(np.log(x_mtr)**2)))
plt.legend(['Метод Рунге-Кутта', 'f(x)=0.5*(ln(x))^2'],loc=1)
plt.grid(True)
plt.xlim([1, 1.4])
plt.ylim([0, 0.06])
plt.show()

```

### 3.4 Чисельні методи розв'язання звичайних диференціальних рівнянь для «жорстких» задач

Існують ЗДР, для яких важко отримати задовільний розв'язок задач із використанням вищеописаних чисельних методів. Визначення таких задач пов'язано із поняттям сталої часу диференціального рівняння, яке вводиться стосовно аналітичного розв'язку. Для рівнянь першого порядку – це проміжок часу, коли змінна частина розв'язку убиває в  $e$  разів. Рівняння порядку  $n$  має відповідно  $n$  сталих часу; якщо будь-які дві з них сильно (на практиці в сто і більше разів) відрізняються за величиною або будь-яка з них досить мала порівняно з інтервалом часу, на якому виконується пошук розв'язку, тоді задача називається «жорсткою» і її практично неможливо розв'язати звичайними методами. Коефіцієнти в таких рівняннях відрізняються між собою на декілька порядків.

Доцільно розглянути «жорстку» систему на прикладі розв'язання ЗДР:

$$\begin{cases} \frac{du}{dx} = \lambda u, & x \in (x_0; X]; \\ u(x_0) = u_0. \end{cases} \quad (3.82)$$

Аналітичний розв'язок ЗДР (3.82) подається виразом  $u(t) = u_0 e^{\lambda(x-x_0)}$  на рисунку 3.18.

Якщо припустити, що у вхідних даних допущено помилку  $\delta u$ , то точний розв'язок з часом зміниться:

$$u(x) = (u_0 + \delta u) e^{\lambda(x-x_0)} = u_0 e^{\lambda(x-x_0)} + \delta u e^{\lambda(x-x_0)}. \quad (3.83)$$

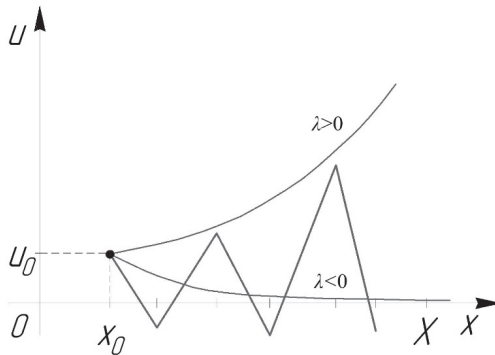


Рисунок 3.18 – Схема аналітичного розв’язку ЗДР за різних значень  $\lambda$

Із рівняння (3.83) за  $\lambda > 0$  помилка буде тільки зростати, в такому випадку ця задача є погано обумовленою. За  $\lambda < 0$  помилка завжди убиває, тому в такому випадку задача є стійкою відносно помилок вхідних даних (див. рис. 3.18). Проте, тільки за умови, коли  $\lambda < 0$  виникають проблеми під час розв’язання задачі чисельними методами. Під час розв’язання «жорстких» задач звичайними чисельними методами крок інтегрування має бути досить малим, щоб можна було враховувати приріст найбільш швидкозмінних складових розв’язку навіть після того, як їх внесок стане практично не помітним. Але зменшення кроку призводить до збільшення витрат машиночасу комп’ютерних систем, накопиченню помилок. Причому навіть на гладкій ділянці розв’язку збільшення кроку призводить до приросту похибок округлення й дискретизації.

Найбільш простим апаратом розв’язання «жорстких» задач є неявний метод Ейлера, що має перший порядок точності, де розв’язок визначається із такого ітераційного рівняння:

$$y_{n+1} = y_n + hf(y_{n+1}, x_{n+1}). \quad (3.84)$$

Алгоритм неявного метода Ейлера для розв’язання «жорстких» задач подано на рисунку 3.19.

«Жорсткі» задачі часто зустрічаються в теорії автоматичного керування, наприклад під час аналізування перехідних процесів у системі, що містить ланки високих порядків, коефіцієнти яких значно відрізняються між собою. Також прикладом «жорстких» систем є диференціальні рівняння, що описують керований рух робота-маніпулятора, оскільки перехідні процеси в системі керування приводом затухають швидше, ніж перехідні процеси в механічній частині робота.

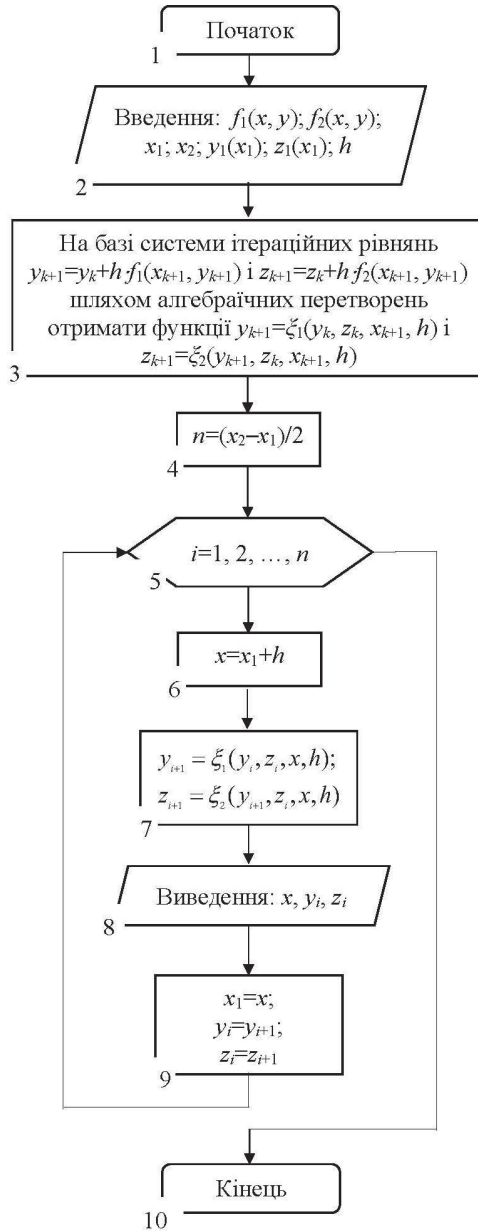


Рисунок 3.19 – Схема алгоритму неявного метода Ейлера для систем ЗДР

**Приклад 3.9.** Розв'язати чисельним методом ЗДР для «жорсткої» задачі математичної моделі, що описує поведінку концентрації хімічних речовин у суміші, в якій відбувається екзотермічна хімічна реакція:

$$\begin{cases} \frac{du}{dt} = a_1 u(t) + b_1 v(t); \\ \frac{dv}{dt} = -a_2 u(t) - b_2 v(t), \end{cases} \quad (3.85)$$

яка задовольняє початкові умови за  $u(0) = v(0) = 1,0$  моль/л, де  $a_1 = 998 \text{ c}^{-1}$ ,  $a_2 = 999 \text{ c}^{-1}$ ,  $b_1 = 1998 \text{ c}^{-1}$ ,  $b_2 = 1999 \text{ c}^{-1}$  – параметри, що визначають кінетику протікання хімічної реакції;  $u(t)$ ,  $v(t)$  – концентрація вихідної хімічної речовини та кінцевого продукту, відповідно.

Порівняти результати чисельного розв'язку із аналітичним розв'язком системи ЗДР «жорсткої» задачі (3.84).

*Розв'язання:*

Оскільки система ЗДР (3.85) відноситься до «жорсткого» типу задач, застосуємо відповідно неявний метод Ейлера на основі ітераційної формули (3.84), а саме:

$$\begin{cases} \frac{du}{dt} = a_1 u(t) + b_1 v(t); \\ \frac{dv}{dt} = -a_2 u(t) - b_2 v(t); \end{cases} \rightarrow \begin{cases} u_{n+1} = u_n + h(a_1 u_{n+1} + b_1 v_{n+1}); \\ v_{n+1} = v_n + h(-a_2 u_{n+1} - b_2 v_{n+1}), \end{cases} \quad (3.86)$$

де  $h$  – крок інтегрування за часом протікання хімічної реакції.

Систему ітераційних рівнянь (3.86) можна розв'язати за допомогою методу Ньютона (див. розд. 2) або можна виразити параметри  $u_{n+1}$  і  $v_{n+1}$ :

$$\begin{cases} v_{n+1} = \frac{v_n - u_n \lambda h a_2}{1 + a_2 b_1 h^2 \lambda + b_2 h}; \\ u_{n+1} = u_n \lambda h + v_{n+1} \lambda h b_1, \end{cases} \quad (3.87)$$

де  $\lambda = 1 / (1 - a_1 h)$ .

Для розв'язання цієї задачі вибирається відрізок часу  $[0; 0,08]$ , який ділиться на відповідну кількість частин із кроком  $h = 5,0 \cdot 10^{-6}$  с. Тоді значення  $v_1, v_2, \dots, v_n$  та  $u_1, u_2, \dots, u_n$  будуть визначатись неявним методом Ейлера за формулою (3.87), а саме:

$$\left\{ \begin{aligned} v_1 &= \frac{v_0 - (u_0 \lambda h a_2)}{1 + (a_2 b_1 h^2 \lambda) + (b_2 h)} = \\ &= \frac{1,0 - (1,0 \cdot 1,01 \cdot 5,0 \cdot 10^{-6} \cdot 999,0)}{1 + (999,0 \cdot 1998,0 \cdot (5,0 \cdot 10^{-6})^2 \cdot 1,01) + (1999,0 \cdot 5,0 \cdot 10^{-6})} = 0,985; \\ u_1 &= u_0 \lambda h + v_1 \lambda h b_1 = 1,0 \cdot 1,01 \cdot 5,0 \cdot 10^{-6} + 0,985 \cdot 1,01 \cdot 5,0 \cdot 10^{-6} \cdot 1998,0 = 1,015; \end{aligned} \right.$$

$$\left\{ \begin{aligned} v_2 &= \frac{v_1 - (u_1 \lambda h a_2)}{1 + (a_2 b_1 h^2 \lambda) + (b_2 h)} = \\ &= \frac{0,985 - (1,015 \cdot 1,01 \cdot 5,0 \cdot 10^{-6} \cdot 999,0)}{1 + (999,0 \cdot 1998,0 \cdot (5,0 \cdot 10^{-6})^2 \cdot 1,01) + (1999,0 \cdot 5,0 \cdot 10^{-6})} = 0,970; \\ u_2 &= u_1 \lambda h + v_2 \lambda h b_1 = 1,015 \cdot 1,01 \cdot 5,0 \cdot 10^{-6} + 0,970 \cdot 1,01 \cdot 5,0 \cdot 10^{-6} \cdot 1998,0 = 1,029; \end{aligned} \right.$$

..... ,

де  $\lambda = \frac{1}{1 - a_1 h} = \frac{1}{1 - 998,0 \cdot 5,0 \cdot 10^{-6}} = 1,01$ .

Також для порівняння було отримано аналітичний розв'язок системи ЗДР «жорсткої» задачі для  $u(0)=v(0)=1,0$ :

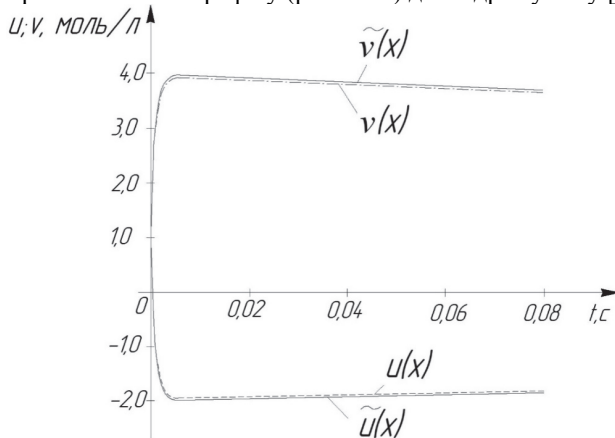
$$\begin{cases} \tilde{u}(t) = 4e^{-t} - 3e^{-1000t}, \\ \tilde{v}(t) = -2e^{-t} + 3e^{-1000t}. \end{cases} \quad (3.88)$$

Чисельні результати обчислення розв'язку системи ЗДР «жорсткої» задачі (3.85) для десяти кроків інтегрування подано в таблиці 3.8.

Таблиця 3.8 – Результати обчислення розв'язку диференціального рівняння

$n$	$t_n \cdot 10^{-6}$ , с	$v(t)$	$u(t)$	$\tilde{v}(t)$	$\tilde{u}(t)$
0	0,0	1,00000	1,00000	1,00000	1,00000
1	0,5	0,98508	1,01475	0,98505	1,01494
2	1,0	0,97024	1,02944	0,97017	1,02981
3	1,5	0,95548	1,04405	0,95537	1,04460
4	2,0	0,94079	1,05858	0,94064	1,05932
5	2,5	0,92617	1,07305	0,92598	1,07397
6	3,0	0,91163	1,08744	0,91140	1,08854
7	3,5	0,89715	1,10175	0,89689	1,10304
8	4,0	0,88276	1,11600	0,88244	1,11747
9	4,5	0,86843	1,13017	0,86808	1,13183
10	5,0	0,85418	1,14427	0,85379	1,14611

Також для візуального порівняння зручно подати результати чисельного розв'язання на графіку (рис. 3.20) для відрізка часу  $[0; 0,08]$ .



$u(x), v(x)$  – неявний метод Ейлера;  $\tilde{u}(x), \tilde{v}(x)$  – точний розв'язок

Рисунок 3.20 – Діаграма результатів чисельного розв'язку системи ЗДР «жорсткої задачі»

Із графіка (рис. 3.20) видно, що після незначного проміжку часу  $t=0,004$  с, розв'язок дуже близький до функцій:

$$\begin{cases} u \cong 4e^{-t}, \\ v \cong -2e^{-t}. \end{cases}$$

Із отриманих результатів, наведених у таблиці 3.8, видно, що похибки, допущені в процесі визначення розв'язку, зростають до кінця таблиці, зокрема в точках  $x_5 = 2,5 \cdot 10^{-6}$  с і  $x_{10} = 5,0 \cdot 10^{-6}$  с:

– відносна похибка шляхом порівняння між значеннями аналітичного розв'язку і неявним методом Ейлера

$$\mathcal{E}_{x_5}^u = \left| \frac{\tilde{u}_{x_5} - u_{x_5}}{\tilde{u}_{x_5}} \right| \cdot 100\% = \left| \frac{1,07397 - 1,07305}{1,07397} \right| \cdot 100\% = 0,09\%,$$

$$\mathcal{E}_{x_5}^v = \left| \frac{\tilde{v}_{x_5} - v_{x_5}}{\tilde{v}_{x_5}} \right| \cdot 100\% = \left| \frac{0,92617 - 0,92598}{0,92617} \right| \cdot 100\% = 0,02\%,$$

$$\mathcal{E}_{x_{10}}^u = \left| \frac{\tilde{u}_{x_{10}} - u_{x_{10}}}{\tilde{u}_{x_{10}}} \right| \cdot 100\% = \left| \frac{1,14611 - 1,14427}{1,14611} \right| \cdot 100\% = 0,16\%,$$

$$\mathcal{E}_{x_{10}}^v = \left| \frac{\tilde{v}_{x_{10}} - v_{x_{10}}}{\tilde{v}_{x_{10}}} \right| \cdot 100\% = \left| \frac{0,85379 - 0,85418}{0,85379} \right| \cdot 100\% = 0,05\%.$$



Розглянемо реалізацію неявного методу Ейлера для системи ЗДР «жорсткої» задачі мовою програмування PYTHON:

```

#підключення обчислювальних бібліотек
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
#введення кроку інтегрування
h_0=float(input())
#введення початкового значення аргументу інтегрування (загального проміжку
# інтегрування) x0
a=int(input())
#введення кінцевого значення загального проміжку інтегрування
b=float(input())
#введення початкового значення аргументу функції диф. рівняння v0
v_0=float(input())
#введення початкового значення аргументу функції диф. рівняння u0
u_0=float(input())
#введення допустимої похибки обчислення
e_0=float(input())
lambda_a=1/(1-(998*h_0))
#зادання 1-ої ітераційної формули системи диф. рівняння
def f_1(u,v,h):
    return (v-(u*lambda_a*h*999))/(1+(999*1998*(h**2)*lambda_a)+(1999*h))
#задання 2-ої ітераційної формули системи диф. рівняння
def f_2(u,v,h):
    return (u*lambda_a)+(f_1(u,v,h)*lambda_a*h*1998)
#обчислення ітераційних формул неявним методом Ейлера
def euler_impl(a,b,u_0,v_0,h):
    v_mas=[v_0]; x=a; x_mas=[x]; u_mas=[u_0]
    u=u_0; v=v_0; x_mtr=[x]; x_mtr=np.array(x)
    while x<=b:
        x=x+h
        v=f_1(u,v,h)
        u=f_2(u,v,h)
        v_mas.append(v)
        u_mas.append(u)
        x_mas.append(x)
        x_mtr=np.append(x_mtr, x)
    return x_mas, v_mas, u_mas
#будуємо графіки рішень диф. рівняння
plt.figure(figsize=(7, 7))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
plt.plot(euler_impl(a,b,u_0,v_0,h_0)[0], euler_impl(a,b,u_0,v_0,h_0)[1])
plt.plot(euler_impl(a,b,u_0,v_0,h_0)[0], euler_impl(a,b,u_0,v_0,h_0)[2])
plt.plot(euler_impl(a,b,u_0,v_0,h_0)[3],
        (4*np.exp(-euler_impl(a,b,u_0,v_0,h_0)[3]))-
        (3*np.exp(-euler_impl(a,b,u_0,v_0,h_0)[3]*1000)))
plt.plot(euler_impl(a,b,u_0,v_0,h_0)[3],
        (-2*np.exp(-euler_impl(a,b,u_0,v_0,h_0)[3]))+
        (3*np.exp(-euler_impl(a,b,u_0,v_0,h_0)[3]*1000)))
plt.legend(['Неявний метод Ейлера для v(x)', 'Неявний метод Ейлера для u(x)',
        'u(x)=4*e^(-x)-3*e^(-x*1000)',
        'v(x)=-2*e^(-x)+3*e^(-x*1000)'], loc=1)
plt.grid(True)
plt.xlim([0, 0.015])
plt.ylim([-2.5, 4.5])
plt.show()

```

## **Висновки щодо застосування методів чисельного розв'язання звичайних диференціальних рівнянь**

Чисельні методи розв'язання ЗДР не дозволяють знайти загального розв'язку; вони можуть дати тільки деякий частинний розв'язок. Проте ці методи можуть бути застосовані до широкого класу диференціальних рівнянь й усіх типів задач для них. Чисельні методи можуть бути застосовані тільки до коректно поставлених (або регульованих) задач. Проте потрібно відмітити, що для успішного застосування чисельних методів формальне виконання умов коректності може бути недостатнім. Необхідно, щоб задача була добре обумовленою, тобто малі зміни початкових умов привели б до достатньо малої зміни інтегральних кривих. Якщо ця умова не виконується, тобто задача погано обумовлена (слабо стійка), тоді невеликі зміни початкових умов або еквівалентні цим змінам невеликі похибки чисельного методу можуть сильно спотворювати результат. Для розв'язання звичайних диференціальних рівнянь доцільно використовувати методи: Ейлера, Рунге-Кутта четвертого порядку, Адамса («прогнозування та корекції»), «стрілянини», а також кінцевих різниць. Перед застосуванням цих методів виконується ідентифікація виду задачі на основі типу ЗДР (задача Коші, крайова задача або «жорстка задача») й застосовуються відповідні методи. Для розв'язання задач Коші існують такі чисельні методи двох видів: однокрокові методи, в яких для знаходження наступної точки на кривій необхідна інформація лише про один попередній крок (методи Ейлера і Рунге-Кутта четвертого порядку); багатокрокові методи, в яких для пошуку наступної точки кривої необхідна інформація більш ніж про одну з попередніх точок (методи «прогнозування та корекції» або метод Адамса). Під час порівняння ефективності однокрокових і багатокрокових методів виділяються такі особливості: багатокрокові методи потребують великого обсягу пам'яті обчислювальних комп'ютерних систем, оскільки оперують великою кількістю вихідних даних; під час використання багатокрокових методів існує можливість оцінення похибки на кроці, тому величина кроку вибирається оптимальною, що порівняно з однокроковими дає певний запас, який знижує швидкодію обчислень; за однакової точності однокрокові методи потребують меншого обсягу обчислень (наприклад, застосовуючи метод Рунге-Кутта четвертого порядку, необхідно обчислити чотири значення функції на кожному кроці, а для забезпечення збіжності методу «прогнозування та корекції» того самого порядку точності – два); однокрокові методи, на відміну від багатокрокових, дозволяють почати розв'язання задачі (самостартування) і легко змінювати крок у процесі розв'язання. Якщо задача Коші дуже важка, тоді зазвичай перевага надається методу «прогнозування та корекції» (Адамса), який до того ж має більш високу швидкодію. Початок розв'язання задачі в такому разі виконується за допомогою однокрокових методів, а саме Ейлера або Рунге-

Кутта четвертого порядку. Якщо для обчислення чергового значення  $u_i$  необхідно більше двох ітерацій або якщо помилка зрізу досить велика, то необхідно зменшити значення обчислювального кроку  $h$ . З іншого боку, у разі дуже малої похибки зрізу можна збільшити крок, що дозволить підвищити швидкодню, але в цьому випадку увесь процес розв'язання виконується спочатку. Іноді на практиці необхідно мінімізувати час підготовки задачі до розв'язання. Тоді доцільно використовувати однокрокові методи. Для розв'язання крайових задач ЗДР використовуються метод «стрілянини» або різницеві методи. У випадку нелінійних диференціальних рівнянь перевага надається різницеvim методам. Під час розв'язання «жорстких задач» найпростішим шляхом є застосування «неявного» методу Ейлера, де крок має бути достатньо малим, щоб можна було врахувати приріст найбільш швидкозмінюваних складових розв'язку навіть після того, як їхній внесок стане практично непомітним. Загалом для ефективного розв'язання задач велике значення має досвід, інтуїція і кваліфікація дослідника як в процесі постановки задачі, так і в процесі вибору методу, розробки алгоритму й програмних засобів розв'язання засобами комп'ютерних систем.

### Контрольні запитання та завдання

1. Наведіть приклади застосування диференціального числення в різних напрямках наукового дослідження.
2. Сформулюйте узагальнену постановку задачі для звичайних диференціальних рівнянь.
3. Сформулюйте задачу Коші і крайову задачу. У чому відмінність цих задач?
4. Які види чисельних методів для розв'язання ЗДР існують? Дайте їм порівняльну характеристику.
5. У чому відмінність між звичайним методом Ейлера й уточненим для чисельного розв'язання задачі Коші ЗДР?
6. Дайте геометричну інтерпретацію метода Ейлера.
7. У математичній моделі задачі балістики, а саме вертикального падіння тіла масою  $m$ :

$$\frac{dv}{dt} = -\frac{\alpha}{m}v + g, \quad (3.89)$$

на яке діє сила в'язкого тертя  $F_{mp}$ , пропорційна величині швидкості ( $F_{mp} = -\alpha v$ , де  $\alpha$  – коефіцієнт в'язкого тертя), за допомогою звичайного чисельного методу Ейлера і уточненого визначить значення його швидкості залежно від часу. Порівняйте отримані результати чисельного дослідження

із аналітичним розв'язком  $v(t) = \frac{mg}{\alpha} \left( 1 - e^{-\frac{\alpha}{m}t} \right)$  за початкової умови  $v(0) = 0$  м/с на проміжку часу  $t \in [0; 10,0]$  секунд.

Таблиця 3.9 – Вихідні дані до завдання

Варіант	Маса тіла ( $m$ , кг)	Коефіцієнт в'язкого тертя ( $\alpha$ , Н·с/м)
1	2,0	0,00035
2	3,2	0,00027
3	3,5	0,00010
4	4,6	0,00004

8. Найпростіша математична модель битви двох протидіючих армій описується рівняннями бою Ланчестера:

$$\begin{cases} \frac{dx(t)}{dt} = -by(t); \\ \frac{dy(t)}{dt} = -ax(t), \end{cases} \quad (3.90)$$

де  $a, b$  – ефективність зброї армій  $x$  та  $y$ , відповідно, тобто кожна бойова одиниця армії  $x$  знищує за одиницю часу  $a$  солдат армії  $y$  і навпаки. За допомогою чисельного методу Ейлера визначіть значення зміни чисельності протидіючих армій під час бою на проміжку часу  $t \in [0; 100,0]$  секунд. Як треба змінити початкову чисельність армій, щоб змінився результат битви?

Таблиця 3.10 – Вихідні дані до завдання

Варіант	Ефективність зброї армії $x$ ( $a$ , од./год.)	Ефективність зброї армії $y$ ( $b$ , од./год.)	Початкова чисельність армії $x$ ( $x(0)$ , од.)	Початкова чисельність армії $y$ ( $y(0)$ , од.)
5	2,0	2,8	100,0	180,0
6	3,2	4,1	220,0	280,0
7	3,5	2,4	300,0	240,0
8	4,6	3,1	410,0	390,0

9. Розкрийте суть чисельного методу Рунге-Кутта четвертого порядку для розв'язання задачі Коші ЗДР.

10. Дайте геометричну інтерпретацію методу Рунге-Кутта четвертого порядку.

11. Як визначається похибка обчислення чисельним методом Рунге-Кутта четвертого порядку?

12. За допомогою чисельного метода Рунге-Кутта четвертого порядку для найпростішої математичної моделі епідемії:

$$\frac{dx}{dt} = \alpha \cdot x(t) \cdot [N + 1 - x(t)], \quad (3.91)$$

де  $N$  – кількість людей в дослідній групі, а  $\alpha$  – коефіцієнт пропорційності. Визначіть значення  $x(t)$  – чисельності захворюваності людей залежно від одиниць часу із заданою точністю похибки зрізу  $R_n^h = 0,001$ . Порівняйте отримані результати чисельного дослідження із аналітичним розв'язком ЗДР (3.91) на проміжку часу  $t \in [0; 150,0]$  секунд, а саме:

$$x(t) = \frac{N + 1}{N \cdot e^{-\alpha(N+1)t} + 1}.$$

Таблиця 3.11 – Вихідні дані до завдання

Варіант	Кількість людей у дослідній групі ( $N$ , чол.)	Коефіцієнт пропорційності ( $\alpha$ , с <sup>-2</sup> )
9	100,0	0,0010
10	80,2	0,0012
11	50,5	0,0022
12	60,6	0,0035

13. За допомогою чисельного метода Рунге-Кутта четвертого порядку для математичної моделі задачі балістики, а саме вертикального падіння тіла масою  $m$ :

$$\frac{d^2x}{dt^2} = \left(-\frac{\alpha}{m}\right) \frac{dx}{dt} + g, \quad (3.92)$$

на яке діє сила в'язкого тертя  $F_{mp}$ , пропорційна величині швидкості ( $F_{mp} = -\alpha v$ , де  $\alpha$  – коефіцієнт в'язкого тертя) визначіть значення його переміщення залежно від часу  $t$  із заданою точністю  $R_n^h = 0,001$  за початкової умови  $v(0) = 0$  м/с і висоти падіння  $h$ .

Таблиця 3.12 – Вихідні дані до завдання

Варіант	Маса тіла ( $m$ , кг)	Коефіцієнт в'язкого тертя ( $\alpha$ , Н⋅с/м)	Початкова висота падіння ( $h$ , м)
13	2,0	0,00035	120,0
14	3,2	0,00026	110,0
15	3,5	0,00010	100,0
16	4,6	0,00004	150,0

14. Розкрийте суть чисельного метода «прогнозування та корекції» (Адамса) для розв'язання задачі Коші ЗДР.

15. Як визначається похибка обчислення чисельним методом «прогнозування та корекції» (Адамса) для розв'язання задачі Коші ЗДР?

16. Що таке властивість «самостартування»? Які методи розв'язання ЗДР мають її?

17. Який порядок точності чисельних методів Ейлера, Рунге-Кутта, Адамса для розв'язання задачі Коші ЗДР?

18. За допомогою чисельного метода «прогнозування та корекції» (Адамса) для математичної моделі обмеженого зростання чисельності популяції (логістична модель Ферхюльста):

$$\frac{dN}{dt} = rN - kN^2, \quad (3.93)$$

де  $r$  – середня швидкість зростання популяції,  $k$  – коефіцієнт зустрічі конкуруючих осіб, визначіть значення чисельності популяції  $N$  залежно від часу  $t$  із заданою точністю похибки зрізу  $R_n^h = 0,0001$  на проміжку одиниць часу  $t \in [0; 5,0]$ . Визначіть найменшу чисельність популяції за наявності конкурентної боротьби.

Таблиця 3.13 – Вихідні дані до завдання

Варіант	Початкова чисельність популяції ( $N$ , од.)	Середня швидкість зростання популяції ( $r$ , 1/од. часу)	Коефіцієнт зустрічі конкуруючих осіб ( $k$ , ос./од. часу)
17	1000,0	3,23	2,15
18	900,0	3,80	2,80
19	800,0	4,10	5,10
20	700,0	2,50	1,95

19. У математичній моделі вільних коливань підпружиненого тіла масою  $m$  за лінійного опору середовища ( $F_{mp} = -\alpha(dx/dt)$ , де  $\alpha$  – коефіцієнт в'язкого тертя):

$$m \frac{d^2x}{dt^2} = -kx - \alpha \frac{dx}{dt}, \quad (3.94)$$

де  $k$  – коефіцієнт жорсткості пружного елемента, за допомогою чисельного методу «прогнозування та корекції» (Адамса) визначіть значення переміщення  $x$  залежно від часу  $t$  із заданою точністю  $\Delta = 0,0001$  за початкової умови  $x(0) = 0,02$  м на проміжку часу  $t \in [0; 5,0]$  секунд.

Таблиця 3.14 – Вихідні дані до завдання

Варіант	Маса тіла ( $m$ , кг)	Коефіцієнт в'язкого тертя ( $\alpha$ , Н·с/м)	Коефіцієнт жорсткості пружного елемента ( $k$ , Н/м)
21	10,0	45,0	$5,0 \cdot 10^3$
22	12,0	42,0	$8,0 \cdot 10^3$
23	8,0	52,0	$11,0 \cdot 10^3$
24	9,0	38,0	$13,0 \cdot 10^3$

20. Математична модель прогнозування кліматичних умов в основі якої покладено модель Лоренца (модель фізичного процесу двовимірної теплової конвекції):

$$\begin{cases} \frac{dx(t)}{dt} = \sigma(y(t) - x(t)); \\ \frac{dy(t)}{dt} = Rx(t) - y(t) - x(t)z(t); \\ \frac{dz(t)}{dt} = x(t)y(t) - bz(t), \end{cases} \quad (3.95)$$

де  $x(t)$  – інтенсивність конвекції;  $y(t)$  – різниця між температурами висхідними і низхідними потоками повітря;  $z(t)$  – відхилення вертикального температурного профілю від лінійної залежності;  $R$  – нормоване число Релея (відображає поведінку потоку повітря під впливом градієнта температури);  $\sigma$  – число Прандтля (критерій подібності теплових процесів в рідини і газах);  $b$  – геометричні параметри конвективної розрахункової комірки. За допомогою будь-якого однокрокового або багатокрокового чисельного методу побудуйте діаграму залежності  $x(y, z)$  на проміжку часу  $t \in [0; 100]$  секунд із кроком обчислення  $t=0,01$  с. Як початкові параметри виберіть такі значення:  $x(0)=0,0$ ;  $y(0)=1,0$ ;  $z(0)=1,05$ .

Таблиця 3.15 – Вихідні дані до завдання

Варіант	Нормоване число Релея ( $R$ )	Число Прандтля ( $\sigma$ )	Геометричні параметри конвективної розрахункової комірки ( $b$ )
25	28,0	10,0	10/3
26	30,0	12,0	11/3
27	31,0	13,0	13/5
28	27,0	9,0	9/2
29	29,0	11,0	14/3
30	33,0	14,0	15/4

21. У чому відмінність початкових і крайових умов постановки задачі під час розв'язання ЗДР?

22. Розкрийте суть чисельного метода «стрілянини» для розв'язання крайової задачі ЗДР.

23. Задачу про брахістохрон (оптимальний шлях, який проходить тіло за мінімальний час під дією сили земного тяжіння) подано математичною моделлю:

$$2y \frac{d^2y}{dx^2} + 1 + \left( \frac{dy}{dx} \right)^2 = 0, \quad (3.96)$$

де  $y$  – координата шляху по осі  $x$ ;  $0 < x < L$  см – координати шляху по осі  $x$ ;  $\frac{dy(0)}{dx} = 0$ ;  $0 < y < H$  см – координати шляху по осі  $y$ . Чисельним методом «стрілянини» визначіть значення координат у оптимальної кривої шляху залежно від координати  $x$  із точністю  $\varepsilon = 0,0001$ .

Таблиця 3.16 – Вихідні дані до завдання

Варіант	Початкове положення тіла по вертикальній осі $y$ ( $H$ , см)	Початкове положення тіла по горизонтальній осі $x$ ( $L$ , см)
1	35,0	45,0
2	30,0	20,0
3	40,0	50,0
4	25,0	38,0

24. Розкрийте суть чисельного методу кінцевих різниць для розв'язання крайової задачі ЗДР. Із яких етапів складається розв'язок задачі методом кінцевих різниць?

25. Запишіть загальну формулу для апроксимації похідних у деякій точці?

26. Як визначаються різницеві відношення для апроксимації похідної із першим і другим порядком?

27. Задачу стискання в'язкопластичного стержня за повздовжнього удару подано математичною моделлю:

$$\frac{d^2\delta}{dx^2} + ax \left( \frac{d\delta}{dx} \right)^b = 0, \quad (3.97)$$

де  $\delta$  – значення стиску стержня по осі  $x$ ;  $0 < x < L$  м – координата довжини стержня по осі  $x$ ;  $b=1,0$  – безрозмірний параметр ударної характеристики системи;  $a$  – жорсткість стержня за розтягу і стискання. Чисельним методом кінцевих різниць визначіть значення локального стиску стержня залежно від повздовжньої координати  $x$  за таких крайових умов  $\delta(0)=0$  мм,  $\delta(L) = \Delta$  мм.



Таблиця 3.17– Вихідні дані до завдання

Варіант	Довжина стержня ( $L$ , м)	Жорсткість стержня при розтягу і стисканні ( $a$ )	Деформація краю стержня в місці удару ( $\Delta$ , мм)
5	1,0	0,0034	1,2
6	1,5	0,0038	1,5
7	1,8	0,0023	2,1
8	2,5	0,0048	2,8

28. Які диференціальні рівняння називаються «жорсткими»? У чому полягають особливості їх розв’язання?

29. Розкрийте суть чисельного неявного методу Ейлера для розв’язання «жорстких» задач ЗДР?

30. Задачу кінетики хімічної реакції подано математичною моделлю:

$$\begin{cases} \frac{dx}{dt} = -0,5x + 30y; \\ \frac{dy}{dt} = -30y, \end{cases} \quad (3.98)$$

де  $x$  – поточне значення концентрації хімічної речовини  $A$ ;  $y$  – поточне значення концентрації хімічної речовини  $B$ . За допомогою неявного чисельного методу Ейлера для цієї «жорсткої задачі» системи ЗДР (3.98) визначить значення зміни концентрації хімічних речовин  $x(t)$  та  $y(t)$  на проміжку часу  $t \in [0; 2,0]$  секунд із початковими даними  $x(0) = X_0$  та  $y(0) = Y_0$ .

Таблиця 3.18 – Вихідні дані до завдання

Варіант	Початкове значення концентрації хімічної речовини $A$ ( $X_0$ , моль/л)	Початкове значення концентрації хімічної речовини $B$ ( $Y_0$ , моль/л)
9	1,6	2,8
10	5,5	3,5
11	3,8	2,9
12	4,5	5,1

## РОЗДІЛ 4 ДИФЕРЕНЦІАЛЬНІ РІВНЯННЯ МАТЕМАТИЧНОЇ ФІЗИКИ

Однією із характерних рис сучасних досліджень є математизація фізичного пізнання, інтенсивне застосування методів математичного моделювання в нетрадиційних і навіть «описових» науках (екологія, медицина та ін.). Практика сьогодення потребує від науковця вирішення різного роду проблем, повне дослідження яких може бути проведене в більшості випадків тільки чисельним шляхом або за допомогою ретельно поставленого фізичного експерименту. Тому створення загальних чисельних методів (алгоритмів) для розв'язання задач математичної фізики і нелінійної механіки є актуальним.

Предметом математичної фізики є побудова і дослідження математичних моделей фізичних явищ. Задачі класичної математичної фізики зводяться до крайових задач для рівнянь у частинних похідних. Основними засобами дослідження таких задач є теорія диференціальних рівнянь разом із теорією функцій, варіаційним численням, функціональним аналізом, теорією ймовірності та обчислювальної математики.

Якщо позначити через  $D$  область  $n$ -вимірного простору  $R^n$  точок  $x = (x_1, x_2, \dots, x_n)$ ;  $x_1, x_2, \dots, x_n$ ;  $n \geq 2$  – декартові координати точки  $x$ , тоді рівняння виду:

$$F\left(x, u, \frac{\partial u}{\partial x_1}, \frac{\partial u}{\partial x_1}, \dots, \frac{\partial^k u}{\partial x_1^i \dots \partial x_n^j}, \frac{\partial u}{\partial x_n^m}\right) = 0 \quad (x \in D; \sum_{j=1}^n i_j = k; k=0, 1, \dots, m) \quad (4.1)$$

називаються диференціальним рівнянням у частинних похідних порядку  $m$

відносно невідомої функції  $u=u(x)$ , де  $F = F\left(x, u, \frac{\partial u}{\partial x_1}, \dots\right)$  – задана дійсна

функція точок  $x \in D$ , невідомої функції  $u$  і її частинних похідних. Ліва частина рівняння (4.1) називається диференціальним оператором із частинними похідними порядку  $m$ .

Дійсна функція  $u=u(x_1, x_2, \dots, x_n)$ , визначена в області  $D$  задання рівняння (4.1), неперервна разом зі своїми частинними похідними, що входять у це рівняння, і обернена його в тотожність, називається класичним (регулярним) розв'язком рівняння (4.1).

Розв'язок рівняння (4.1) в  $n+1$ -вимірному просторі змінних  $x_1, x_2, \dots, x_n, u$  задає деяку гладку поверхню розмірності  $n$ , яка називається інтегральною поверхнею рівняння (4.1).

Багато задач фізики суцільних середовищ зводиться до розв'язання диференціальних рівнянь із частинними похідними. У такому випадку як шукані функції зазвичай є густина, температура, напруження та інші, аргументами яких є координати розглянутої точки простору, а також і самий час.

Зокрема для опису розподілу температури в заданій області простору і її зміни в часі використовується рівняння теплопровідності:

$$\frac{\partial u}{\partial t} - \alpha \left( \frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2} + \dots + \frac{\partial^2 u}{\partial x_n^2} \right) = f(x, t), \quad (4.2)$$

де  $x = (x_1, \dots, x_n)$  – декартові координати,  $f(x, t)$  – функція теплових джерел,  $\alpha$  – коефіцієнт температуропровідності,  $u(x, t)$  – шукана функція температури в точці із координатами  $x$  в момент часу  $t$ . Якщо  $f(x, t) \equiv 0$ , тобто всередині системи відсутні джерела і «стоки» тепла, тоді рівняння теплопровідності (4.2) називається однорідним.

Повна математична постановка задачі нарівні із диференціальними рівняннями містить також деякі додаткові умови. Якщо пошук розв'язку виконується в обмеженій області, то також задаються крайові умови. Тоді задача називається крайовою задачею для рівнянь із частинними похідними.

Якщо однією із незалежних змінних є час  $t$ , то задаються значення шуканих функцій у початковий момент часу  $t=0$  і вони називаються початковими умовами. Зокрема для рівняння теплопровідності (4.2) крайовими і початковими умовами є  $u|_{\Omega} = \psi(t)$  і  $u|_{t=0} = \varphi(x)$ , відповідно, де  $\Omega \in \mathbb{R}^n$  ( $n = 2, 3$ ) – плоска або тривимірна область.

Задача, в якій необхідно розв'язати диференціальне рівняння в частинних похідних за заданих початкових умов, називається задачею Коші. Водночас задача розв'язується в необмеженому просторі, і граничні умови не задаються. Задачі, під час формулювання яких виконується постановка одночасно граничних і початкових умов, називаються нестационарними (або змішаними) крайовими задачами. Розв'язок, який в цьому випадку буде отримано, змінюється протягом часу.

У цьому розділі буде розглянуто тільки коректно поставлені задачі, тобто задачі, розв'язок яких існує і єдиний у деякому класі початкових та граничних умов.

#### 4.1 Класифікація диференціальних рівнянь у частинних похідних

Нехай  $u(x, y)$  – невідома функція двох змінних  $x$  і  $y$ , яку необхідно визначити, тоді достатньо вузький клас задач для рівнянь першого і другого порядків, лінійних відносно похідних, виражається такою формою рівняння:

$$\begin{aligned} A(x, y) \frac{\partial^2 u}{\partial x^2} + 2B(x, y) \frac{\partial^2 u}{\partial x \partial y} + C(x, y) \frac{\partial^2 u}{\partial y^2} + D(x, y) \frac{\partial u}{\partial x} + \\ + E(x, y) \frac{\partial u}{\partial y} + F(x, y) = G, \end{aligned} \quad (4.3)$$

де  $A, B, C, D, E, F$  – функціональні коефіцієнти, які можуть залежати як від аргументів  $x, y$ , так і від функції  $u$ .

Залежно від цього рівняння (4.3) може бути:

а) рівнянням другого порядку в частинних похідних із сталими коефіцієнтами;

б) лінійними, якщо права частина рівняння лінійно залежить від функції  $u$ , а коефіцієнти залежать тільки від  $x, y$ ;

с) квазілінійними, якщо коефіцієнти залежать від  $u$ .

Аналогічно зі звичайними диференціальними рівняннями єдиний розв'язок рівняння можна отримати лише задавши додаткові умови, але оскільки в рівнянні (4.2) присутні дві незалежні змінні  $x$  та  $y$ , умова має задаватися для якої-небудь кривої в площині  $x, y$ . Ця умова може бути накладена на функцію  $u$  або (та) на її похідні та залежати від типу рівняння, яке визначає її вид і характер зміни.

Розрізняють різні види рівнянь залежно від співвідношення між коефіцієнтами:

1) за  $A = B = C = D = F = 0, D \neq 0, E \neq 0$  маємо рівняння переносу

$$\frac{\partial u}{\partial x} + P \frac{\partial u}{\partial y} = G \quad (P=E/D). \quad (4.4)$$

Якщо в рівнянні (4.3) однією із незалежних змінних є час, то це рівняння називається еволюційним.

2) якщо хоча б один із коефіцієнтів  $A = B = C \neq 0$ , то рівняння (4.3) є рівнянням другого порядку. У такому випадку залежно від дискримінанта  $Ds = 4B^2 - 4AC$  рівняння (4.2) може належати до одного із трьох типів: гіперболічного ( $Ds > 0$ ), параболічного ( $Ds = 0$ ), еліптичного ( $Ds < 0$ ).

Рівняння можуть переходити з одного типу в інший залежно від значень відповідних коефіцієнтів.

У випадку, коли коефіцієнти  $A, B, C$  є сталими, рівняння (4.3) має один і той самий тип у всіх точках площини змінних  $x$  і  $y$ . У випадку, якщо коефіцієнти  $A, B, C$  неперервно залежать від  $x$  і  $y$ , множина точок, в яких це рівняння відноситься до гіперболічного (еліптичного) типу, утворює на площині відкриту область, яка називається гіперболічною (еліптичною), а множина точок, в яких рівняння відноситься до параболічного типу, замкненою. Рівняння (4.3) називається змішаним (змішаного типу), якщо в деяких точках площини є гіперболічним, а в деяких – еліптичним. У цьому випадку параболічні точки, як правило, утворюють лінію, яка називається лінією зміни типу або лінією виродження.

Існують два види методів розв'язання рівнянь цього типу: аналітичний (результат виводиться різними математичними перетвореннями), чисельний, за якого отриманий результат відповідає дійсному із заданою точністю, але необхідно багато алгебраїчних обчислень, що потребує використання обчислювальних потужностей комп'ютерних систем.

Цей розділ присвячено чисельним методам, алгоритмам та їх застосуванню для диференціальних рівнянь у частинних похідних другого

порядку, що найбільш часто використовуються в науково-прикладних інженерних задачах.

Приклади деяких диференціальних рівнянь у частинних похідних, які описують різні типи задач, наведено в таблиці 4.1.

Таблиця 4.1 – Диференціальні рівняння в частинних похідних

Тип рівняння	Математична форма	Приклади задач
Лапласа	$\Delta u = 0$	Усталена течія рідини. Стационарні теплові поля
Пуассона	$\Delta u = -k$	Теплопередача із внутрішнім джерелом тепла
Дифузії	$\Delta u = \frac{1}{h^2} \frac{\partial^2 u}{\partial t^2}$	Нестационарна теплопровідність
Хвильове	$\Delta u = \frac{1}{c^2} \frac{\partial^2 u}{\partial t^2}$	Поширення хвиль (звукових, електромагнітних тощо)
Бігармонічне	$\Delta^2 u = F(x, y)$	Деформація пластин

У таблиці 4.1 використано прийняті позначення найбільш поширених операторів:  $\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$  – Лапласа,  $\Delta^2 u = \frac{\partial^4 u}{\partial x^4} + 2 \frac{\partial^4 u}{\partial x^2 \partial y^2} + \frac{\partial^4 u}{\partial y^4}$  – бігармонічний.

Існують два основних методи чисельного розв'язання диференціальних рівнянь у частинних похідних: різницевий метод (метод скінчених різниць) і метод скінчених елементів. У сучасній прикладній математиці обидва методи розглядаються як інтерпретації використання загальної теорії різницевих схем до розв'язання диференціальних рівнянь у частинних похідних.

В основі методу скінчених елементів лежить варіаційне обчислення. Диференціальні рівняння, які описують задачу й відповідні граничні умови, використовуються для постановки варіаційної задачі. У методі скінчених елементів фізична задача замінюється кусково-гладкою моделлю. Цей метод потребує складної постановки задачі, високої кваліфікації й досвіду, неуніверсальний (кожне розв'язання застосовується лише для конкретної задачі). Метод скінчених елементів знайшов широке використання для розв'язання спеціальних задач у теоретичній механіці, гідродинаміці, теорії поля, він складний, потребує серйозної підготовки і знань у конкретній сфері використання.

## 4.2 Метод кінцевих різниць

Апарат різницевих методів (*difference methods*) являє собою ефективний засіб чисельного розв'язання диференціальних рівнянь як звичайних, так і в частинних похідних. У розділі 3.3 було розглянуто основні положення побудови різницевих систем, в основу яких покладено подання незалежного аргументу у вигляді дискретної множини точок, яка називається сіткою. Окрім найпоширенішої прямокутної сітки використовуються полярна, трикутна, скошена та інші (рис. 4.1). Багатовимірні сітки знаходять використання в задачах із частинними похідними за декількома незалежними змінними.

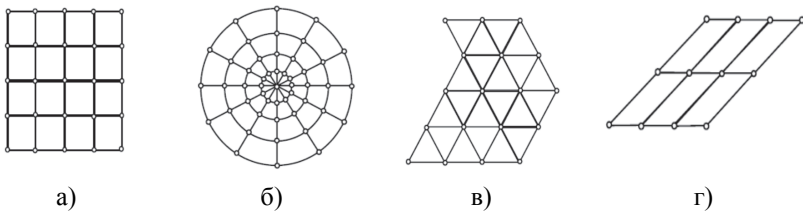


Рисунок 4.1 – Схеми розрахункових сіток:

а) прямокутна; б) полярна; в) трикутна; г) скошена

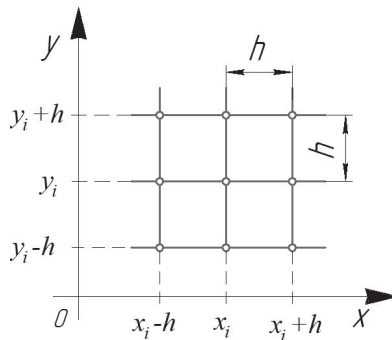


Рисунок 4.2 – Розрахункова схема двовимірної квадратної сітки

Для диференціальних рівнянь другого порядку в частинних похідних найчастіше використовується двовимірна прямокутна сітка (рис. 4.1, а). Центрально-різницеві шаблони, які застосовуються на двовимірній квадратній сітці із кроком  $h$  (рис. 4.2) можуть бути отримані аналогічно одновимірному випадку (індекс  $j$  відноситься до незалежної змінної  $y$ , а індекс  $i$  до  $x$ ).

Для зручності позначення  $u(x_i+h, y_i)$  необхідно замінити на  $u_{i+1,j}$ . Використовуючи такі позначення і виконуючи розкладання в ряд Тейлора, отримуємо вирази для частинних похідних, а саме:

$$\frac{\partial u}{\partial x} \approx \frac{u_{i+1,j} - u_{i-1,j}}{2h} \approx \frac{1}{2h} \left[ \begin{array}{c} (-1) \\ \text{---} \\ 0 \\ \text{---} \\ 1 \end{array} \right]_{i,j};$$

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \approx \frac{1}{h^2} \left[ \begin{array}{c} 1 \\ \text{---} \\ -2 \\ \text{---} \\ 1 \end{array} \right]_{i,j};$$

$$\frac{\partial u}{\partial y} \approx \frac{u_{i,j+1} - u_{i,j-1}}{2h} \approx \frac{1}{2h} \left[ \begin{array}{c} 1 \\ \text{---} \\ 0 \\ \text{---} \\ -1 \end{array} \right]_{i,j}; \quad \frac{\partial^2 u}{\partial y^2} \approx \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} \approx \frac{1}{h^2} \left[ \begin{array}{c} 1 \\ \text{---} \\ -2 \\ \text{---} \\ 1 \end{array} \right]_{i,j};$$

$$\frac{\partial^2 u}{\partial x \partial y} \approx \frac{u_{i+1,j+1} - u_{i-1,j+1} - u_{i+1,j-1} + u_{i-1,j-1}}{4h^2} \approx \frac{1}{4h^2} \left[ \begin{array}{ccc} (-1) & 0 & 1 \\ | & | & | \\ 0 & 0 & 0 \\ | & | & | \\ 1 & 0 & -1 \end{array} \right]_{i,j};$$

$$\frac{\partial^4 u}{\partial x^4} \approx \frac{u_{i-2,j} - 4u_{i-1,j} + 6u_{i,j} - 4u_{i+1,j} + u_{i+2,j}}{h^4} \approx \frac{1}{h^4} \left[ \begin{array}{ccccc} 1 & -4 & 6 & -4 & 1 \end{array} \right]_{i,j};$$

$$\frac{\partial^4 u}{\partial y^4} \approx \frac{u_{i,j+2} - 4u_{i,j+1} + 6u_{i,j} - 4u_{i,j-1} + u_{i,j-2}}{h^4} \approx \frac{1}{h^4} \left[ \begin{array}{c} 1 \\ \text{---} \\ -4 \\ \text{---} \\ 6 \\ \text{---} \\ -4 \\ \text{---} \\ 1 \end{array} \right]_{i,j}.$$

З цих елементів будуються більш складні обчислювальні шаблони для диференціальних рівнянь. Додавання похідних здійснюється суперпозицією відповідних обчислювальних шаблонів. Цим методом конструюються шаблони для  $\Delta u$  і  $\Delta^2 u$ , які мають похибку порядку  $h^2$ :

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \approx \frac{u_{i-1,j} - 4u_{i,j} + u_{i+1,j} + u_{i,j+1} + u_{i,j-1}}{h^2} \approx \frac{1}{h^2} \begin{bmatrix} & & 1 & & \\ & 1 & -4 & 1 & \\ & & |_{i,j} & & \\ & & 1 & & \end{bmatrix};$$

$$\Delta^2 u = \frac{\partial^4 u}{\partial x^4} + 2 \frac{\partial^4 u}{\partial x^2 \partial y^2} + \frac{\partial^4 u}{\partial y^4} \approx \frac{1}{h_4} \begin{bmatrix} & & & & 1 & & & & \\ & & & & 2 & -8 & 2 & & \\ & & & & 1 & -8 & 20 & -8 & 1 & \\ & & & & 2 & -8 & 2 & & \\ & & & & 1 & & & & \end{bmatrix}.$$

Усі наведені обчислювальні шаблони мають похибку другого порядку. Можна побудувати більш точні обчислювальні шаблони, якщо внести у розгляд додаткові вузли. В основі усіх побудованих вище обчислювальних шаблонів покладено центрально-різницеву апроксимацію. Іноді, щоб звести до мінімуму поширення похибок, користуються лівими або правими різницями. Під час використання обчислювальних шаблонів різницевої рівняння (апроксимоване диференціальне рівняння у частинних похідних) може бути нестійким. Різницева схема вважається нестійкою, якщо похибка, із кожним ітераційним кроком, не зменшується. Проблеми нестійкості різницевої схем особливо виникають в еволюційних задачах.

Застосувавши обчислювальний шаблон до кожного із  $n$  вузлів сітки, отримуємо систему з  $n$  рівнянь, яка може бути лінійною, якщо початкове диференціальне рівняння має відповідну структуру. У цьому випадку розв'язання задачі зводиться до розв'язання системи рівнянь вигляду:

$$\begin{bmatrix} \text{матриця} \\ \text{коефіцієнтів} \end{bmatrix} \begin{bmatrix} \text{невідоме значення } u \\ \text{вузлах (вектор - стовпець)} \end{bmatrix} = \begin{bmatrix} \text{вектор - стовпець} \\ \text{вільних членів} \end{bmatrix},$$

яка розв'язується найчастіше ітераційними методами (див. розд. 1).

### 4.3 Розв'язання різних видів диференціальних рівнянь в частинних похідних

Практичні методи й алгоритми розв'язання різних видів диференціальних рівнянь у частинних похідних мають певні особливості і потребують окремого розгляду на прикладі найпоширеніших задач.



## Розв'язання еліптичних рівнянь

До еліптичних рівнянь зводиться багато різних фізичних задач: розрахунок напружень, що виникають під час пружного кручення довгого циліндричного стержня; розподіл електричних напружень на провідниковій площині; задача про стаціонарні потоки тепла у плоскому тілі та інші.

Більшість еліптичних рівнянь описується рівнянням Пуассона або його частинним випадком – рівнянням Лапласа.

Одною із відомих задач є класична задача Діріхле для рівняння Лапласа в прямокутній області. Необхідно визначити неперервну функцію  $u(x, y)$ , яка

задовольняє рівняння Лапласа  $\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$  усередині прямокутної області  $\Omega = \{(x; y) \mid 0 \leq x \leq a, 0 \leq y \leq b\}$ , а також набуває на границі області заданих значень:  $x = 0, u(0, y) = u_1(y)$ ;  $x = a, u(a, y) = u_2(y)$ ;  $y = 0, u(x, 0) = u_3(x)$ ;  $x = b, u(x, b) = u_4(x)$ .

В область розв'язання вводиться двовимірний сітка із кроком  $h$  по осі  $x$  і  $l$  по осі  $y$ . Тоді, використовуючи взяті позначення й апроксимуючи рівняння Лапласа різницеvim рівняннями (див. розд. 4.2), отримаємо таку систему лінійних рівнянь ( $l = h$ ):

$$\begin{cases} u_{i,j} = \frac{1}{4}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}); \\ u_{i,0} = u_3(x_i), u_{i,m} = u_4(x_i), u_{0,j} = u_1(y_j), u_{n,j} = u_2(y_j); \\ i = 1, 2, \dots, n-1; j = 1, 2, \dots, m-1. \end{cases} \quad (4.5)$$

Така система рівнянь має велику кількість нульових елементів і задовольняє умову збіжності в разі використання ітераційних методів. Для розв'язання систем рівнянь типу (4.5) найчастіше використовують метод Гаусса-Зейделя (див. розд. 1), який у разі застосування до еліптичних різницеvих рівнянь називається методом Лібмана або методом послідовних зміщень. Алгоритм розв'язання диференціальних рівнянь еліптичного виду на основі рівняння Лапласа та різницеvої схеми (4.5) методом Гаусса-Зейделя подано на рисунку 4.3.

Потрібно відмітити, що будь-які еліптичні рівняння, що не містять  $\frac{\partial^2 u}{\partial x \partial y}$ , зводяться до систем різницеvих рівнянь, які можна розв'язувати як методом Лібмана, так й іншим ітераційними методами, оскільки для них використовуються достатні умови збіжності. Для еліптичних рівнянь, які містять  $\frac{\partial^2 u}{\partial x \partial y}$ , у загальному випадку питання збіжності ітераційних методів не має теоретичного розв'язку, тому необхідно розглядати отриману систему рівнянь у кожному випадку окремо.

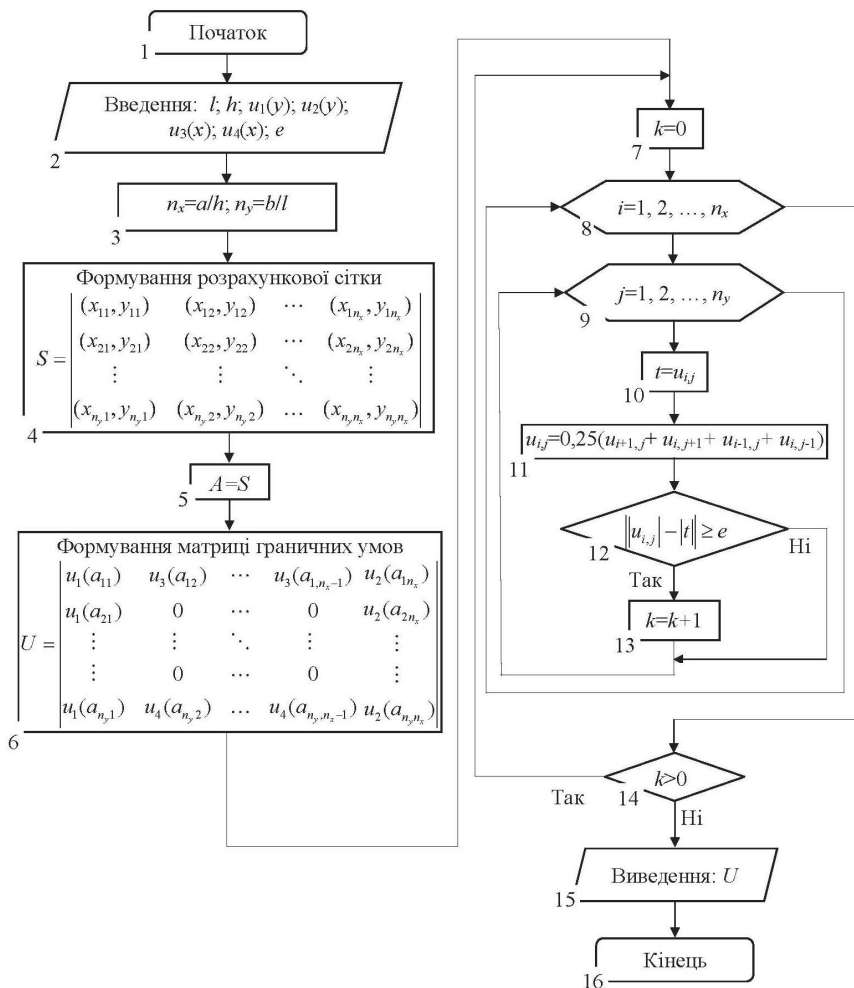


Рисунок 4.3 – Схема алгоритму розв’язання еліптичних диференціальних рівнянь у частинних похідних

**Приклад 4.1.** Визначити стаціонарний розподіл температури в пластині розмірами  $L \times H = 1,0 \times 1,0$  м, для якої задано такі крайові умови:  $u(0,y) = u_1(y) = 0^\circ\text{C}$ ;  $u(L,y) = u_2(y) = 100^\circ\text{C}$ ;  $u(x,0) = u_3(x) = 100x$  град;  $u(x,H) = u_4(x) = 100x^2$  град.

Розв'язання:

Стаціонарний розподіл температури для плоского тіла описується однорідним рівнянням теплопровідності (4.1), а саме рівнянням Лапласа з двома незалежними змінними  $x, y$ :

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0. \quad (4.6)$$

Для постановки задачі необхідно ввести на пластині двовимірну сітку із відстанню між вузлами  $h=0,25$  м (рис. 4.4). Сітка містить 25 вузлів, у 16 із яких температура відома згідно з граничними умовами. Необхідно визначити температуру у всіх 9-ти внутрішніх вузлах сітки. Порядковий номер розрахункового вузла по осі  $x$  позначається індексом  $i$ , а по осі  $y$  – індексом  $j$ . Нове значення температури вузла  $u_{i,j}$  може бути обчислено за допомогою обчислювального шаблону (див. розд. 4.3) із рівняння (4.6), а саме:

оскільки

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} = 0,$$

то

$$u_{i,j} = 0,25(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}). \quad (4.7)$$

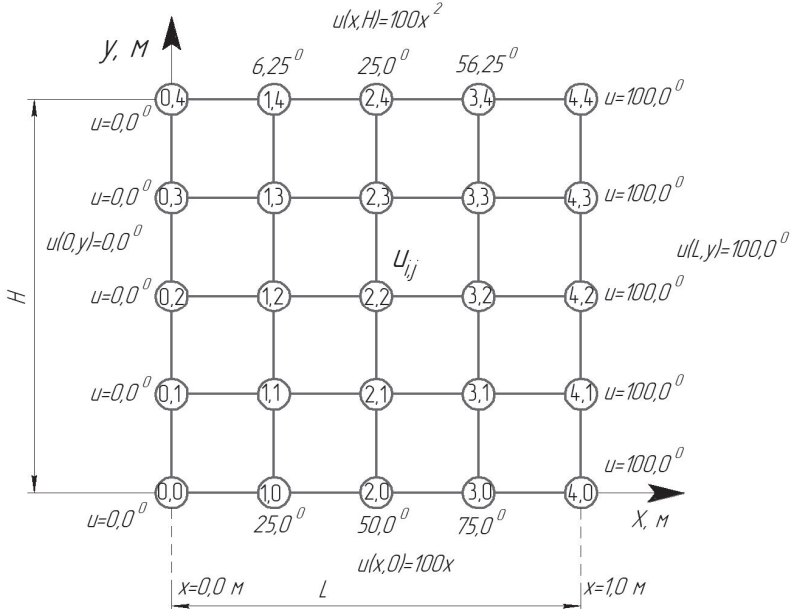


Рисунок 4.4 – Розрахункова схема стаціонарного розподілу температури

На основі рівняння (4.7) запишемо систему рівнянь значення температури для кожного вузла розрахункової сітки пластини:

$$\left\{ \begin{array}{l} u_{11} = 0,25(u_{21} + u_{12} + u_{01} + u_{10}); \\ u_{21} = 0,25(u_{31} + u_{22} + u_{11} + u_{20}); \\ u_{31} = 0,25(u_{41} + u_{32} + u_{21} + u_{30}); \\ u_{12} = 0,25(u_{22} + u_{13} + u_{02} + u_{11}); \\ u_{22} = 0,25(u_{32} + u_{23} + u_{12} + u_{21}); \\ u_{32} = 0,25(u_{42} + u_{33} + u_{22} + u_{31}); \\ u_{13} = 0,25(u_{23} + u_{14} + u_{03} + u_{12}); \\ u_{23} = 0,25(u_{33} + u_{24} + u_{13} + u_{22}); \\ u_{33} = 0,25(u_{43} + u_{34} + u_{23} + u_{32}), \end{array} \right. \quad (4.8)$$

де  $u_{00} = u_{01} = u_{02} = u_{03} = u_{04} = 0$  °C;  $u_{40} = u_{41} = u_{42} = u_{43} = u_{44} = 100$  °C; оскільки  $u_{14} = 100x^2 = 100h^2 = 100 \cdot 0,25^2 = 6,25$  °C, тоді  $u_{24} = 25,0$  °C;  $u_{34} = 56,25$  °C; оскільки  $u_{14} = 100x = 100h = 100 \cdot 0,25 = 25,0$  °C, тоді  $u_{20} = 50,0$  °C;  $u_{30} = 75,0$  °C.

Дійсні значення температури у всіх дев'яти внутрішніх вузлах сітки будуть визначатись методом Гаусса-Зейделя (див. розд. 1), для цього систему рівнянь (4.8) необхідно записати в ітераційному вигляді, вважаючи, що початкове значення температури у шуканих вузлах дорівнює нулю:

$$\left\{ \begin{array}{l} u_{11}^{(n)} = 0,25(u_{21}^{(n-1)} + u_{12}^{(n-1)} + u_{01} + u_{10}); \\ u_{21}^{(n)} = 0,25(u_{31}^{(n-1)} + u_{22}^{(n-1)} + u_{11}^{(n)} + u_{20}); \\ u_{31}^{(n)} = 0,25(u_{41} + u_{32}^{(n-1)} + u_{21}^{(n)} + u_{30}); \\ u_{12}^{(n)} = 0,25(u_{22}^{(n-1)} + u_{13}^{(n-1)} + u_{02} + u_{11}^{(n)}); \\ u_{22}^{(n)} = 0,25(u_{32}^{(n-1)} + u_{23}^{(n-1)} + u_{12}^{(n-1)} + u_{21}^{(n)}); \\ u_{32}^{(n)} = 0,25(u_{42} + u_{33}^{(n-1)} + u_{22}^{(n)} + u_{31}^{(n)}); \\ u_{13}^{(n)} = 0,25(u_{23}^{(n-1)} + u_{14} + u_{03} + u_{12}^{(n)}); \\ u_{23}^{(n)} = 0,25(u_{33}^{(0)} + u_{24} + u_{13}^{(1)} + u_{22}^{(1)}); \\ u_{33}^{(n)} = 0,25(u_{43} + u_{34} + u_{23}^{(1)} + T_{32}^{(1)}), \end{array} \right. \quad (4.9)$$

де  $u_{11}^{(0)} = u_{21}^{(0)} = u_{31}^{(0)} = u_{12}^{(0)} = u_{22}^{(0)} = u_{32}^{(0)} = u_{13}^{(0)} = u_{23}^{(0)} = u_{33}^{(0)} = 0$  – початкові значення температури в розрахункових вузлах;  $n = 1, 2, \dots$  – номер обчислювальної ітерації.

Використавши алгоритм обчислень Гаусса-Зейделя (див. рис. 4.3) із похибкою обчислень  $\varepsilon = 0,001$  після 17-ти ітерацій було отримано такий результат обчислень:

$$u^{(17)} = \begin{pmatrix} u_{04} & u_{14} & u_{24} & u_{34} & u_{44} \\ u_{03} & u_{13} & u_{23} & u_{33} & u_{43} \\ u_{02} & u_{12} & u_{22} & u_{32} & u_{42} \\ u_{01} & u_{11} & u_{21} & u_{31} & u_{41} \\ u_{00} & u_{10} & u_{20} & u_{30} & u_{40} \end{pmatrix} = \begin{pmatrix} 0,0 & 6,25 & 25,0 & 56,25 & 100,0 \\ 0,0 & 16,35 & 38,06 & 66,35 & 100,0 \\ 0,0 & 21,09 & 44,53 & 71,09 & 100,0 \\ 0,0 & 23,49 & 47,88 & 73,49 & 100,0 \\ 0,0 & 25,0 & 50,0 & 75,0 & 100,0 \end{pmatrix}. \quad (4.10)$$

На основі результатів обчислень (4.10) можна побудувати тривимірний графік стаціонарного розподілу температури в пластині (рис. 4.5).

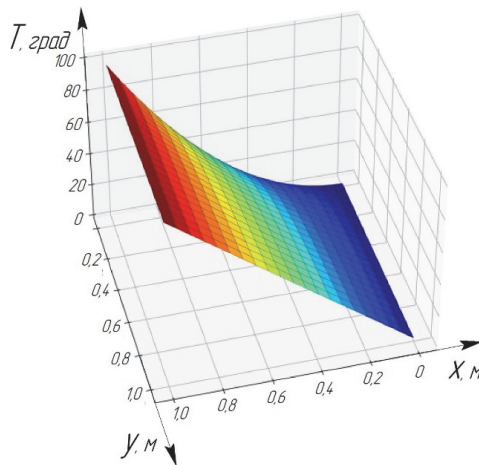


Рисунок 4.5 – Діаграма стаціонарного розподілу температури в пластині

Результат розв’язання прикладу 4.1 мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from scipy import *
from sympy import *
from matplotlib.colors import LinearSegmentedColormap
#введення значення довжини пластини
l=float(input())
#введення значення ширини пластини
h=float(input())
#введення значення кроку розрахункової сітки по вісі x
h_x=float(input())
#введення значення кроку розрахункової сітки по вісі y
h_y=float(input())
```

```

#введення значення крайових умов:
#введення значення температури на лівій стінці пластини
t_left=float(input())
#введення значення температури на правій стінці пластини
t_right=float(input())
#введення функції початкового розподілу температури на верхній стінці
# пластини
def t_top(x):
    return (t_right/(l*1))*x*x
#введення функції початкового розподілу температури на нижній стінці
# пластини
def t_down(x):
    return (t_right/l)*x
#введення значення похибки обчислення
e=float(input())
#введення значень крайових умов:
#для верхньої стінки
side_top, side_down=[], []
x=0
for i in range(0,int(l/h_x)+1):
    side_top.append(t_top(x))
#для нижньої стінки
    side_down.append(t_down(x))
    x=x+h_x
side_top, side_down=np.array(side_top),np.array(side_down)
#для лівої стінки і правої стінок
side_left, side_right=[float(t_left)]*(int(h/h_y)+1),
    [float(t_right)]*(int(h/h_y)+1)
side_left, side_right=np.array(side_left),np.array(side_right)
side_right=side_right.reshape(-1,1); side_left=side_left.reshape(-1,1)
u=np.full((int(h/h_y)+1,int(l/h_x)+1),0.0)
u=np.append(u,side_right,axis=1); u=np.delete(u,int(l/h_x),axis=1)
u=np.insert(u,[0],side_left,axis=1); u=np.delete(u,1,axis=1)
u=np.insert(u,int(h/h_y)+1,side_down,axis=0)
u=np.delete(u,int(h/h_y),axis=0)
u=np.insert(u,0,side_top,axis=0); u=np.delete(u,1,axis=0)
print('Початкова розрахункова сітка із крайовими умовами'); print(u)
#обчислення системи різницевих рівнянь методом Гаусса-Зейделя
k=0
while True:
    count=0
    for i in range(1,int(h/h_y)):
        for j in range(1,int(l/h_x)):
            u_last=u[i,j]
            u[i,j]=0.25*(u[i+1,j]+u[i,j+1]+u[i-1,j]+u[i,j-1])
            if abs(abs(u_last)-abs(u[i,j]))>=e:
                count=count+1
        k=k+1
    if count==0:
        break
u=np.around(u, decimals=2)
print('Кількість обчислювальних ітерацій k=',k)
print('Розподіл температури на розрахунковій сітці'); print(u)
#побудова тривимірного графіка розподілу температури
x=np.linspace(0, l, int(l/h_x)+1)
y=np.linspace(0, h, int(h/h_y)+1)
x,y=np.meshgrid(x, y)
fig = plt.figure(figsize=(20, 20))
axes = fig.add_subplot(1, 2, 1, projection='3d')
axes.plot_surface(x, y, u, rcount=1000, ccount=1000, linewidth=0.2,
    edgecolors='k', cmap='jet')
axes.view_init(elev=35, azim=75)
axes.set_xlabel('X')
axes.set_ylabel('Y')
axes.set_zlabel('T')
plt.show().

```

## Розв'язання гіперболічних рівнянь

Одним із найпоширеніших в інженерній практиці видів гіперболічного рівняння із частинними похідними другого порядку є хвильове рівняння, яке описує різні види коливань:

$$\frac{\partial^2 u}{\partial t^2} = a^2 \left( \frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2} + \dots + \frac{\partial^2 u}{\partial x_n^2} \right) + f(x, t), \quad (4.11)$$

де  $x = (x_1, \dots, x_n)$  – декартові координати;  $f(x, t)$  – функція зовнішнього впливу (зовнішня сила);  $t \in R$  – час;  $a$  – фазова швидкість;  $u(x, t)$  – функція положення хвилі в точці із координатами  $x$  в момент часу  $t$ .

Залежно від кількості декартових координат розрізняють одновимірне, двовимірне й тривимірне хвильові рівняння.

Одновимірне хвильове рівняння описує повздовжні коливання стержня, перерізи якого здійснюють плоскопаралельні коливальні рухи, а також поперечні коливання тонкого стержня (струни) та інші задачі.

Двовимірне хвильове рівняння використовується для дослідження коливань тонкої пластини (мембрани).

Тривимірне хвильове рівняння описує поширення хвиль у просторі (наприклад, звукових хвиль у рідині, пружних хвиль у суцільному середовищі тощо).

Одновимірне однорідне хвильове рівняння для випадку вільних коливань, на основі рівняння (4.11), записується у такому вигляді:

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2}, \quad (4.12)$$

де  $u(x, t)$  – функція, яка описує положення струни в момент часу  $t$ ;  $a^2 = T/\rho$  ( $T$  – сила натягу струни,  $\rho$  – її лінійна (погонна) густина);  $f(x, t) = 0$  – функція зовнішнього впливу (див. (4.11)).

Коливання припускаються малими, тобто їх амплітуда мала порівняно із довжиною струни. Опір середовища коливальному процесу не враховується.

Найпростішою задачею для рівняння (4.12) є задача Коші: в початковий момент часу задаються дві умови (кількість умов дорівнює порядку похідної за часом  $t$ ):

$$u|_{t=0} = u(x, 0) = \varphi(x), \quad \frac{\partial u}{\partial t} \Big|_{t=0} = \psi(x). \quad (4.13)$$

Ці умови описують початкову форму струни  $u = \varphi(x)$  і швидкість руху її точок  $\psi(x)$ . На практиці розв'язується не задача Коші для нескінченної струни, а змішана задача для обмеженої струни деякої довжини  $l$ . У цьому випадку задаються граничні умови на її кінцях  $u(0, t) = \mu_1(t)$  і  $u(l, t) = \mu_2(t)$ .

Наприклад, за закріплених кінців їх зміщення дорівнюють нулю, й граничні умови мають вигляд:

$$u|_{t=0} = 0, u|_{x=l} = 0. \quad (4.14)$$

Для розв'язання задачі (4.12)–(4.14) найчастіше використовують тришарову схему, де сукупність вузлів за  $t = \text{const}$  називається шаром. Водночас вводиться рівномірна прямокутна сітка:  $x_i = i \cdot h$  ( $i = 0, 1, \dots, n$ ),  $\tau_i = j \cdot \tau$  ( $j = 0, 1, \dots, m$ ). На основі базових різницьових схем (див. розд. 4.3) рівняння (4.12) подається кінцево-різницьовими співвідношеннями:

$$\begin{cases} \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{\tau^2} = a^2 \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}; \\ i = 1, 2, \dots, n-1; j = 1, 2, \dots, m-1. \end{cases} \quad (4.15)$$

Із рівняння (4.15) визначається явний вираз для значення сіткової функції на  $j+1$  шарі:

$$u_{i,j+1} = \lambda(u_{i+1,j} + u_{i-1,j}) + 2(1-\lambda)u_{i,j} - u_{i,j-1}, \quad (4.16)$$

де  $\lambda = \frac{a^2 \tau^2}{h^2}$ .

Схема розв'язання на основі рівняння (4.16) називається тришаровою, оскільки пов'язує значення  $u_{ij}$  на трьох часових шарах  $j-1, j, j+1$ . Для визначення невідомих значень на  $j+1$  шарі необхідно знати розв'язки на  $j$ -ому й  $(j-1)$ -ому шарах. Тому необхідно розпочати обчислення за формулами (4.16) з другого шару, а розв'язки на нульовому й першому шарах мають бути відомими. Тобто вони визначаються за допомогою початкових умов (4.13), а саме на нульовому шарі:

$$u_{i,0} = \varphi(x_i) \quad (i=0, 1, \dots, n). \quad (4.17)$$

Для отримання розв'язку на першому шарі необхідно використати другу початкову умову (4.13), де похідна  $\partial u / \partial t$  замінюється кінцево-різницевою апроксимацією:

$$\left. \frac{\partial u}{\partial t} \right|_{t=0} \approx \frac{u_{i,1} - u_{i,0}}{\tau} \approx \psi(x_i). \quad (4.18)$$

Із співвідношення (4.18) визначається значення сіткової функції на першому часовому шарі:

$$u_{i,1} = u_{i,0} + \tau \psi(x_i) \quad (i=0, 1, \dots, n; t=0). \quad (4.19)$$

Потрібно відмітити, що апроксимація початкової умови у вигляді (4.18) погіршує апроксимацію вихідної диференціальної задачі, похибка апроксимації стає порядку  $O(h^2 + \tau)$ , тобто першого порядку за  $\tau$ , хоча сама



схема (4.16) має другий порядок апроксимації за  $h$  і  $\tau$ . Тому для підвищення точності замість рівняння (4.19) вибирається більш точне подання:

$$u_{i,1} = u_{i,1} + \tau \left. \frac{\partial u}{\partial t} \right|_{t=0} + \frac{\tau^2}{2} \left. \frac{\partial^2 u}{\partial t^2} \right|_{t=0}. \quad (4.20)$$

Замість  $\partial u / \partial t$  використовується  $\psi(x)$ . А вираз для другої похідної рівняння (4.20) може бути визначений за допомогою рівняння (4.12) і першої початкової умови (4.13), а саме:

$$\left. \frac{\partial^2 u}{\partial t^2} \right|_{t=0} = a^2 \left. \frac{\partial^2 u}{\partial t^2} \right|_{t=0} = a^2 \frac{\partial^2 \varphi}{\partial x^2}.$$

Тоді рівняння (4.19) набуває такого вигляду:

$$u_{i,1} = u_{i,0} + \tau \psi(x_i) + a^2 \frac{\tau^2}{2} \varphi''(x_i) \quad (i=0, 1, \dots, n). \quad (4.21)$$

Різницева схема (4.15) із врахуванням (4.21) має похибку апроксимації другого порядку точності  $O(h^2 + \tau^2)$ .

Під час розв'язування змішаної задачі із граничними умовами виду (4.14), тобто коли на кінцях розглянутого відрізка задано значення самої функції, другий порядок апроксимації зберігається. У такому випадку крайні вузли сітки розташовуються в граничних точках ( $x_0 = 0$ ,  $x_1 = l$ ). Проте граничні умови можуть бути задані і для похідної. Наприклад, у випадку вільних повздовжніх коливань стержня на його незакріпленому кінці задається умова:

$$\left. \frac{\partial u}{\partial x} \right|_{x=l} = 0. \quad (4.22)$$

Якщо цю умову записати в різницевому вигляді з першим порядком апроксимації, то похибка апроксимації схеми стане порядку  $O(h^2 + \tau^2)$ . Тому для збереження другого порядку цієї схеми за  $h$  необхідно граничну умову (4.22) апроксимувати із другим порядком точності.

Різницева схема (4.16) розв'язання задачі (4.12) – (4.14) умовно стійка, тому необхідна і достатня умова стійкості має вигляд:

$$r = a \frac{\tau}{h} < 1. \quad (4.23)$$

Умова (4.23) забезпечує прийнятну точність отримання розв'язку  $u(x, t)$ , який має неперервні похідні четвертого порядку. Причому за умови  $r > 1$  розв'язок нестійкий, а за  $r < 1$  розв'язок хоча і стійкий, але точність його у разі зменшенні  $r$  знижується; за умови, що  $r = 1$  різницевий розв'язок стійкий і збігається із точним.

Алгоритм моделювання вільних коливань струни на основі хвильового рівняння (4.12) подано на рисунку 4.6.

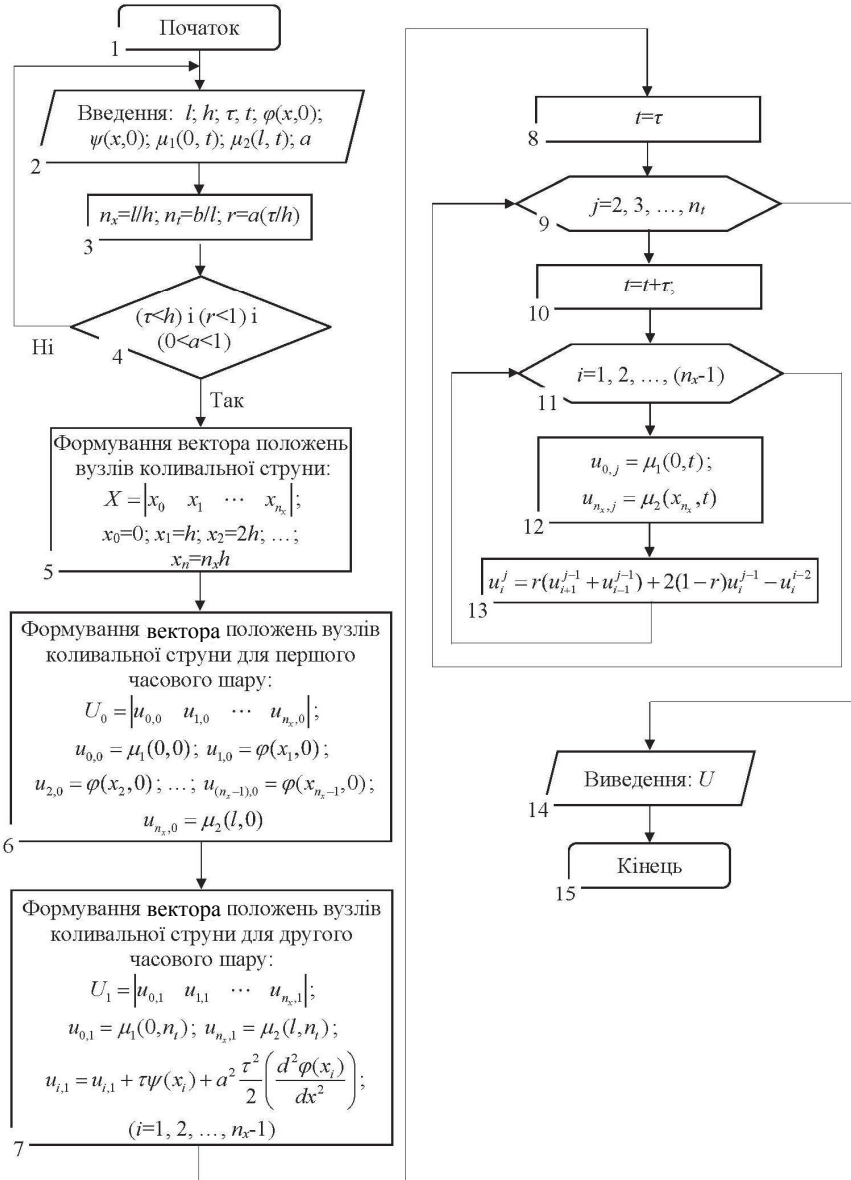


Рисунок 4.6 – Схема алгоритму розв’язання гіперболічних диференціальних рівнянь у частинних похідних

**Приклад 4.2.** Визначити положення струни залежно від часу, яка виконує вільні коливання протягом часу  $t = 2,0$  с із закріпленими кінцями (відстань між точками закріплення кінців  $L = 4,0$  м) і для якої задано такі умови:

- коефіцієнт фазової швидкості  $a = 1,0$  м/с;
- початкові

$$\varphi(x) = u(x, 0) = \cos^2\left(\frac{x}{2}\right), \quad \psi(x) = \frac{\partial u(x, 0)}{\partial t} = -\frac{\sin(x)}{2} \quad (0 \leq x \leq L);$$

- граничні

$$u(0, t) = \cos^2\left(\frac{t}{2}\right), \quad u(L, t) = \cos^2\left(\frac{t}{2} + 2, 0\right) \quad (0 \leq t \leq 2, 0).$$

Порівняти отримане рішення із точним розв'язком:  $u(x, t) = \cos^2\left(\frac{t+x}{2}\right)$ .

*Розв'язання:*

Вільні коливання струни описуються одним із видів гіперболічних рівнянь, а саме однорідним хвильовим рівнянням (4.12) з двома незалежними змінними  $x, t$ :

$$\frac{\partial^2 u}{\partial t^2} = a^2 \frac{\partial^2 u}{\partial x^2}. \quad (4.24)$$

Для постановки задачі і виконання умови стійкості розв'язку різницевої схеми (4.23) необхідно ввести двовимірну розрахункову сітку, а саме: по горизонтальній осі  $x$  із відстанню між вузлами  $h=0,2$  м (рис. 4.7), а також для часу (змінна  $t$ ) із відстанню між вузлами  $\tau=0,1$  с. Сітка по горизонтальній осі  $x$  містить  $n_x=(L/h)+1=(4,0/0,2)+1=21$  вузол, а по змінній часу  $\tau$  містить також  $n_\tau=(t/\tau)+1=(2,0/0,1)+1=21$  вузол. Порядковий номер розрахункового вузла по осі  $x$  позначається індексом  $i$ , а по часовій осі – індексом  $j$ .

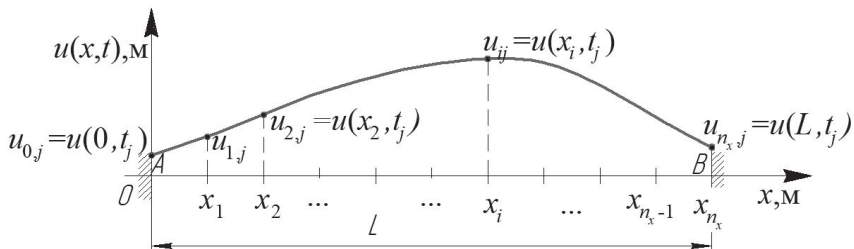


Рисунок 4.7 – Розрахункова схема вільних коливань струни

Значення вектора положень коливальної струни для першого часового шару в момент часу  $t_0 = 0,0$  с ( $j = 0$ ):

$$U_0 = (u_{0,0} \quad u_{1,0} \quad \dots \quad u_{i,0} \quad \dots \quad u_{n_x,0}) = (1,0 \quad 0,990 \quad 0,961 \quad 0,913 \quad 0,848 \\ 0,770 \quad 0,681 \quad 0,585 \quad 0,485 \quad 0,386 \quad 0,292 \quad 0,206 \quad 0,131 \quad 0,716 \quad 0,289 \\ 0,005 \quad 0,00085 \quad 0,017 \quad 0,052 \quad 0,105 \quad 0,173),$$

$$\text{де } u_{0,0} = u(0,0) = \cos^2\left(\frac{t}{2}\right) = \cos^2\left(\frac{0,0}{2}\right) = 1,00 \text{ м; } \quad u_{i,0} = u(x_i, 0) = \cos^2\left(\frac{x_i}{2}\right);$$

$$u_{1,0} = u(h,0) = \cos^2\left(\frac{x_1}{2}\right) = \cos^2\left(\frac{0,2}{2}\right) = 0,99 \text{ м;}$$

$$u_{20,0} = u(4,0;0) = \cos^2\left(\frac{t}{2} + 2,0\right) = \cos^2\left(\frac{0,0}{2} + 2,0\right) = 0,173 \text{ м.}$$

Значення вектора положень коливальної струни для другого часового шару, в момент часу  $t_1 = \tau = 0,1$  с ( $j = 1$ ) на основі рівняння (4.21):

$$U_1 = (u_{0,1} \quad u_{1,1} \quad \dots \quad u_{i,1} \quad \dots \quad u_{n_x,1}) = (0,998 \quad 0,978 \quad 0,939 \quad 0,882 \quad 0,811 \\ 0,727 \quad 0,634 \quad 0,535 \quad 0,435 \quad 0,338 \quad 0,248 \quad 0,167 \quad 0,099 \quad 0,048 \quad 0,014 \\ 0,0004 \quad 0,0063 \quad 0,032 \quad 0,076 \quad 0,137 \quad 0,213),$$

$$\text{де } u_{0,1} = u(0; 0,1) = \cos^2\left(\frac{t_1}{2}\right) = \cos^2\left(\frac{0,1}{2}\right) = 0,998 \text{ м; } \quad u_{20,1} = u(4; 0,1) = \cos^2\left(\frac{t_1}{2} + 2,0\right) = \cos^2\left(\frac{0,1}{2} + 2,0\right) = 0,173 \text{ м;}$$

$$u_{i,1} = u_{i,0} + \tau \psi(x_i) + a^2 \frac{\tau^2}{2} \left( \frac{d^2 \varphi(x)}{dx^2} \right) = u_{i,0} +$$

$$+ \tau \left( -\frac{\sin(x_i)}{2} \right) + a^2 \frac{\tau^2}{2} \left( -\frac{\cos(x_i)}{2} \right); \quad u_{1,1} = u(h; 0,1) = u_{1,0} + \tau \left( -\frac{\sin(x_1)}{2} \right) +$$

$$+ a^2 \frac{\tau^2}{2} \left( -\frac{\cos(x_1)}{2} \right) = 0,99 + 0,1 \left( -\frac{\sin(0,1)}{2} \right) + 1^2 \frac{0,1^2}{2} \left( -\frac{\cos(0,2)}{2} \right) = 0,978 \text{ м.}$$

Значення вектора положень коливальної струни для третього часового шару, в момент часу  $t_2 = 2\tau = 2 \cdot 0,1$  с на основі рівняння (4.16) за того самого

значення  $j=1$ , для  $\lambda = \frac{a^2 \tau^2}{h^2} = \frac{1,0^2 \cdot 0,1^2}{0,2^2} = 0,25$ :

$$U_2 = (u_{0,2} \quad u_{1,2} \quad \dots \quad u_{i,2} \quad \dots \quad u_{n_x,2}) = (0,990 \quad 0,961 \quad 0,913 \quad 0,848 \quad 0,770 \\ 0,681 \quad 0,585 \quad 0,485 \quad 0,386 \quad 0,292 \quad 0,206 \quad 0,131 \quad 0,071 \quad 0,029 \quad 0,005 \\ 0,0008 \quad 0,0167 \quad 0,052 \quad 0,105 \quad 0,173 \quad 0,255),$$

$$\text{де } u_{0,2} = u(0; 0,2) = \cos^2\left(\frac{t_2}{2}\right) = \cos^2\left(\frac{0,2}{2}\right) = 0,990 \text{ м; } \quad u_{i,j+1} = \lambda(u_{i+1,j} + u_{i-1,j}) +$$

$$+ 2(1 - \lambda)u_{i,j} - u_{i,j-1}; \quad u_{1,2} = \lambda(u_{2,1} + u_{0,1}) + 2(1 - \lambda)u_{1,1} - u_{1,0} = 0,25(0,939 + 0,998) +$$

$$+2(1-0,25)0,978-0,990=0,961 \text{ м}; \quad u_{20,2} = u(4,0; 0,2) = \cos^2\left(\frac{t_2}{2} + 2,0\right) =$$

$$= \cos^2\left(\frac{0,2}{2} + 2,0\right) = 0,255 \text{ м.}$$

Аналогічно за допомогою формули (4.16) визначаються вектори положень коливальної струни для решти часових шарів ( $j = 3, 4, \dots, 20$ ). На основі результатів обчислень можна побудувати діаграму положень струни для різних моментів часу під час вільного коливання (рис. 4.8), де чітко можна визначити переміщення хвилі (т.  $A$ ).

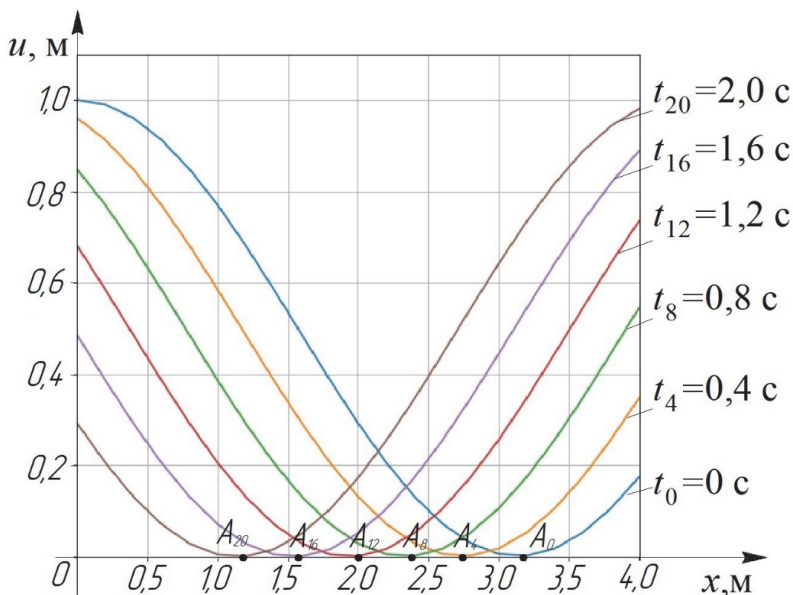


Рисунок 4.8 – Діаграма положень струни для різних моментів часу під час вільного коливання

Відносна похибка обчислення хвильового рівняння шляхом порівняння між значеннями аналітичного розв'язку і чисельним методом для:

– третього часового шару у вузловій точці  $i=10$  ( $x_{10}=2,0$  м)

$$\varepsilon_{10,2} = \left| \frac{u_{10,2}^{an} - u_{10,2}^c}{u_{10,2}^{an}} \right| \cdot 100\% = \left| \frac{0,2058 - 0,2060}{0,2058} \right| \cdot 100\% = 0,01\%,$$

де  $u_{10,2}^{an} = \cos^2\left(\frac{t_2 + x_{10}}{2}\right) = \cos^2\left(\frac{0,2 + 2,0}{2}\right) = 0,2058$  м,  $u_{10,2}^c = 0,2060$  м.

– четвертого часового шару у вузловій точці  $i=15$  ( $x_{10}=3,0$  м)

$$\varepsilon_{15,3} = \left| \frac{u_{15,3}^{an} - u_{15,3}^u}{u_{15,3}^{an}} \right| \cdot 100\% = \left| \frac{0,00630 - 0,00626}{0,00626} \right| \cdot 100\% = 0,64\%,$$

$$\begin{aligned} \text{де } u_{15,3}^{an} &= \cos^2 \left( \frac{t_3 + x_{15}}{2} \right) = \cos^2 \left( \frac{0,3 + 3,0}{2} \right) = 0,00630 \text{ м, } u_{15,3}^u = \lambda(u_{16,2} + u_{14,2}) + \\ &+ 2(1 - \lambda)u_{15,2} - u_{15,1} = 0,25(0,0167 + 0,0050) + 2(1 - 0,25)0,0008 - 0,0004 = \\ &= 0,00626 \text{ м.} \end{aligned}$$

Значення відносних похибок обчислення чисельним методом кінцевих різниць показує високу точність обчислення диференціальних рівнянь у частинних похідних гіперболічного типу за допомогою явної різницевої схеми значення сіткової функції.

Результат розв'язання прикладу 4.2 мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d
from scipy import *
from sympy import *
#введення значення довжини струни
l=float(input()) #l=4.0
#введення значення кроку розрахункової сітки по довжині струни
h_x=float(input()) #h_x=0.2
#введення значення кроку розрахункової сітки по часу
h_t=float(input()) #h_t=0.1
#введення значення крайових умов:
#введення значення координати положення лівого кінця струни
def u_left(t):
    return np.cos(0.5*t)*np.cos(0.5*t)
#введення значення координати положення правого кінця струни
def u_right(t):
    return np.cos((0.5*t)+2)*np.cos((0.5*t)+2)
#введення функції початкового положення струни
def u_x0(x):
    return np.cos(0.5*x)*np.cos(0.5*x)
#введення функції початкової швидкості точок струни
def v_first(x):
    return -0.5*np.sin(x)
#введення значення загального часу колювання
t=float(input()) #t=2.0
#введення значення похибки обчислення
e=float(input()) #e=0.005
#введення показника натягу струни
a=float(input()) #a=1
lamb_da=(a*a*h_t*h_t)/(h_x*h_x)
#контроль умови стійкості розв'язку
if np.sqrt(lamb_da)>1:
    print("Умова розв'язку не стійка. Змініть параметри розв'язку!")
#введення значень крайових умов:
u_first,u_second,x_coord=[u_left(0)],[u_left(h_t)],[0]
x=h_x
for i in range(0,int(1/h_x)):
#для першого часового шару
    u_first.append(u_x0(x))
```

```

#для другого часового шару
if i==int(1/h_x):
    u_first.append(u_right(0))
else:
    u_second.append(u_x0(x)+(v_first(x)*h_t)+(0.5*(a*a)*(h_t*h_t)*
                                                (0.5*(-np.cos(x)))))
    x_coord.append(x)
    x=x+h_x
u_second[int(1/h_x)]=u_right(h_t)
u_first, u_second=np.array(u_first),np.array(u_second)
u=np.array([u_first,u_second])
# розв'язання тришарової системи кінцево-різницевих рівнянь
t_0=h_t
for j in range(2,int(t/h_t)+1):
    t_0=t_0+h_t; u_next=[u_left(t_0)]
    for i in range(0,int(1/h_x)-1):
        u_next.append(lamb_da*(u[j-1,i+2]+u[j-1,i])+
                      (2*(1-lamb_da)*u[j-1,i+1])-u[j-2,i+1])
    u_next.append(u_right(t_0)); u=np.insert(u,j,u_next,axis=0)
    del u_next
print('Матриця значень положення коливальної струни для кожного моменту
      часу:')
print(u)
#побудова графіку рішень гіперболічного (хвильового) рівняння
k=0
plt.figure(figsize=(8, 8))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
for i in range(0,6):
    plt.plot(x_coord, u[k]); k=k+4
plt.grid(True)
plt.xlim([0, 4])
plt.ylim([-0.1, 1.1])
plt.show()
#обчислення значень точного розв'язку гіперболічного (хвильового) рівняння
x_an=0; t_an=0
u_an=[]
for i in range(0,int(1/h_x)+1):
    u_an.append(np.cos(0.5*(x_an+t_an))*np.cos(0.5*(x_an+t_an)))
    x_an=x_an+h_x
t_an=t_an+h_t
u_an=np.array([u_an])
for j in range(1,int(t/h_t)+1):
    u_tran=[]; x_an=0
    for i in range(0,int(1/h_x)+1):
        u_tran.append(np.cos(0.5*(x_an+t_an))*np.cos(0.5*(x_an+t_an)))
        x_an=x_an+h_x
    t_an=t_an+h_t
    u_an=np.insert(u_an,j,u_tran,axis=0)
    del u_tran
print('Матриця точних значень положення коливальної струни для кожного
      моменту часу:')
print(u_an)
#визначення похибки обчислення
e_tran=[]; count=0
for j in range(0,int(t/h_t)+1):
    for i in range(0,int(1/h_x)-1):
        e_tran.append(abs(u_an[j,i]-u[j,i]))
        if abs(u_an[j,i]-u[j,i])>=e:
            count=count+1
if count>0:
    print("Похибка обчислення перевищує задану точність. Змініть початкові
          параметри обчислення!").

```

## Розв'язання параболічних рівнянь

Прикладом задачі, яка зводиться до параболічного рівняння в частинних похідних, є задача нестационарної теплопровідності, що описується однорідним рівнянням (4.2). Зокрема в однорідній задачі теплопровідності необхідно визначити функцію  $u(x, t)$ , яка задовольняє в області  $\Omega = \{(x, t), 0 \leq x \leq l, 0 \leq t \leq T\}$  рівняння:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} \quad (k = \text{const} > 0), \quad (4.25)$$

початкову умову  $u(x, 0) = u_0(x)$  і крайові умови першого роду  $u(0, t) = \mu_1(t)$  і  $u(l, t) = \mu_2(t)$ .

Можливі два варіанти побудови різницевого рівняння на розрахунковій сітці із кроком  $h$  по  $x$  і  $\tau$  по  $t$ .

Варіант апроксимації за допомогою чотириточкового шаблону, центрально-різницевої схеми другого роду (див. розд. 4.3) і лівої різницевої схеми  $\frac{\partial u}{\partial t} \approx \frac{u_{i,j+1} - u_{i,j}}{\tau} \approx \frac{1}{\tau} [(1) \text{---} (-1) \text{---} (0)]_{i,j}$  призводить до явної двошарової

схеми, а саме:  $\frac{u_{i,j+1} - u_{i,j}}{\tau} \approx \alpha \left( \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \right)$  ( $i = 1, 2, \dots, n-1$ ;

$j = 0, 1, \dots, m-1$ ), тоді

$$u_{i,j+1} = \delta u_{i+1,j} + (1 - 2\delta)u_{i,j} + \delta u_{i-1,j}, \quad (4.26)$$

де  $\delta = \alpha \frac{\tau}{h^2}$ .

Явна двошарова різницева схема (4.26) стійка тільки за  $\delta \leq 0.5$ , що призводить до необхідності проводити обчислення із дуже малим кроком по  $t$  ( $\tau \leq 0.5h^2$ ), а це також обмежує швидкодію і потребує великих витрат машиночасу комп'ютерних систем.

Алгоритм розв'язання диференціальних рівнянь параболічного виду на основі рівняння теплопровідності та явної різницевої схеми (4.26) подано на рисунку 4.9.

Тому для параболічних рівнянь найширшого застосування набула неявна схема, де апроксимація виконується за допомогою чотириточкового шаблону, центрально-різницевої схеми другого роду (див. розд. 4.3) і правої

різницевої схеми  $\frac{\partial u}{\partial t} \approx \frac{u_{i,j} - u_{i,j-1}}{\tau} \approx \frac{1}{\tau} [(0) \text{---} (1) \text{---} (-1)]_{i,j}$  призводить до неявної

двошарової схеми, а саме:  $\frac{u_{i,j} - u_{i,j-1}}{\tau} \approx \alpha \left( \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} \right)$  ( $i = 1, 2, \dots,$

$n-1; j = 1, 2, \dots, m$ ), тоді

$$-u_{i,j-1} = \delta u_{i+1,j} - (1 + 2\delta)u_{i,j} + \delta u_{i-1,j}. \quad (4.27)$$



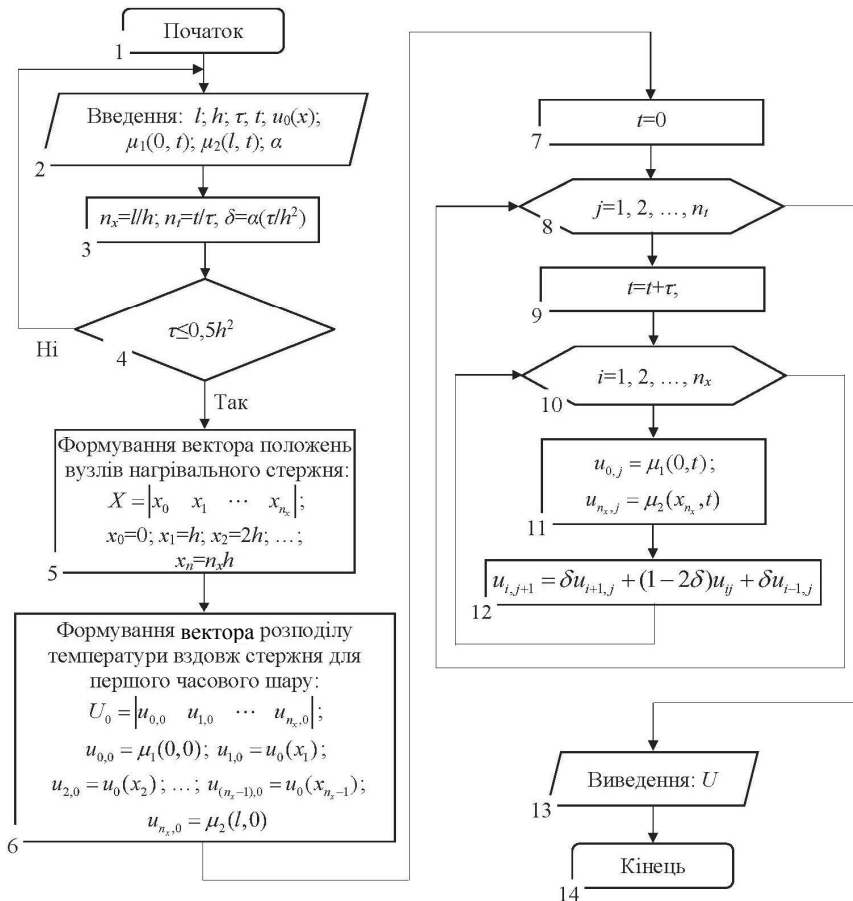


Рисунок 4.9 – Схема алгоритму розв’язання параболічних диференціальних рівнянь у частинних похідних явним різницеvim методом

Неявна двошарова різницева схема (4.27), доповнена рівняннями із крайових умов  $u_{0,j} = \mu_1(t_j)$  і  $u_{n_x,j} = \mu_2(t_j)$  приводить до системи рівнянь, яка має стійкий розв’язок за будь-яких значеннях  $\delta$ .

Алгоритм розв’язання диференціальних рівнянь параболічного виду на основі рівняння теплопровідності та неявної різницевої схеми (4.27) методом прогонки подано на рисунку 4.10.

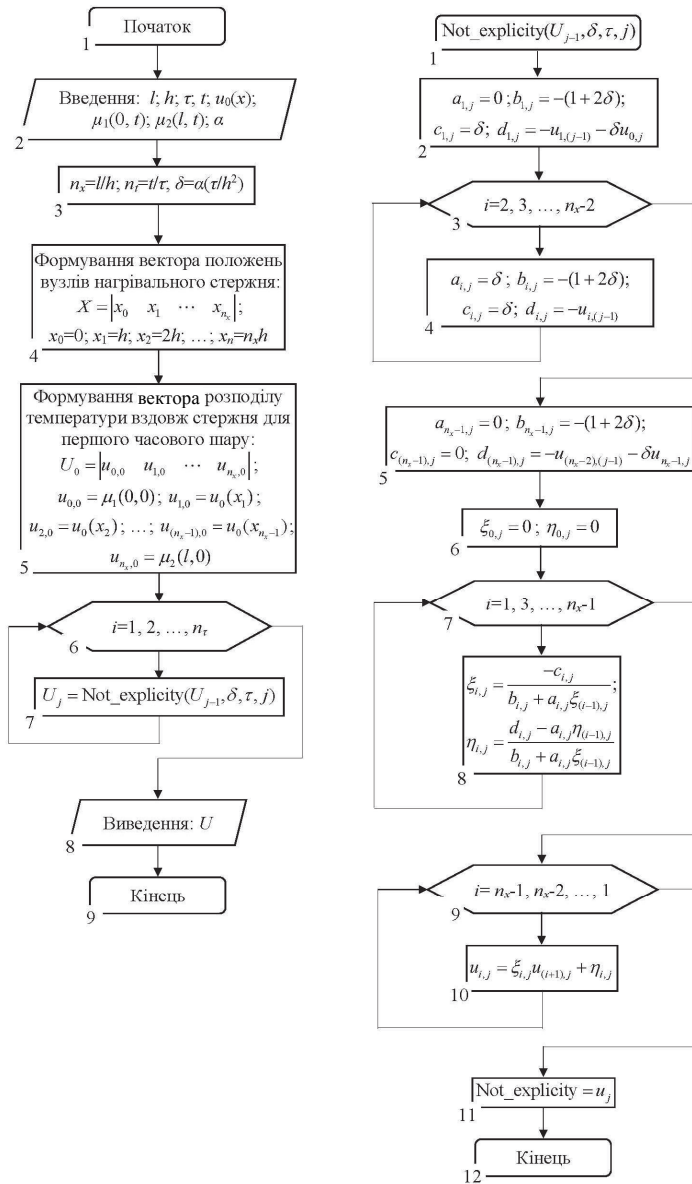


Рисунок 4.10 – Схема алгоритму розв’язання параболічних диференціальних рівнянь у частинних похідних неявним різницевим методом

**Приклад 4.3.** Мідний стержень довжиною  $L = 2,0$  м із постійною по довжині площею поперечного перерізу розміщений в ізолюваному матеріалі таким чином, що з навколишнім середовищем взаємодіють тільки його крайні правий і лівий торці. У початковий момент часу стержень має урівноважену температуру  $T = 0$  °С, а на його лівому і правому торцях постійно підтримується температура  $T_l = 80,0$  °С і  $T = 10,0$  °С, відповідно. Необхідно визначити зміну розподілу температури вздовж стержня залежно від часу протягом однієї години ( $t=3600,0$  с). Порівняти між собою результати обчислень за допомогою явної і неявної різницевих схем.

*Розв'язання:*

Нестационарний розподіл температури по довжині тіла описується однорідним рівнянням теплопровідності (4.1), а саме:

$$\frac{\partial^2 u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}. \quad (4.28)$$

Для постановки задачі і виконання умови стійкості розв'язку різницевої схеми (4.26) необхідно ввести двовимірну розрахункову сітку: по горизонтальній осі  $x$  із відстанню між вузлами  $h = 0,2$  м (рис. 4.11); для явної різницевої схеми відстань по вертикальній осі для змінної часу  $t$  із відстанню між вузлами  $\tau = 0,5h^2 = 0,5 \cdot 0,2^2 = 0,02$  с; для неявної –  $\tau = 0,1$  с. Сітка по горизонтальній осі  $x$  містить  $n_x = (L/h) + 1 = (2,0/0,2) + 1 = 11$  вузлів, а сітка по вертикальній осі  $\tau$  містить: для явної різницевої семи –  $n_\tau^* = (t/\tau) + 1 = (3600,0/0,02) + 1 = 180001$  вузлів; для неявної –  $n_\tau^* = (t/\tau) + 1 = (3600,0/0,1) + 1 = 36001$ . Порядковий номер розрахункового вузла по осі  $x$  позначається індексом  $i$ , а по часовій осі – індексом  $j$ .

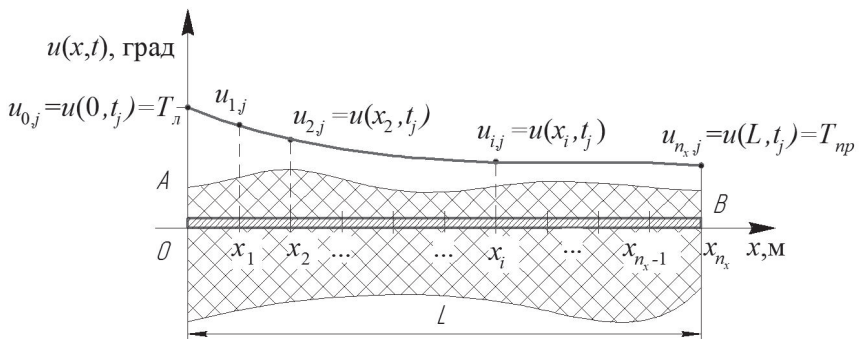


Рисунок 4.11 – Розрахункова схема теплопередачі вздовж стержня

Значення вектора значень температури вздовж стержня для першого часового шару в момент часу  $t_0 = 0$  с ( $j = 0$ ) як для явної, так і неявної різницевої схем:

$$U_0 = (u_{0,0} \quad u_{1,0} \quad \dots \quad u_{i,0} \quad \dots \quad u_{10,0}) = (20,0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 10,0),$$

де  $u_{0,0} = u(0,0) = 20,0$  град;

$$u_{i,0} = u(x_i, 0) = 0,0 \text{ град};$$

$$u_{10,0} = u(2,0; 0) = 20,0 \text{ град}.$$

Значення вектора температурного розподілу для другого часового шару, в момент часу  $t_1 = 0,02$  с на основі рівняння явної різницевої схеми (4.26) за

того самого значення  $j = 0$ , для  $\delta = \alpha \frac{\tau}{h^2} = 0,00011 \frac{0,02}{0,2} = 5,49 \cdot 10^{-5}$

( $\alpha = 0,00011$  м/с<sup>2</sup> – коефіцієнт теплопровідності для міді):

$$U_1^a = (u_{0,1}^a \quad u_{1,1}^a \quad \dots \quad u_{i,1}^a \quad \dots \quad u_{10,1}^a) = (20,0 \quad 0,0011 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0,00055 \quad 10,0),$$

де  $u_{0,1}^a = u(0; 0,02) = 20,0$  град;  $u_{10,1}^a = u(2,0; 0,02) = 20,0$  град;

$$u_{i,j+1}^a = \delta u_{i+1,j} + (1 - 2\delta)u_{ij} + \delta u_{i-1,j}; \quad u_{1,1}^a = \delta u_{2,0} + (1 - 2\delta)u_{1,0} + \delta u_{0,0} = 5,49 \cdot 10^{-5} \times \\ \times 0,0 + (1 - 2 \cdot 5,49 \cdot 10^{-5}) \cdot 0,0 + 5,49 \cdot 10^{-5} \cdot 20,0 = 0,00011 \text{ град}, \quad u_{9,1}^a = \delta u_{10,0} + \\ + (1 - 2\delta)u_{9,0} + \delta u_{8,0} = 5,49 \cdot 10^{-5} \cdot 10,0 + (1 - 2 \cdot 5,49 \cdot 10^{-5}) \cdot 0,0 + 5,49 \cdot 10^{-5} \cdot 0,0 = \\ = 0,00055 \text{ град}.$$

Аналогічно на основі рівняння явної різницевої схеми (4.26) визначаються вектори температурного розподілу для решти часових шарів ( $j = 2, 3, \dots, n_x^a - 1$ ), зокрема для останнього часового шару:

$$U_{180000}^a = (u_{0,180000}^a \quad u_{1,180000}^a \quad \dots \quad u_{i,180000}^a \quad \dots \quad u_{10,180000}^a) = \\ = (20,0 \quad 16,74 \quad 13,71 \quad 11,12 \quad 9,12 \quad 7,81 \quad 7,21 \quad 7,25 \quad 7,84 \quad 8,82 \quad 10,0).$$

Використовуючи рівняння неявної різницевої схеми (4.27), кількість обчислень можна значно скоротити, а саме для другого часового шару за значення  $j = 1$  для кожного розрахункового вузла:

$$\begin{cases} \delta u_{0,1}^u - (1 + 2\delta)u_{1,1}^u + \delta u_{2,1}^u = -u_{1,0} & (i = 1); \\ \delta u_{1,1}^u - (1 + 2\delta)u_{2,1}^u + \delta u_{3,1}^u = -u_{2,0} & (i = 2); \\ \dots \dots \dots; \\ \delta u_{(k-1),1}^u - (1 + 2\delta)u_{k,1}^u + \delta u_{(k+1),1}^u = -u_{k,0}; & (i = k); \\ \dots \dots \dots; \\ \delta u_{(n_x-3),1}^u - (1 + 2\delta)u_{(n_x-2),1}^u + \delta u_{(n_x-1),1}^u = -u_{(n_x-2),0} & (i = n_x - 2). \end{cases} \quad (4.29)$$

Якщо систему рівнянь (4.29) звести до такого вигляду:

$$\left\{ \begin{array}{l} a_{1,1}u''_{0,1} + b_{1,1}u''_{1,1} + c_{1,1}u''_{2,1} = d_1 \quad (i=1); \\ a_{2,1}u''_{1,1} + b_{2,1}u''_{2,1} + c_{2,1}u''_{3,1} = d_{2,1} \quad (i=2); \\ \dots\dots\dots; \\ a_{k,1}u''_{(k-1),1} + b_{k,1}u''_{k,1} + c_{k,1}u''_{(k+1),1} = d_{k,1} \quad (i=k); \\ \dots\dots\dots; \\ a_{n_x-2,1}u''_{(n_x-3),1} + b_{n_x-2,1}u''_{(n_x-2),1} + c_{n_x-2,1}u''_{(n_x-1),1} = d_{n_x-2,1} \quad (i=n_x-2). \end{array} \right. \quad (4.30)$$

де  $a_i = (a_{1,1}, a_{2,1}, \dots, a_{k,1}, \dots, a_{(n_x-2),1}) = (0, \delta, \dots, \delta, \dots, \delta)$ ;  $b_i = (b_{1,1}, b_{2,1}, \dots, b_{k,1}, \dots, b_{(n_x-2),1}) = (-(1+2\delta), -(1+2\delta), \dots, -(1+2\delta), \dots, -(1+2\delta))$ ;  
 $c_i = (c_{1,1}, c_{2,1}, \dots, c_{k,1}, \dots, c_{(n_x-2),1}) = (\delta, \delta, \dots, \delta, \dots, 0)$ ;  $d_i = (d_{1,1}, d_{2,1}, \dots, d_{k,1}, \dots, d_{(n_x-2),1}) = ((-u_{1,0} - \delta u_{0,1}), (-u_{2,0}), \dots, (-u_{k,0}), \dots, (-u_{(n_x-2),0} - \delta u_{(n_x-1),1}))$ ,

то в отриманій тридіагональній матриці системи рівнянь (4.30) виконується умова переваги діагональних елементів ( $|b_{i,1}| \geq |a_{i,1}| + |c_{i,1}|$ ), що дозволяє використати метод прогонки (див. розд. 1) для розв'язання цієї СЛАР.

Тоді значення прогоночних коефіцієнтів на прямому ході:

$$- \xi_{i,1} = \frac{-c_{i,1}}{b_{i,1} + a_{i,1}\xi_{(i-1),1}} \quad (i = 1, 2, \dots, n_x-2)$$

$$\xi_{1,1} = \frac{-c_{1,1}}{b_{1,1} + a_{1,1}\xi_{0,1}} = \frac{-\delta}{-(1+2\delta) + 0 \cdot 0} = \frac{-5,49 \cdot 10^{-5}}{-(1+2 \cdot (5,49 \cdot 10^{-5}))} = 0.000275,$$

$$\xi_{2,1} = \frac{-c_{2,1}}{b_{2,1} + a_{2,1}\xi_{1,1}} = \frac{-5,49 \cdot 10^{-5}}{-(1+2 \cdot (5,49 \cdot 10^{-5})) + 5,49 \cdot 10^{-5} \cdot 0,000275} = 0.000275,$$

$$\dots\dots\dots, \\ \xi_i = (\xi_{1,1}, \xi_{2,1}, \dots, \xi_{k,1}, \dots, \xi_{n_x-1,1}) = (0,000275 \ 0,000275 \ 0,000275 \ 0,000275 \\ 0,000275 \ 0,000275 \ 0,000275 \ 0,000275 \ 0,0);$$

$$- \eta_{i,1} = \frac{d_{i,1} - a_{i,1}\eta_{(i-1),1}}{b_{i,1} + a_{i,1}\xi_{(i-1),1}} \quad (i = 1, 2, \dots, n_x-2)$$

$$\eta_{1,1} = \frac{d_{1,1} - a_{1,1}\eta_{0,1}}{b_{1,1} + a_{1,1}\xi_{0,1}} = \frac{(-u_{1,0} - \delta u_{0,1}) - 0 \cdot 0}{-(1+2\delta) + 0 \cdot 0} = \frac{-20,0 - (5,49 \cdot 10^{-5}) \cdot 20,0}{-(1+2(5,49 \cdot 10^{-5}))} = 0.0055,$$

$$\eta_{2,1} = \frac{d_{2,1} - a_{2,1}\eta_{1,1}}{b_{1,2} + a_{2,1}\xi_{1,1}} = \frac{(-u_{2,0}) - \delta \eta_{1,1}}{-(1+2\delta) + \delta \xi_{1,1}} =$$



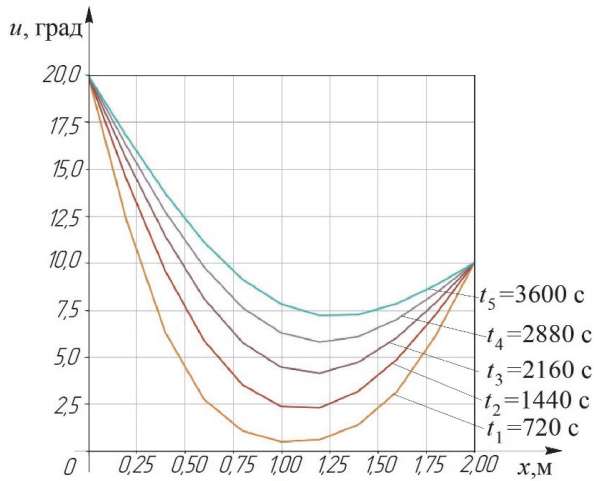


Рисунок 4.12 – Діаграма розподілу температури вздовж стержня

Результат розв’язання прикладу 4.3 мовою програмування PYTHON:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d
from scipy import *
from sympy import *
#введення значення довжини стержня
l=float(input()) #l=2.0
#введення значення часу тривалості теплообміну
t=float(input()) #t=3600.0
#введення значення кроку розрахункової сітки по довжині стержня
h_x=float(input()) #h_x=0.2
#введення значення кроку розрахункової сітки по часу
print("Введіть значення кроку розрахункової сітки по часу для явної
різницевої схеми (крок не має бути більше)", 0.5*h_x*h_x, "с")
h_texp=float(input()) #h_texp=0.02
print("Введіть значення кроку розрахункової сітки по часу для неявної
різницевої схеми")
h_tnexr=float(input()) #h_tnexr=0.1
#введення значення коефіцієнту теплопровідності
a=0.00011 #a=float(input()) a=0.00011
del_tae=(a*h_texp)/(h_x*h_x); del_tane=(a*h_tnexr)/(h_x*h_x)
#введення значення крайових умов:
#введення значення температури лівого торця стержня
def u_left(t):
    return 20.0
#введення значення температури правого торця стержня
def u_right(t):
    return 10.0
#введення функції початкового розподілу температури по довжині стержня
def u_x0(x):
    return 0
#введення значень крайових і початкових умов:
u_first,x_coord=[u_left(0)], [0]
```

```

x=h_x
for i in range(0,int(1/h_x)):
#для першого часового шару
    u_first.append(u_x0(x)); x_coord.append(x)
    x=x+h_x
u_first[int(1/h_x)]=u_right(0)
ru_first=np.array(u_first)
u_exp=np.array([u_first]); u_nexp=np.array([u_first])
#розрахунок розподілу температури по стержню за допомогою явної різничевої
# схеми
t_0=0
for j in range(1,int(t/h_texp)+1):
    t_0=t_0+h_texp; u_next=[u_left(t_0)]
    for i in range(1,int(1/h_x)):
        u_next.append((del_tae*u_exp[j-1,i-1])+
            ((1-(2*del_tae))*u_exp[j-1,i])+
            (del_tae*u_exp[j-1,i+1]))
        u_next.append(u_right(t_0))
        u_exp=np.insert(u_exp,j,u_next,axis=0)
        del u_next
print("Матриця температурного розподілу вздовж стержня по часовим шарам
    визначених неявним різницевим методом:")
print(u_exp)
#визначання вектора температурного розподілу вздовж стержня в поточному
    часовому шарі неявним різницевим методом
def u_rawnexp(del_tane,h_tnexp, k):
    a_kof=[0]; b_kof=[-1-(2*del_tane)]; c_kof=[del_tane]
    d_kof=[-u_nexp[k,1]-(del_tane*u_left(h_tnexp))]
    for i in range(2,int(1/h_x)):
        a_kof.append(del_tane)
        b_kof.append((-1-(2*del_tane)))
        if i==int(1/h_x)-1:
            c_kof.append(0)
            d_kof.append((-u_nexp[k,int(1/h_x)-1]-
                (del_tane*u_right(h_tnexp))))
        else:
            c_kof.append(del_tane)
            d_kof.append(-u_nexp[k,i])
    et_ta,tet_ta=[-c_kof[0]/b_kof[0],[d_kof[0]/b_kof[0]]
    for i in range(1,int(1/h_x)-1):
        et_ta.append((-c_kof[i]/(b_kof[i]+(a_kof[i]*et_ta[i-1]))))
        tet_ta.append(((d_kof[i]-(a_kof[i]*tet_ta[i-1]))/
            (b_kof[i]+(a_kof[i]*et_ta[i-1]))))
    u=[tet_ta[int(1/h_x)-2]]
    for i in range(int(1/h_x)-3,-1,-1):
        u.append((et_ta[i]*u[int(1/h_x)-3-i])+tet_ta[i])
    u=list(reversed(u))
    u.append(u_right(h_tnexp))
    u.insert(0,u_left(h_tnexp))
    return u
#визначання матриці розподілу температурного розподілу вздовж стержня за
    допомогою неявної різничевої схеми
for k in range(0,int(t/h_tnexp)-1):
    u_nexp=np.insert(u_nexp,k+1,u_rawnexp(del_tane,h_tnexp, k),axis=0)
print(u_nexp)
print(len(u_nexp))
#побудова діаграми температурного розподілу вздовж стержня по часовим шарам
plt.figure(figsize=(8, 8))
plt.xlabel('x',fontsize=15, color='blue')
plt.ylabel('y',fontsize=15, color='blue')
for k in range(1,6,2):
    plt.plot(x_coord, u_exp[36000*k])
    plt.plot(x_coord, u_nexp[1440*k])
plt.grid(True)
plt.xlim([0, 2])
plt.ylim([0, 20])
plt.show().

```



## **Висновки щодо застосування методів чисельного розв'язання диференціальних рівнянь математичної фізики**

Із диференціальними рівняннями в частинних похідних пов'язано розв'язок інженерних задач у багатьох сферах науки і техніки. У них містяться частинні похідні, і шукана функція одночасно залежить від декількох змінних.

Повна математична постановка задачі нарівні із диференціальними рівняннями в частинних похідних містить також деякі додаткові умови. Якщо пошук розв'язку виконується в обмеженій області, тоді задаються граничні (крайові) умови і тоді задача називається крайовою задачею для рівнянь із частинними похідними. Задача, в якій необхідно розв'язати диференціальне рівняння в частинних похідних за заданих початкових умов називається задачею Коші. Задача розв'язується в необмеженому просторі, і крайові (граничні) умови не задаються. Задачі, під час формулювання яких виконується постановка одночасно крайових і початкових умов, називаються нестационарними (або змішаними) крайовими задачами. Розв'язок, який отримується, змінюється протягом часу.

Залежно від значень функціональних коефіцієнтів та їх співвідношень розрізняють різні види диференціальних рівнянь математичної фізики: переносу, еволюційне, гіперболічне, параболічне та еліптичне.

Існують два види методів розв'язання рівнянь таких типів:

- аналітичний (результат виводиться різними математичними перетвореннями),

- чисельний, за якого отриманий результат відповідає дійсному із заданою точністю, але який потребує багато алгебраїчних обчислень і використання обчислювальних потужностей комп'ютерних систем.

Порівнюючи методи розв'язання диференціальних рівнянь у частинних похідних необхідно пам'ятати, що в методі кінцевих елементів апроксимується розв'язок задачі, тоді як у методі кінцевих різниць – похідні шуканих функцій. Метод кінцевих елементів, на відміну від різницевого метода, потребує трудомісткої постановки задачі, високої кваліфікації і досвіду дослідника, проте зручний під час розв'язання задачі із складною формою границі, неоднорідним розподілом параметрів.

На початковому етапі розв'язання диференціальних рівнянь у частинних похідних вибирається метод розв'язання задачі. Зазвичай простіше використовувати метод кінцевих різниць, який потребує більш простої підготовки задачі до розв'язання, проте у ряді випадків для задач із добре розробленою теорією (наприклад, задач механіки) доцільно звертатись до методу кінцевих елементів.

Під час визначення кроку розв'язання задачі основним фактором є точність: якщо висока, тоді необхідні або дуже дрібна сітка, або розбиття на дуже дрібні елементи. Водночас необхідно враховувати, що похибка кінцево-різницевих методів має перший порядок, тобто співрозмірна із  $h^2$ .

У випадку симетрії в області розв'язку можна число вузлів зменшити в два або чотири рази за симетрії по обох осях координат. Це дозволяє зекономити час і обсяг пам'яті комп'ютерної системи.

Велике значення для ефективного розв'язання задачі має вибір початкових значень змінних, від цього в процесі використання ітераційних методів суттєво залежить швидкість збіжності результатів обчислення. Часто доцільно розв'язувати задачу в декілька етапів: на першому етапі за допомогою грубої сітки (або розбиття на крупні елементи) отримують початкове вихідне наближення, після чого виконують точне розв'язання на дрібній сітці.

Для розв'язання диференціальних рівнянь у частинних похідних розроблено ряд сучасних ефективних програмних засобів, що використовуються в задачах автоматизації проектування технологічних систем.

### **Контрольні запитання та завдання**

1. Дайте означення диференціальних рівнянь у частинних похідних.
2. Які види необхідних умов потрібно мати, щоб отримати розв'язок диференціальних рівнянь у частинних похідних?
3. Наведіть приклади інженерних задач, які описуються диференціальними рівняннями в частинних похідних.
4. Яким узагальненим функціональним рівнянням характеризуються усі види диференціальних рівнянь у частинних похідних?
5. Які існують види диференціальних рівнянь другого порядку в частинних похідних залежно від вхідних функціональних коефіцієнтів?
6. Наведіть приклади інженерних задач, які описуються відповідними типами диференціальних рівнянь математичної фізики.
7. Визначіть тип диференціальних рівнянь у частинних похідних, наведених у таблиці 4.2.

Таблиця 4.2 – Вихідні дані до завдання

Варіант	Диференціальне рівняння
1	$\frac{\partial^2 u}{\partial t^2} + 6 \frac{\partial^2 u}{\partial t \partial x} = 0$
2	$\frac{\partial^2 u}{\partial t^2} - 6 \frac{\partial^2 u}{\partial t \partial x} + 9 \frac{\partial^2 u}{\partial^2 x} = 0$
3	$\frac{\partial^2 u}{\partial y^2} - 6 \frac{\partial^2 u}{\partial x \partial y} + 9 \frac{\partial^2 u}{\partial^2 x} = 0$
4	$\frac{\partial^2 u}{\partial t \partial x} - 3 \frac{\partial u}{\partial t} + 5 \frac{\partial u}{\partial x} = u$
5	$\frac{\partial^2 u}{\partial x^2} - 2x \frac{\partial^2 u}{\partial x \partial y} + \frac{\partial^2 u}{\partial y^2} = xu^2$
6	$\frac{\partial^2 u}{\partial x^2} + 10 \frac{\partial^2 u}{\partial x \partial y} + 25 \frac{\partial^2 u}{\partial y^2} - xy^2 u = 0$

8. Які існують методи розв'язання диференціальних рівнянь у частинних похідних? Дайте їх порівняльну характеристику.

9. Охарактеризуйте рівняння теплопровідності.

10. Як конструюються обчислювальні шаблони для частинних похідних?

11. Складіть обчислювальні шаблони для оператора Лапласа і бігармонічного оператора.

12. Складіть обчислювальні шаблони для диференціальних рівнянь у частинних похідних, наведених у таблиці 4.3.

Таблиця 4.3 – Вихідні дані до завдання

Варіант	Диференціальне рівняння
7	$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y \partial x} = 0$
8	$\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial t \partial x} + \frac{\partial^2 u}{\partial^2 x} = 0$
9	$\frac{\partial^2 u}{\partial y^2} - \frac{\partial^3 u}{\partial^3 x} = 0$
10	$\frac{\partial^2 u}{\partial t \partial x} - \frac{\partial^4 u}{\partial x^4} = u(x, t)$
11	$\frac{\partial^2 u}{\partial x^2} - x \frac{\partial^2 u}{\partial x \partial y} + \frac{\partial^2 u}{\partial y^2} = xu(x, y)$
12	$\frac{\partial u}{\partial x} + \frac{\partial^2 u}{\partial x^2} + \frac{\partial^3 u}{\partial y^3} + \frac{\partial^2 u}{\partial x \partial y} = 0$

13. Якими різницевиими методами обчислюється класична задача Діріхле для рівняння Лапласа в прямокутній області?

14. Охарактеризуйте хвильове рівняння.

15. Побудуйте алгоритм розв'язання еліптичних рівнянь у частинних похідних.

16. Складіть тришарову схему розв'язку гіперболічних рівнянь.

17. Яку необхідну і достатню умову стійкості необхідно витримувати для чисельного розв'язання гіперболічних рівнянь у частинних похідних?

18. Побудуйте алгоритм розв'язання гіперболічних рівнянь у частинних похідних.

19. Порівняйте ефективність використання явної і неявної схем розв'язку параболічних рівнянь у частинних похідних.

20. Побудуйте алгоритм розв'язання параболічних рівнянь у частинних похідних явним різницевим методом.

21. Побудуйте алгоритм розв'язання параболічних рівнянь у частинних похідних неявним різницевим методом.

22. За яких умов явна і неявна двошарові різницеві схеми для розв'язання параболічних рівнянь у частинних похідних є стійкими?

23. Яким чином обирається крок під час розв'язання диференціальних рівнянь у частинних похідних?

24. Наведіть загальний алгоритм розв'язку диференціальних рівнянь у частинних похідних різницевим методом.

25. Визначіть стаціонарний розподіл температури в прямокутній пластині розмірами  $l \times h = 2,0 \times 1,5$  м, для якої задано такі граничні умови (табл. 4.4):  $u(0, y) = \varphi(y)$ ,  $u(l, y) = \Psi(y)$ ,  $u(x, 0) = \mu_1(x)$ ,  $u(x, h) = \mu_2(x)$  ( $0 \leq x \leq l$ ,  $0 \leq y \leq h$ ). Побудуйте діаграму стаціонарного температурного розподілу у пластині.

Таблиця 4.4 – Вихідні дані до завдання

Варіант	Функції початкових умов		Функції граничних умов	
	$\varphi(y)$	$\Psi(y)$	$\mu_1(x)$	$\mu_2(x)$
13	$\cos(0,5y)$	$e^{-4}\cos(0,5y)$	$e^{-x}$	$e^{-x}\cos(0,5x)$
14	$e^{2y}(2y^2+1)$	$e^{2y}(2y^2+3)$	$e^6(x+19)$	$x+1,0$
15	$\operatorname{tgy}$	$\operatorname{tg}^2(y+0,9)$	$\operatorname{tg}^2(x+0,4)$	$\operatorname{tg}^2(x)$
16	0,0	$\cos y$	$5\cos x + \sin x$	$10\sin x + x\sin(1,0)$
17	$-\cos y$	$62\cos(y^2)$	$2x^3+3x-1$	$(15x^2+1)\cos(x)$
18	$5y(y-1)$	$11y(y-1)$	$12(x^2-x)+60$	0,0

26. Потрібно визначити положення струни залежно від часу (загальний час коливального процесу  $T$ ), яка виконує вільні коливання із закріпленими кінцями (відстань між точками закріплення кінців  $l = 4,0$  м) із початковими

умовами  $u(x, 0) = \varphi(x)$ ,  $\frac{\partial u(x, 0)}{\partial t} = \psi(x)$  ( $0 \leq x \leq l$ ) і граничними умовами  $u(0, t) = \mu_1(t)$ ,  $u(l, t) = \mu_2(t)$  ( $0 \leq t \leq T$ ) (табл. 4.5). Коефіцієнт фазової швидкості  $a=1,2$  м/с. Побудуйте графік положень коливальної струни для різних моментів часу.

Таблиця 4.5 – Вихідні дані до завдання

Ва-ріант	Функції початкових умов		Загальний час коливання $T$ (с)	Функції граничних умов	
	$\varphi(x)$	$\psi(x)$		$\mu_1(t)$	$\mu_2(t)$
19	$0,5(x+1)^2$	$(x+0,5)\cos(\pi x)$	1,0	0,5	$2,0-3t$
20	$x^2\cos(\pi x)$	$x^2(x+1)$	0,2	$0,5t$	$t-1,0$
21	$x+1$	0,0	1,2	$0,5t^2$	$t+1,0$
22	$e^x\sin x$	$e^x(\cos x+\sin x)$	2,0	$\cos^2 t+\sin t$	$\cos^2(t+6,0)+\sin x$
23	$\cos^2 x+\sin x$	$2x\sin(2x)$	2,0	$e^t\sin t$	$e^{t+2}\sin(t+2,0)$
24	$\cos^2(0,5x)$	$-0,5\sin x$	6,0	$\cos^2(0,5t)$	$\cos^2(0,5t+3,0)$

27. Необхідно визначити зміну розподілу температури вздовж стержня залежно від часу протягом часу  $t_0 = 30,0$  хв. Стержень довжиною  $L$  із постійною по довжині площею поперечного перерізу розміщений в ізольованому матеріалі таким чином, що з навколишнім середовищем взаємодіють тільки його крайні правий і лівий торці. У початковий момент часу стержень має урівноважену температуру  $u(x, 0) = \varphi(x)$ , а на його лівому і правому торцях постійно підтримується температура  $u(0, t) = \mu_1(t)$ ,  $u(l, t) = \mu_2(t)$  ( $0 \leq t \leq T$ ) (табл. 4.6). Побудуйте діаграму температурного розподілу стержня для різних моментів часу.

Таблиця 4.6 – Вихідні дані до завдання

Ва-ріант	Функція початкових умов $\varphi(x)$	Довжина стержня, $l$ (м)	Функції граничних умов		Матеріал стержня	Вид різнищевого методу
			$\mu_1(t)$	$\mu_2(t)$		
25	$x^2$	1,3	$50\sin t$	$40\cos^2 t$	Сталь	Неявний
26	15,0	2,2	$20\cos t+10\sin t$	80,0	Мідь	Явний
27	$2x+5$	1,5	$0,05t+5,0$	$0,01t-15,0$	Срібло	Неявний
28	$10\sin x$	1,4	$2e^{0,001t}+3,0$	$0,01t+20\sin t$	Графіт	Явний
29	$5e^x$	2,5	$-30,0$	$70\sin t$	Чавун	Неявний
30	$\cos(x)+10\sin(x)$	3,0	$35\sin^2(0,3t)$	$-20,0$	Скло	Явний

28. Складіть обчислювальний шаблон оператора Лапласа для трьох координат.

29. Поперечна деформація  $w$  тонкої прямокутної пластини розмірами  $l \times h = 3,0 \times 1,7$  м, рівномірно навантаженої тиском  $p$ , визначається рівнянням:

$$\Delta^2 w = p/D, \quad (4.31)$$

де  $D = \frac{Et^3}{12(1-\nu^2)}$  – згинальна жорсткість,  $E = 2,1 \cdot 10^{11}$  МПа – модуль пружності,  $\nu = 0,3$  – коефіцієнт Пуассона,  $t = 0,005$  м – товщина пластини,  $p = 100,0$  Па – тиск, яким рівномірно навантажена пластинка. Визначіть розподіл деформації пластини  $w(x, y)$  залежно від заданого навантаження  $p$ .

30. Рівняння Нав'є-Стокса, яке описує встановлений рух в'язкої рідини в трубі довільного поперечного перерізу, має вигляд:

$$\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} = \frac{1}{\mu} \frac{dp}{dz}, \quad (4.32)$$

де  $v$  – модуль швидкості рідини (на стінках труби дорівнює нулю),  $\mu$  – коефіцієнт в'язкості,  $dp/dz$  – похідна тиску по довжині труби. Взявши  $dp/dz = -5000,0$  Па/м і  $\mu = 1,5 \cdot 10^{-4}$  Н·с/м<sup>2</sup>, визначіть розподіл модуля швидкості в поперечному перерізі труби, показаному на рисунку 4.13.

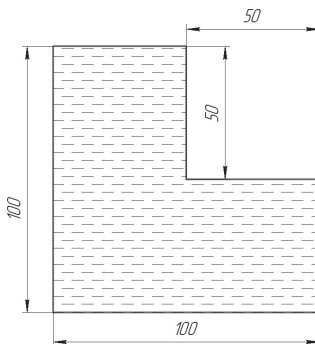


Рисунок 4.13 – Схема поперечного перерізу трубопроводу

## РОЗДІЛ 5 ЗАДАЧІ ОБРОБКИ ДАНИХ

Дуже важливою задачею в процесі ідентифікації математичних моделей є обробка експериментальних даних, отриманих під час активних або пасивних ідентифікаційних експериментів. Для цього застосовуються різноманітні методи та алгоритми, вибір яких зумовлено видом об'єкта моделювання, моделі та наявними обчислювальними потужностями комп'ютерних засобів. Серед головних методів обробки даних виділимо інтерполяцію, апроксимацію та статистичну обробку даних.

### 5.1 Інтерполяція

Метою інтерполяції (*interpolation*) є побудова функції  $F(x)$  (інтерполянта) із заданого класу функцій, яка набуває в окремих точках  $x_i \in [a; b]$  ( $i=0, 1, 2, \dots, n$ ) (вузли інтерполяції, а їх сукупність – інтерполяційна сітка) значень:

$$F(x_0) = y_0, F(x_1) = y_1, \dots, F(x_i) = y_i, \dots, F(x_n) = y_n, \quad (5.1)$$

що збігаються з раніше заданими (наприклад, отриманими з експерименту) значеннями в цих точках невідомої функції  $y = f(x)$ . Геометрично це означає, що необхідно визначити криву  $y = F(x)$  певного типу, яка проходить через систему точок  $M(x_i, y_i)$  ( $i = 0, 1, 2, \dots, n$ ).

У загальних випадках у цієї задачі можливі як нескінченна множина розв'язків так і зовсім відсутні розв'язки, але вона стає однозначною, якщо замість довільної функції  $F(x)$  шукати поліном  $P_n(x)$  не вище  $n$ -ого степеня, який задовольняє умову (5.1), тобто:

$$P_n(x_0) = y_0, P_n(x_1) = y_1, \dots, P_n(x_i) = y_i, \dots, P_n(x_n) = y_n.$$

Інтерполяційну формулу  $Y = F(x)$  використовують для наближеного обчислення значень функції  $f(x)$  для  $x \neq x_i$  ( $i = 0, 1, 2, \dots, n$ ). Потрібно відзначити, що є інтерполяція в вузькому розумінні, коли  $x \in [x_0; x_n]$ , та екстраполяція, коли  $x$  знаходиться за межами інтервалу  $[x_0; x_n]$  тобто  $x < x_0$  чи  $x > x_n$ .

Аналізуючи процедуру інтерполювання, обов'язково потрібно вказати обмеження, які накладаються на набір базових точок  $(x_i, y_i)$ . Початкова інтерполяційна сітка точок має описувати лише гладку функцію. Відповідно до умов конкретної задачі обов'язково мають задаватися значення похідної функції у крайових точках вхідної інтерполяційної сітки для отримання однозначного результату.

До основних методів інтерполяції належать:

1. Лінійна інтерполяція (*linear interpolation*). Найбільш простий і швидкий метод, в якому задані вузлові точки  $(x_i, y_i)$  з'єднуються прямими.

2. Інтерполяція із застосуванням многочлена (*interpolation using*

*polynomial*). Використовується многочлен  $n$ -го порядку, який у загальному випадку має вигляд:

$$P(x) = P_n(x) = a_0x^n + a_1x^{n-1} + \dots + a_{n-1}x + a_n,$$

де  $a_i$  ( $i=0, 1, \dots, n$ ) – постійні коефіцієнти.

Усі методи знаходження інтерполяційного многочлена зводяться до отримання постійних коефіцієнтів. До таких методів належать:

- а) інтерполяція різницевиими методами;
- б) інтерполяція за Лагранжем;
- в) Ермітова поліноміальна інтерполяція.

3. Поліноміальна інтерполяція сплайнами (*polynomial spline interpolation*). Вузлові точки з'єднуються з використанням многочлена заданого порядку, який обирається залежно від методу. До найбільш поширених методів інтерполяції сплайнами належать:

- а) класичні кубічні сплайни;
- б) Ермітові сплайни;
- в) В-сплайни;
- г) криві Без'є.

У цьому розділі розглядається алгоритмізація та застосування найбільш поширених у інженерній практиці методів інтерполяції.

### 5.1.1 Різницеві методи

Існує багато відомих кінцево-різницевих методів інтерполяції. Найбільш поширеним є метод Ньютона для інтерполяції «уперед» (метод Ньютона–Грегорі). Інтерполяційний поліном у цьому випадку має вигляд:

$$P_n(x) = C_0 + C_1(x - x_0) + C_2(x - x_0)(x - x_1) + \dots \\ \dots + C_n(x - x_0)(x - x_1) \dots (x - x_{n-1}).$$

Коефіцієнти  $C_i$  визначаються із рівнянь:

$$P_n(x_i) = y_i \quad (i=0, 1, 2, \dots, n),$$

які дозволяють записати систему:

$$\begin{cases} C_0 = y_0, \\ C_0 + C_1(x_1 - x_0) = y_1, \\ C_0 + C_1(x_2 - x_0) + C_2(x_2 - x_0)(x_2 - x_1) = y_2, \\ \dots \\ C_0 + C_1(x_n - x_0) + \dots + C_n(x_n - x_0)(x_n - x_1) \dots (x_n - x_{n-1}) = y_n. \end{cases} \quad (5.2)$$



Рівняння (5.2) є СЛАР із трикутною матрицею. Якщо взяти крок  $x_{i+1}-x_i = h$ , то в області зміни значень вузлів інтерполяції  $x \in [x_0; x_n]$  буде отримано одновимірну рівномірну інтерполяційну сітку, що дозволить скористатись різницеvim зображенням системи (5.2), яке приводить до таких різницеvих виразів для визначення коефіцієнтів:

$$C_0 = y_0, \quad C_1 = \frac{y_1 - y_0}{h} = \frac{\Delta y_0}{h},$$

де  $\Delta y_0$  – права різниця першого порядку в точці  $y_0$ ;

$$C_2 = \frac{y_2 - 2y_1 + y_0}{2h^2} = \frac{\Delta^2 y_0}{2h^2},$$

де  $\Delta^2 y_0$  – права різниця другого порядку;

$$C_j = \frac{\Delta^j y_0}{(j!)h^j},$$

де  $\Delta^j y_0$  – права різниця  $j$ -го порядку.

Тоді рівняння (5.2) може бути записано, як:

$$\begin{aligned} P_n(x) = & y_0 + \frac{\Delta y_0}{1!h}(x - x_0) + \frac{\Delta^2 y_0}{2!h^2}(x - x_0)(x - x_1) + \dots \\ & \dots + \frac{\Delta^n y_0}{n!h^n}(x - x_0)(x - x_1) \dots (x - x_{n-1}). \end{aligned} \quad (5.3)$$

З практичного погляду для визначення різниць вищих порядків використовуються вираз:

$$\Delta^j y_i = \Delta(\Delta^{j-1} y_i) = \Delta^{j-1} y_{i+1} - \Delta^{j-1} y_i, \quad (i=0, 1, 2, \dots, n-j).$$

У разі, коли  $n = 1$  із (5.3) отримуємо формулу для лінійної інтерполяції:

$$P_1(x) = y_0 + \frac{\Delta y_0}{h}(x - x_0),$$

а якщо  $n=2$  – формулу параболічної чи квадратичної інтерполяції:

$$P_2(x) = y_0 + \frac{\Delta y_0}{h}(x - x_0) + \frac{\Delta^2 y_0}{2h^2}(x - x_0)(x - x_1).$$

Якщо задано необмежену кількість значень функції  $x$ , то число  $n$  може бути будь-яким. Практично в цьому випадку  $n$  вибирають таким, щоб різниця  $\Delta^n y_i$  була постійною із заданою точністю. За початкове значення  $x_0$  можна брати будь-яке табличне значення аргументу  $x$ . Коли кількість значень функції скінченна, то кількість  $n$  обмежена та не може бути більше кількості значень функції  $y$ , зменшеної на одиницю.

Схема алгоритму реалізації інтерполяції за першою інтерполяційною формулою Ньютона набуде вигляду, який подано на рисунку 5.1.

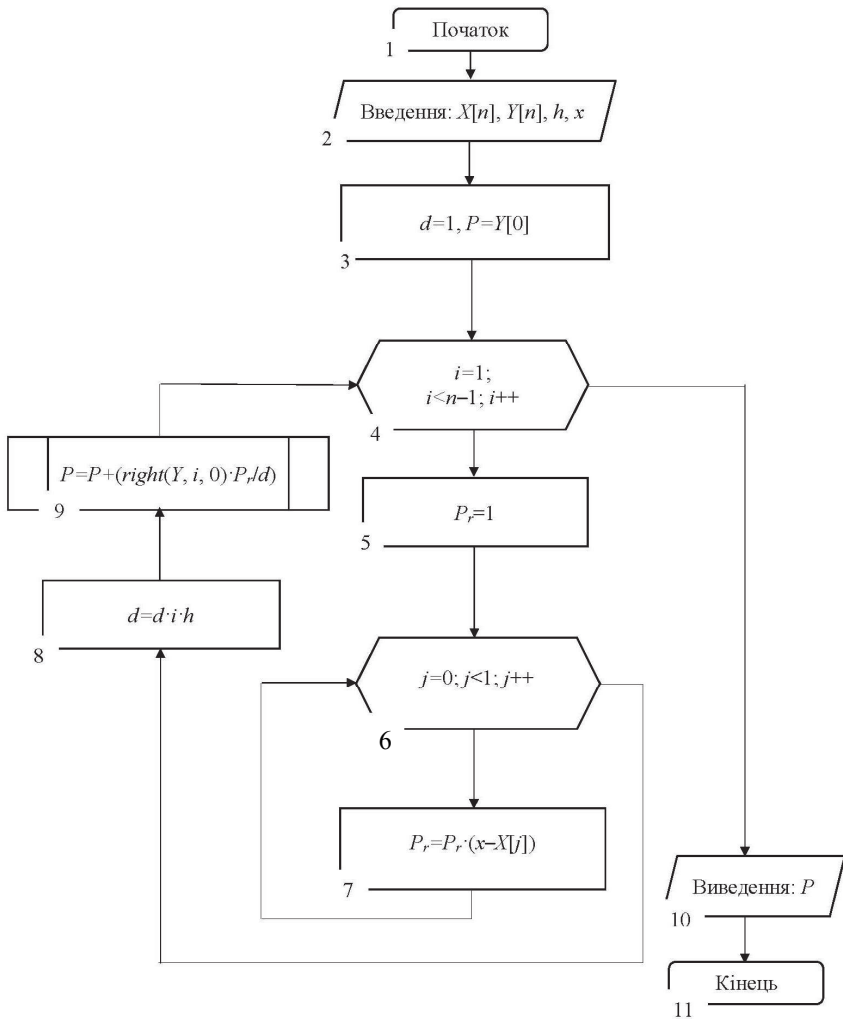


Рисунок 5.1 – Алгоритм методу інтерполяції за першою інтерполяційною формулою Ньютона

Для знаходження різниць необхідно використати рекурсію, для цього складається функція *float right(float y[], int p, int i)*. Алгоритм визначення кінцевих різниць для заданої функції наведено на рисунку 5.2.

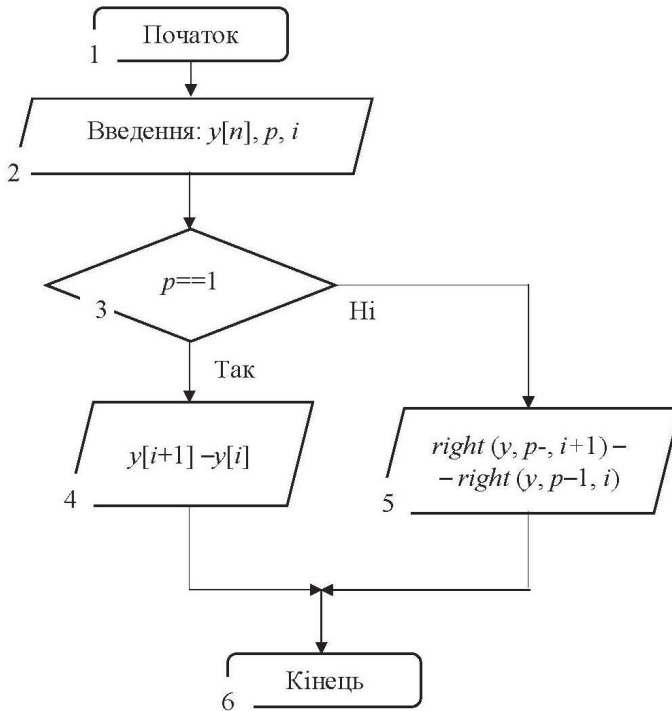


Рисунок 5.2 – Алгоритм визначення скінченних різниць:  
 $y[]$  – масив значень  $y$ ;  $p$  – порядок різниці;  $i$  – порядковий номер змінної  $y$ , для якої обчислюється різниця

Розглянемо процедуру знаходження різниць мовою програмування C++:

```

#include <stdio.h>
#include <math.h>
#include <conio.h>
const int n = 6;
float y (float x)
{return (1/(pow(x,4)+5));}
float right(float y[], int p, int i)
{
  if (p == 1)
  return y[i+1]-y[i];
  else
  return right(y,p-1,i+1)-right(y,p-1,i);
}
void main()
{
float x[6] = {1, 1.1, 1.2, 1.3, 1.4, 1.5};
float y[6] = {0.16667, 0.15470, 0.14137, 0.12729, 0.11310, 0.09938};
for (int i = 1; i<n; i++) {

```

```

for (int j = 0; j<n-i; j++)
printf("%9.5f ", right(y, i, j));
printf("\n");
} }

```

Результат:

```

-0.01197   -0.01333   -0.01408   -0.01419   -0.01372
-0.00136   -0.00075   -0.00011   0.00047
0.00061           0.00064           0.00058
0.00003           -0.00006
-0.00009

```

Реалізація алгоритму методу Ньютона за першою інтерполяційною формулою мовою програмування C++:

```

#include <stdio.h>
#include <math.h>
#include <conio.h>
const int n = 6;
float y (float x)
{return (1/(pow(x,4)+5));}
float right(float y[], int p, int i) //допоміжна функція для знаходження
різниць з масиву Y, порядку різниці p, коефіцієнта i
{
if (p == 1)
return y[i+1]-y[i];
else
{
return right(y,p-1,i+1)-right(y,p-1,i);
}
};
void main()
{
float X[6] = {1, 1.1, 1.2, 1.3, 1.4, 1.5};
float Y[6] = {0.16667, 0.15470, 0.14137, 0.12729, 0.11310, 0.09938};
float h = 0.1, d = 1, Pr;
float x = 1.07;
float P = Y[0];
for (int i = 1; i<n-1; i++)
{
Pr = 1;
for (int j = 0; j<i; j++)
Pr = Pr * (x-X[j]);
d = d * i * h;
P = P + (right(Y, i, 0)*Pr/d);
}
printf("x = %.5f y = %.5f", x, P);
getch();
}.

```

Формула (5.3) носить назву першої інтерполяційної формули Ньютона. Цей вираз незручний для інтерполяції в околі останніх значень  $y_i$ . У цьому випадку, як правило, використовують другу інтерполяційну формулу Ньютона, яка отримана за використання лівих різниць від останнього значення  $(x_n, y_n)$  (інтерполяція «назад»). Тоді інтерполяційний поліном матиме такий вигляд:

$$P_n(x) = C_0 + C_1(x - x_n) + C_2(x - x_n)(x - x_{n-1}) + C_3(x - x_n)(x - x_{n-1})(x - x_{n-2}) + \dots \\ \dots + C_n(x - x_n)(x - x_{n-1}) \dots (x - x_1).$$

Коефіцієнти  $C_j$  визначаються таким чином:

$$C_0 = y_n, \quad C_1 = \frac{\Delta y_{n-1}}{h} = \frac{\nabla y_n}{h},$$

де  $\nabla y_n$  – «ліва» різниця першого порядку в точці  $y_n$ ;

$$C_2 = \frac{\Delta^2 y_{n-2}}{2!h^2} = \frac{\nabla^2 y_n}{2h^2},$$

де  $\nabla^2 y_n$  – «ліва» різниця другого порядку;

$$\dots, \quad C_j = \frac{\Delta^j y_{n-j}}{j!h^j} = \frac{\nabla^j y_n}{j!h^j},$$

де  $\nabla^j y_n$  – ліва різниця  $j$ -го порядку.

Кінцевий вираз для другої інтерполяційної формули Ньютона:

$$P_n(x) = y_n + \frac{\Delta y_{n-1}}{1!h}(x - x_n) + \frac{\Delta y_{n-2}^2}{2!h^2}(x - x_n)(x - x_{n-1}) + \dots \\ \dots + \frac{\Delta^n y_0}{n!h^n}(x - x_n)(x - x_{n-1}) \dots (x - x_1).$$

Інтерполяційні формули Ньютона можуть бути використані для екстраполяції функції. Якщо  $x < x_0$ , то зручно використовувати першу інтерполяційну формулу Ньютона, причому  $\frac{x - x_0}{h} < 0$ .

Якщо  $x > x_n$ , тоді використовують другу інтерполяційну формулу Ньютона, де  $\frac{x - x_n}{h} > 0$ .

Таким чином, перша інтерполяційна формула Ньютона, як правило, використовується для інтерполяції «уперед» та екстраполяції «назад», а друга – для інтерполяції «назад» та екстраполяції «уперед».

У формулах Ньютона використовують «ліві» та «праві» різниці. Використання «центральної» різниці для отримання інтерполяційних формул приводить до формул Гаусса, Стірлінга та Бесселя.

Потрібно відзначити, що «центральної» різниці використовуються не в звичайному вигляді, а шляхом застосування «правих» різниць за поступового зсуву індексів вліво.

Ці формули Ньютона зручно розглядати на  $(2n+1)$  рівновіддалених вузлах інтерполяції:

$$x_{-n}, x_{-(n-1)}, \dots, x_{-1}, x_0, x_1, \dots, x_{n-1}, x_n,$$

причому  $\Delta x_i = x_{i+1} - x_i = h = \text{const}$  ( $i = -n, -(n-1), \dots, n-1$ ), а для функції  $y = f(x)$  відомі її значення в цих вузлах  $y_i = f(x_i)$ .

Нехай необхідно побудувати поліном  $P(x)$  степеня не вище  $2n$  таких, що  $P(x_i) = y_i$ . Тоді поліном  $P(x)$  визначається у вигляді:

$$P(x) = C_0 + C_1(x - x_0) + C_2(x - x_0)(x - x_1) + C_3(x - x_0) \times \\ \times (x - x_0)(x - x_1) + \dots + C_{2n-1}(x - x_{-(n-1)}) \dots \times \\ \times \dots (x - x_{-1})(x - x_0)(x - x_1) \dots (x - x_{n-1})(x - x_n). \quad (5.4)$$

Аналогічно інтерполяційним формулам Ньютона, використовуючи (5.4), визначаються:  $C_0 = y_0$ ;  $C_1 = \frac{\Delta y_0}{h}$ ;  $C_2 = \frac{\Delta^2 y_{-1}}{2!h^2}$ ;  $\dots$ ;  $C_{2n-1} = \frac{\Delta^{2n-1} y_{-(n-1)}}{(2n-1)!h^{2n-1}}$ ;

$$C_{2n} = \frac{\Delta^{2n} y_{-n}}{(2n)!h^{2n}}.$$

Підставляючи знайдені значення коефіцієнтів в (5.4), отримуємо першу інтерполяційну формулу Гаусса, яка містить різниці (табл. 5.1):

$$\Delta y_0, \Delta^2 y_{-1}, \Delta^3 y_{-1}, \Delta^4 y_{-2}, \Delta^5 y_{-2}, \Delta^6 y_{-2}, \dots$$

Аналогічно може бути отримана друга інтерполяційна формула Гаусса, яка містить центральні різниці:

$$\Delta y_{-1}, \Delta^2 y_{-1}, \Delta^3 y_{-2}, \Delta^4 y_{-2}, \Delta^5 y_{-3}, \Delta^6 y_{-3}, \dots$$

Використавши середнє арифметичне першої та другої інтерполяційних формул Гаусса, отримаємо формулу Стірлінга. Ці формули дозволяють вивести інтерполяційну формулу Бесселя. Взагалі використання інтерполяційних формул із центральними різницями доцільно всередині інтервалу, тоді як на його крайових вузлах, як правило, використовують формули Ньютона (див. табл. 5.1).

Похибки інтерполяції для формул Ньютона можна оцінити відповідно для першої та другої формул як:

$$\Delta_n(x) = \frac{q(q-1)\dots(q-n)}{(n+1)!} \Delta^{n+1} y_0;$$

$$\Delta_n(x) = \frac{q(q+1)\dots(q+n)}{(n+1)!} \Delta^{n+1} y_n,$$

$$\text{де } q = \frac{x - x_n}{h}.$$

Для формули Стірлінга:

$$\Delta(x_n) = \frac{\Delta^{2n+1} y_{-(n-1)} + \Delta^{2n+1} y_{-n}}{2(2n+1)!} q(q^2 - 1)(q^2 - 2^2) \dots (q^2 - n^2).$$

Для випадку нерівновіддалених значень аргументу можна отримати інтерполяційні формули, використовуючи визначення поділених різниць.

Таблиця 5.1 – Застосування різницевих формул інтерполяції

$x$	$y$	$\Delta y$	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$	Примітки
$x_{-2}$	$y_{-2}$		$\Delta^2 y_{-3}$		$\Delta^4 y_{-4}$	Друга формула Ньютона
		$\Delta y_{-2}$		$\Delta^3 y_{-3}$		
$x_{-1}$	$y_{-1}$		$\Delta^2 y_{-2}$		$\Delta^4 y_{-3}$	Формула Стірлінга
		$\Delta y_{-1}$		$\Delta^3 y_{-2}$		
$x_0$	$y_0$		$\Delta^2 y_{-1}$		$\Delta^4 y_{-2}$	Формула Бесселя
		$\Delta y_0$		$\Delta^3 y_{-1}$		
$x_1$	$y_1$		$\Delta^2 y_0$		$\Delta^4 y_{-1}$	Перша формула Ньютона
		$\Delta y_1$		$\Delta^3 y_0$		
$x_2$	$y_2$		$\Delta^2 y_1$		$\Delta^4 y_0$	
		$\Delta y_2$		$\Delta^3 y_1$		
$x_3$	$y_3$		$\Delta^2 y_2$		$\Delta^4 y_1$	

Наприклад, відношення  $[x_i, x_{i+1}] = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$  називається поділеною різницею першого порядку, а відношення  $[x_i, x_{i+1}, x_{i+2}] = \frac{[x_{i+1}, x_{i+2}] - [x_i, x_{i+1}]}{x_{i+2} - x_i}$  – поділеною різницею другого порядку.

Поділені різниці порядку  $n$  отримуються із рекурентного відношення:

$$[x_i, x_{i+2}, \dots, x_{i+n}] = \frac{[x_{i+1}, \dots, x_{i+n}] - [x_i, \dots, x_{i+n-1}]}{x_{i+n} - x_i}.$$

Також може бути отримана інтерполяційна формула Ньютона для нерівновіддалених значень аргументу:

$$P(x) = y_0 + [x_0, x_1](x - x_0) + [x_0, x_1, x_2](x - x_0)(x - x_1) + \dots + [x_0, x_1, \dots, x_n](x - x_0)(x - x_1) \dots (x - x_{n-1}).$$

**Приклад 5.1.** Розв'язати задачу інтерполяції першою інтерполяційною формулою Ньютона на основі відомих значень вузлів інтерполяції (табл. 5.2) та знайти значення функції в точці  $x = 1,5$ .

Таблиця 5.2 – Вихідні дані

$i$	0	1	2	3
$x_i$	1	2	3	4
$y_i$	2	5	10	17

*Розв'язання:*

Визначаємо значення різниць різних порядків:

1)  $x_0=1, y_0=2;$

2)  $x_1=2, y_1=5, \Delta y_0 = y_1 - y_0 = 5 - 2 = 3;$

3)  $x_2=3, y_2=10, \Delta y_1 = y_2 - y_1 = 10 - 5 = 5;$

4)  $x_3=4, y_3=17, \Delta y_2 = y_3 - y_2 = 17 - 10 = 7;$

5)  $\Delta^2 y_0 = \Delta y_1 - \Delta y_0 = 5 - 3 = 2;$

6)  $\Delta^3 y_0 = \Delta(\Delta^2 y_0) = \Delta^2 y_1 - \Delta^2 y_0 = (\Delta y_2 - \Delta y_1) - (\Delta y_1 - \Delta y_0) = (7 - 5) - (5 - 3) = 0.$

Визначаємо значення коефіцієнтів інтерполяційного полінома (5.3):

$$C_0 = y_0 = 2; C_1 = \frac{\Delta y_0}{1!h^1} = \frac{3}{1 \cdot 1} = 3; C_2 = \frac{\Delta^2 y_0}{2!h^2} = \frac{2}{1 \cdot 2 \cdot 1^2} = 1;$$

$$C_3 = \frac{\Delta^3 y_0}{3!h^3} = \frac{0}{1 \cdot 2 \cdot 3 \cdot 1^3} = 0.$$

Функція інтерполяційного полінома:

$$P(x) = C_0 + C_1(x - 1) + C_2(x - 1)(x - 2) + C_3(x - 1)(x - 2)(x - 3),$$

$$P(x) = 2 + 3(x - 1) + (x - 1)(x - 2).$$

Значення інтерполяційної функції в точці  $x = 1,5$ :

$$P(1,5) = 2 + 3(1,5 - 1) + (1,5 - 1)(1,5 - 2) = 3,25.$$

**Приклад 5.2.** Розв'язати задачу інтерполяції формулою Ньютона на основі відомих значень вузлів інтерполяції (табл. 5.3) та знайти значення функції в точці  $x = 0,4$ .

Таблиця 5.3 – Вихідні дані

$i$	0	1	2	3
$x_i$	0,0	0,1	0,3	0,5
$y_i$	-0,5	0,0	0,2	1,0



### Розв'язання

Використання інтерполяційного полінома Ньютона. За вихідних даних, наведених у таблиці 5.3, маємо випадок нерівновіддалених вузлів для  $n = 3$ . Тоді значення поділених різниць:

$$[x_0, x_1] = \frac{y_1 - y_0}{x_1 - x_0} = \frac{0,0 - (-0,5)}{0,1} = 5,0;$$

$$[x_1, x_2] = \frac{y_2 - y_1}{x_2 - x_1} = \frac{0,2 - 0,0}{0,3 - 0,1} = 1,0;$$

$$[x_2, x_3] = \frac{y_3 - y_2}{x_3 - x_2} = \frac{1,0 - 0,2}{0,5 - 0,3} = 4,0;$$

$$[x_0, x_1, x_2] = \frac{[x_1, x_2] - [x_0, x_1]}{x_2 - x_0} = \frac{1,0 - 5,0}{0,3 - 0,0} = -\frac{40}{3};$$

$$[x_1, x_2, x_3] = \frac{[x_2, x_3] - [x_1, x_2]}{x_3 - x_1} = \frac{4,0 - 1,0}{0,5 - 0,1} = \frac{15}{2};$$

$$[x_0, x_1, x_2, x_3] = \frac{[x_1, x_2, x_3] - [x_0, x_1, x_2]}{x_3 - x_0} = \frac{\frac{15}{2} - \left(-\frac{40}{3}\right)}{0,5 - 0,0} = \frac{125}{3}.$$

Результати розрахунків наведено в таблиці 5.4.

Таблиця 5.4 – Результати розрахунку поділених різниць

$n$	$x_n$	$y_n$	$[x_n, x_{n+1}]$	$[x_n, x_{n+1}, x_{n+2}]$	$[x_n, x_{n+1}, x_{n+2}, x_{n+3}]$
0	0	-0,5			
1	0,1	0,0	5,0	-40/3	125/3
2	0,3	0,2	1,0	15/2	
3	0,5	1,0	4,0		

Інтерполяційна формула Ньютона для нерівновіддалених значень аргументу:

$$\begin{aligned}
 P(x) &= y_0 + [x_0, x_1](x - x_0) + [x_0, x_1, x_2](x - x_0)(x - x_1) + [x_0, x_1, x_2, x_3](x - x_0) \times \\
 &\times (x - x_1)(x - x_2) = -0,5 + 5(x - 0) + \left(-\frac{40}{3}\right)(x - 0)(x - 0,1) + \\
 &+ \frac{125}{3}(x - 0)(x - 0,1)(x - 0,3) = \frac{125}{3}x^3 - 30x^2 + \frac{91}{12}x - 0,5.
 \end{aligned}$$

$$\text{За } x = 0,4; \quad y = P(0,4) = \frac{125}{3}(0,4)^3 - 30 \cdot (0,4)^2 + \frac{91}{12}(0,4) - 0,5 = 0,36.$$

### 5.1.2 Інтерполяція за Лагранжем

Інтерполяція за Лагранжем (*lagrange interpolation*) застосовується в загальному випадку для довільно розташованих вузлів.

Інтерполяційний поліном для методу Лагранжа подано у вигляді:

$$P_n(x) = y_0 b_0(x) + y_1 b_1(x) + \dots + y_n b_n(x),$$

де всі  $b_j(x)$  ( $j=0, 1, 2, \dots, n$ ) – поліноми  $n$ -го степеня, коефіцієнти яких можна знайти з допомогою  $(n+1)$ -го рівняння:

$$P_n(x_i) = y_i,$$

внаслідок чого буде отримано систему рівнянь:

$$\begin{cases} y_0 b_0(x_0) + y_1 b_1(x_0) + \dots + y_n b_n(x_0) = y_0; \\ \dots; \\ y_0 b_0(x_n) + y_1 b_1(x_n) + \dots + y_n b_n(x_n) = y_n. \end{cases}$$

Якщо значення  $b_j(x_i)$  визначати так, що:

$$b_j(x) = \begin{cases} 1, & \text{коли } i = j; \\ 0, & \text{коли } i \neq j, \end{cases}$$

то записана система рівнянь буде визначеною.

Ця умова означає, що будь-який поліном  $b_j(x)$  дорівнює нулю за кожного  $x_i$ , окрім такого, що дорівнює  $x_j$ . Тому в загальному випадку поліном  $b_j(x)$  має такий вигляд:

$$b_j(x) = C_j (x - x_0)(x - x_1) \dots (x - x_{j-1})(x - x_{j+1}) \dots (x - x_n).$$

Якщо  $b_j(x) = 1$ , то коефіцієнти  $C_j$  визначаються із виразу:

$$C_j = 1 / (x_j - x_0) \dots (x_j - x_{j-1})(x_j - x_{j+1}) \dots (x_j - x_n). \quad (5.5)$$

Тоді для полінома, який визначається, отримуємо:

$$P_n(x) = \sum_{j=0}^n y_j \frac{(x - x_0)(x - x_1) \dots (x - x_{j-1})(x - x_{j+1}) \dots (x - x_n)}{(x_j - x_0)(x_j - x_1) \dots (x_j - x_{j-1})(x_j - x_{j+1}) \dots (x_j - x_n)}. \quad (5.6)$$

Вводячи позначення:

$$L_j(x) = (x - x_0)(x - x_1) \dots (x - x_{j-1})(x - x_{j+1}) \dots (x - x_n),$$

можна записати

$$P_n(x) = \sum_{j=0}^n y_j \frac{L_j(x)}{L_j(x_j)}. \quad (5.7)$$

Потрібно відзначити дві головні властивості поліномів Лагранжа:

1.  $\sum_{j=0}^n (L_j(x) / L_j(x_j)) = 1$ .

2. Якщо  $P_n(x)$  лінійно залежить від  $y_j$ , то справедливим є принцип суперпозиції: інтерполяційний поліном суми декількох функцій дорівнює сумі інтерполяційних поліномів складових.

Похибка інтерполяції за Лагранжем оцінюється залишковим членом інтерполяційної формули.

Якщо інтерполяційні вузли відмінні один від одного, а функція  $f(x)$  така, що має неперервну похідну  $(n+1)$  порядку на інтервалі  $[a; b]$ , де розміщені інтерполяційні вузли, можна записати залишковий член інтерполяційної формули  $R_n(x) = f(x) - P_n(x)$  у вигляді:

$$R_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \omega_{n+1}(x),$$

де  $\xi \in [\alpha_1; \alpha_2]$ ,  $\alpha_1 = \min(x, x_0, x_1, \dots, x_n)$ ,  $\alpha_2 = \max(x, x_0, x_1, \dots, x_n)$ .

Тоді

$$|R_n(x)| \leq \frac{M_{n+1}}{(n+1)!} |\omega_{n+1}(x)|,$$

де  $M_{n+1} = \max_{x \in [\alpha_1, \alpha_2]} |f^{(n+1)}(x)|$ .

Схему алгоритму реалізації методу Лагранжа подано на рисунку 5.3.

Розглянемо реалізацію знаходження різниць мовою програмування C++:

```
#include <stdio.h>
#include <math.h>
#include <conio.h>

const int n = 4;

void main()
{
    float x[n] = {1, 2, 3, 4};
    float y[n] = {15, 17, 7, 21};
    float x = 2.5;
    float yx = 0, Pr;
    for (int i=0; i<n; i++)
    {
        Pr = 1;
        for (int j=0; j<n; j++)
            if (i!=j)
                Pr = Pr * ((x-x[j])/(x[i]-x[j]));
        yx = yx + Y[i]*Pr;
    }
    printf("y = %.4f\n", yx);
    getch();
}
```

Результат:  
y = 11.2500.

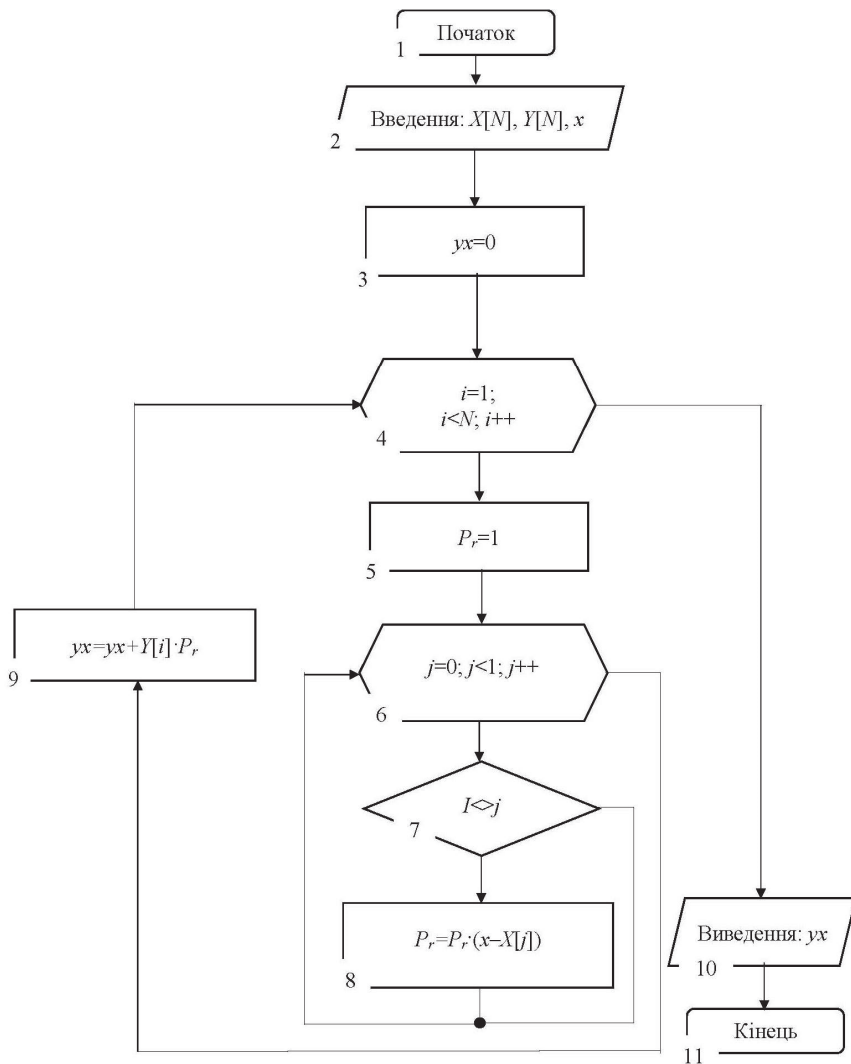


Рисунок 5.3 – Схема алгоритму інтерполяції за методом Лагранжа

**Приклад 5.3.** Знайти значення функції  $y=f(x)$  на основі її табличних значень (табл. 5.5) та знайти значення функції в точці  $x = 0,4$ .

Таблиця 5.5 – Вихідні дані

$i$	0	1	2	3
$x_i$	0,0	0,1	0,3	0,5
$y_i$	-0,5	0,0	0,2	1,0

*Розв'язання:*

Використання формули інтерполяційного полінома Лагранжа:

$$\begin{aligned}
 P_3(x) &= \sum_{i=0}^n y_i \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x-x_j}{x_i-x_j} = \sum_{i=0}^3 y_i \prod_{\substack{j=0 \\ j \neq i}}^3 \frac{x-x_j}{x_i-x_j} = y_0 \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + \\
 &+ y_1 \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} + y_2 \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} + \\
 &+ y_3 \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} = -0,5 \frac{(x-0,1)(x-0,3)(x-0,5)}{(0,0-0,1)(0,0-0,3)(0,0-0,5)} + \\
 &+ 0 \frac{(x-0,0)(x-0,3)(x-0,5)}{(0,1-0,0)(0,1-0,3)(0,1-0,5)} + 0,2 \frac{(x-0,0)(x-0,1)(x-0,5)}{(0,3-0,0)(0,3-0,1)(0,3-0,5)} + \\
 &+ 1,0 \frac{(x-0,0)(x-0,1)(x-0,3)}{(0,5-0,0)(0,5-0,1)(0,5-0,3)} = \frac{125}{3}x^3 - 30x^2 + \frac{91}{12}x - 0,5.
 \end{aligned}$$

$$\text{За } x = 0,4; \quad y \approx L(0,4) = \frac{125}{3}0,4^3 - 30 \cdot 0,4^2 + \frac{91}{12}0,4 - 0,5 = 0,3999.$$

**Приклад 5.4.** Визначити інтерполяційний поліном Лагранжа на основі табличних даних (табл. 5.6), отриманих на основі функції  $y = x + \sin x$ , із похибкою інтерполяції  $\Delta = 0,2 \cdot 10^{-4}$ .

Таблиця 5.6 – Вихідні дані

$i$	0	1	2	3
$x_i$	1,40	1,50	1,70	1,80
$y_i$	2,38545	2,49749	2,69166	2,77385

*Розв'язання:*

Знаходимо інтерполяційний поліном Лагранжа у вигляді:

$$\begin{aligned}
 P_3(x) &= \sum_{i=0}^n y_i \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x-x_j}{x_i-x_j} = \sum_{i=0}^3 y_i \prod_{\substack{j=0 \\ j \neq i}}^3 \frac{x-x_j}{x_i-x_j} = y_0 \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} + \\
 &+ y_1 \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} + y_2 \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} +
 \end{aligned}$$

$$\begin{aligned}
& + y_3 \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} = 2,38548 \frac{(x-1,5)(x-1,7)(x-1,8)}{(1,4-1,5)(1,4-1,7)(1,4-1,8)} + \\
& + 2,49749 \frac{(x-1,4)(x-1,7)(x-1,8)}{(1,5-1,4)(1,5-1,7)(1,5-1,8)} + \\
& + 2,69166 \frac{(x-1,4)(x-1,5)(x-1,8)}{(1,7-1,4)(1,7-1,5)(1,7-1,8)} + \\
& + 2,77385 \frac{(x-1,4)(x-1,5)(x-1,7)}{(1,8-1,4)(1,8-1,5)(1,8-1,7)} = \\
& = -198,79(x-1,5)(x-1,7)(x-1,8) + 416,2483(x-1,4)(x-1,7)(x-1,8) - \\
& - 448,61(x-1,4)(x-1,5)(x-1,8) + 231,1542(x-1,4)(x-1,5)(x-1,7).
\end{aligned}$$

Оцінка похибки інтерполяції за Лагранжем:

$$|R_n(x)| = |R_3(x)| \leq \frac{M_{n+1}}{(n+1)!} |\omega_{n+1}(x)| = \frac{M_4}{4!} |\omega_4(x)| = \frac{0,98545}{4!} \cdot 0,0004 = 1,64 \cdot 10^{-5},$$

де  $\omega_4(1,6) = (x-x_0)(x-x_1)(x-x_2)(x-x_3) = (1,6-1,4)(1,6-1,5)(1,6-1,7) \times$   
 $\times (1,6-1,8) = 0,2 \cdot 0,1 \cdot (-0,1) \cdot (-0,2) = 0,0004$ ;  $\alpha_1 = \min(x, x_0, x_1, x_2, x_3) = \min(1,6;$   
 $1,4; 1,5; 1,7; 1,8) = 1,4$ ;  $\alpha_2 = \max(x, x_0, x_1, x_2, x_3) = \max(1,6; 1,4; 1,5; 1,7; 1,8) = 1,8$ ;

$$M_{n+1} = M_4 = \max_{x \in [\alpha_1, \alpha_2]} |f^{(4)}(x)| = \max_{x \in [1,4; 1,8]} |\sin(x)| = 0,98545.$$

Оскільки  $|R_n(x)| = 1,64 \cdot 10^{-5} < \Delta = 0,2 \cdot 10^{-4}$ , що відповідає умові задачі.

Для визначення значення інтерполяційного поліному Лагранжа в точці  $x = 1,6$  використовується формула  $P_3(x) = \sum_{i=0}^3 y_i \prod_{\substack{j=0 \\ j \neq i}}^3 \frac{x-x_j}{x_i-x_j}$ , на основі якої крок за кроком виконуються обчислення, результати яких зводяться у таблицю 5.7.

Таблиця 5.7 – Результати обчислень інтерполяційного многочлена

$i$	$x-x_i$	$x_i-x_j, i \neq j$			$\prod_{i \neq j} (x_i-x_j)$	$y_i \prod_{\substack{j=0 \\ j \neq i}}^n \left( \frac{x-x_j}{x_i-x_j} \right)$	$\sum_{i=0}^n y_i \prod_{\substack{j=0 \\ j \neq i}}^n \left( \frac{x-x_j}{x_i-x_j} \right)$
0	0,2	-0,1	-0,3	-0,4	-0,012	-0,3978500	-0,3978500
1	0,1	0,1	-0,2	-0,3	0,006	1,6649930	1,2671430
2	-0,1	0,3	0,2	-0,1	-0,006	1,7944400	3,0615830
3	-0,2	0,4	0,3	0,1	0,012	-0,4623083	2,5992747

На основі результатів обчислення (див. табл. 5.6) значення інтерполяційного поліному Лагранжа  $P_3(1,6) = 2,5992747$ . Для порівняння значення функції  $y = x + \sin x$  в точці  $x = 1,6$  із сімома точними десятковими знаками:  $y(1,6) = 2,5995736$ .



$$\text{де } \phi_j(x_i, y_i) = \begin{cases} 1, & i = j; \\ 0, & i \neq j; \end{cases} \quad (j = \overline{0, \dots, n}).$$

Оскільки шуканий поліном дорівнює нулю в  $n$  точках  $(x_0, y_0), (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$ , то виключаючи значення за умови  $i = j$ , матимемо поліном  $\phi_j(x_i, y_i)$  у такому вигляді:

$$\begin{aligned} \phi_j(x, y) = & C_j(x - x_0)(x - x_1) \dots (x - x_{j-1})(x - x_{j+1}) \dots (x - x_n) \times \\ & \times (y - y_0)(y - y_1) \dots (y - y_{j-1})(y - y_{j+1}) \dots (y - y_n), \end{aligned} \quad (5.11)$$

де  $C_j = \text{const}$ .

Вважаючи, що  $x=x_i$  та  $y=y_i$ , а також враховуючи, що  $\phi_j(x_i, y_i) = 1$ , отримуємо:

$$\begin{aligned} C_j = & \frac{1}{(x_j - x_0)(x_j - x_1) \dots (x_j - x_{j-1})(x_j - x_{j+1}) \dots (x_j - x_n)} \times \\ & \times \frac{1}{(y_j - y_0)(y_j - y_1) \dots (y_j - y_{j-1})(y_j - y_{j+1}) \dots (y_j - y_n)}. \end{aligned}$$

Загальний вигляд інтерполяційної формули Лагранжа:

$$\begin{aligned} P_n(x, y) = & \sum_{i=0}^n z_j \frac{(x - x_0)(x - x_1) \dots (x - x_{j-1})(x - x_{j+1}) \dots (x - x_n)}{(x_j - x_0)(x_j - x_1) \dots (x_j - x_{j-1})(x_j - x_{j+1}) \dots (x_j - x_n)} \times \\ & \times \frac{(y - y_0)(y - y_1) \dots (y - y_{j-1})(y - y_{j+1}) \dots (y - y_n)}{(y_j - y_0)(y_j - y_1) \dots (y_j - y_{j-1})(y_j - y_{j+1}) \dots (y_j - y_n)}. \end{aligned} \quad (5.12)$$

Припустимо, що  $(x_0, y_0), (x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$  – це множина  $n+1$  різних чисел, які набувають значення в просторі  $[x_0 \leq x \leq x_n; y_0 \leq y \leq y_n]$ . Існує єдиний поліном  $P_n(x, y)$  степеня не більше ніж  $n^2$  такий, що має властивість:  $f(x_i, y_i) = P_n(x_i, y_i)$  для  $i = 0, 1, \dots, n$ .

Формула Лагранжа для цього полінома має вигляд ( $i = 0, 1, \dots, n$ ):

$$\begin{aligned} P_n(x, y) = & \sum_{i=0}^n z_j \frac{(x - x_0)(x - x_1) \dots (x - x_{j-1})(x - x_{j+1}) \dots (x - x_n)}{(x_j - x_0)(x_j - x_1) \dots (x_j - x_{j-1})(x_j - x_{j+1}) \dots (x_j - x_n)} \times \\ & \times \frac{(y - y_0)(y - y_1) \dots (y - y_{j-1})(y - y_{j+1}) \dots (y - y_n)}{(y_j - y_0)(y_j - y_1) \dots (y_j - y_{j-1})(y_j - y_{j+1}) \dots (y_j - y_n)}. \end{aligned}$$

Причому, якщо в чисельнику та знаменнику деякий співмножник  $(x-x_k)$  або  $(y-y_k)$  зустрічається декілька разів, то усі, крім одного, виключаються з обчислень.

Для випадку, коли  $x_j = x_k$  та  $y_j \neq y_k$  маємо:



$$P_n(x, y) = \sum_{j=0}^n z_j \frac{(x-x_0)(x-x_1)\dots(x-x_{j-1})(x-x_{j+1})\dots(x-x_{k-1})}{(x_j-x_0)(x_j-x_1)\dots(x_j-x_{j-1})(x_j-x_{j+1})\dots(x_j-x_{k-1})} \times \frac{(x-x_{k+1})\dots(x-x_n)(y-y_0)(y-y_1)\dots(y-y_{j-1})(y-y_{j+1})\dots(y-y_n)}{(x_i-x_{k+1})\dots(x_j-x_n)(y_j-y_0)(y_j-y_1)\dots(y_j-y_{j-1})(y_j-y_{j+1})\dots(y_j-y_n)}. \quad (5.13)$$

За умов  $x_j \neq x_k$  та  $y_j = y_k$ :

$$P_n(x, y) = \sum_{j=0}^n z_j \frac{(x-x_0)(x-x_1)\dots(x-x_{j-1})(x-x_{j+1})\dots(x-x_n)}{(x_j-x_0)(x_j-x_1)\dots(x_j-x_{j-1})(x_j-x_{j+1})\dots(x_j-x_n)} \times \frac{(y-y_0)(y-y_1)\dots(y-y_{j-1})(y-y_{j+1})\dots(y-y_{k-1})(y-y_{k+1})\dots(y-y_n)}{(y_j-y_0)(y_j-y_1)\dots(y_j-y_{j-1})(y_j-y_{j+1})\dots(y_j-y_{k-1})(y_j-y_{k+1})\dots(y_j-y_n)}.$$

Введемо такі позначення:

$$L_j^{(n)}(x, y) = (x-x_0)(x-x_1)\dots(x-x_{j-1})(x-x_{j+1})\dots(x-x_n) \times (y-y_0)(y-y_1)\dots(y-y_{j-1})(y-y_{j+1})\dots(y-y_n).$$

Тоді інтерполяційна формула набуде вигляду:

$$P_n(x, y) = \sum_{j=0}^n z_j \frac{L_j(x, y)}{L_j(x_j, y_j)}. \quad (5.14)$$

В іншій формі запису записі формула (5.14) набуде такого вигляду:

$$L_j^{(n)}(x, y) = \frac{\Pi_{x+1}(x) \cdot \Pi_{y+1}(y)}{(x-x_j)(y-y_j) \cdot \Pi'_{n+1}(x) \cdot \Pi'_{n+1}(y)},$$

де  $\Pi_{x+1}(x) = (x-x_0) \dots (x-x_n)$ ,  $\Pi_{y+1}(y) = (y-y_0) \dots (y-y_n)$ .

Розглянемо приклад використання методу.

Нехай задано дві точки в просторі  $z_0(x_0, y_0)$  та  $z_1(x_1, y_1)$ , тоді вираз для лінійної інтерполяції:

$$P_n(x, y) = z_0 \frac{(x-x_1)(y-y_1)}{(y_0-y_1)(x_0-x_1)} + z_1 \frac{(x-x_0)(y-y_0)}{(y_1-y_0)(x_1-x_0)} = \frac{z_0(x-x_1)(y-y_1) + z_1(x-x_0)(y-y_0)}{(x_0-x_1)(y_0-y_1)}. \quad (5.15)$$

За умови, що  $x_0 = x_1$ ,  $y_0 \neq y_1$ , отримаємо:

$$P_n(x, y) = z_0 \frac{(y-y_1)}{(y_0-y_1)} + z_1 \frac{(y-y_0)}{(y_1-y_0)} = \frac{z_0(y-y_1) + z_1(y-y_0)}{(y_0-y_1)}.$$

Також, за умови, що  $x_0 \neq x_1$ ,  $y_0 = y_1$ :

$$P_n(x, y) = z_0 \frac{(x-x_1)}{(x_0-x_1)} + z_1 \frac{(x-x_0)}{(x_1-x_0)} = \frac{z_0(x-x_1) + z_1(x-x_0)}{(x_0-x_1)}. \quad (5.16)$$

Розглянемо послідовність дій за використання методу Лагранжа для інтерполяції функції двох змінних, а саме  $z_i = f(x_i, y_i)$  ( $i = 1, 2, \dots, n$ ).

Для визначення значення функції у точці  $(x, y)$  за методом Лагранжа необхідно провести такі обчислення:

1. Для кожного  $i = 1, 2, \dots, n$ :
  - а) обчислюється  $L_i(x, y)$ ;
  - б) обчислюється  $L_i(x_i, y_j)$ ;
  - в) визначається співвідношення  $L_i(x, y)/L_i(x_i, y_j)$ ;
  - г) отримане значення співвідношення необхідно помножити на  $z_i$ .
2. Усі значення, отримані в пункті 1, г) необхідно додати між собою. Отриманий результат буде значенням інтерполяційного многочлена Лагранжа у точці  $(x, y)$ .

Розглянемо інтерполяцію просторових кривих, які задано в параметричній формі:

$$\begin{cases} y = y(x); \\ z = z(x). \end{cases} \quad (5.17)$$

Відповідно для кожної із функцій  $y(x)$  та  $z(x)$  виконується звичайна одновимірна інтерполяція Лагранжа, і за двома отриманими інтерполювальними поліномами  $P_y(x)$  та  $P_z(x)$  будується крива, яка складається з точок  $\{x_i, P_y(x_i), P_z(x_i)\}$ .

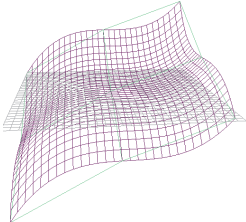
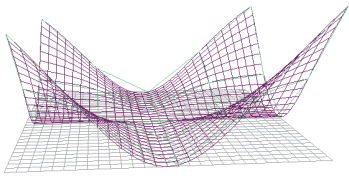
За використання інтерполяції Лагранжа, для отримання поліномів  $P_y(x_i)$  та  $P_z(x_i)$ , необхідно використовувати відповідні формули:

$$P_{y_i}(x) = \sum_{j=0}^n y_j \frac{(x - x_0)(x - x_1) \dots (x - x_{j-1})(x - x_{j+1}) \dots (x - x_n)}{(x_j - x_0)(x_j - x_1) \dots (x_j - x_{j-1})(x_j - x_{j+1}) \dots (x_j - x_n)},$$

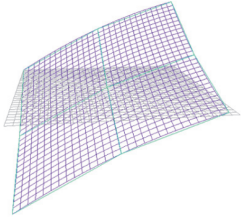
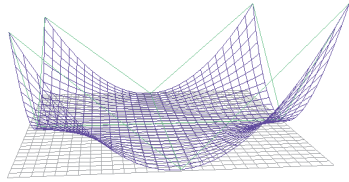
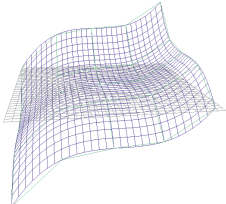
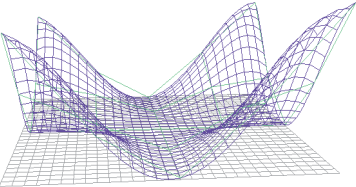
$$P_{z_i}(x) = \sum_{j=0}^n z_j \frac{(x - x_0)(x - x_1) \dots (x - x_{j-1})(x - x_{j+1}) \dots (x - x_n)}{(x_j - x_0)(x_j - x_1) \dots (x_j - x_{j-1})(x_j - x_{j+1}) \dots (x_j - x_n)}.$$

Приклад інтерполяції функцій двох змінних на основі інтерполяційних вузлів, отриманих за допомогою типових функцій, наведено в таблиці 5.8.

Таблиця 5.8 – Приклади інтерполяції функцій двох змінних

Функція	$f(x, y) = x^3 - y^2$	$f(x, y) = \frac{x^2 y^2}{x^2 + y^2 + 1}$
Реальна функція		

Продовження таблиці 5.8

Інтерполяція за методом Лагранжа, (9 вузлів)		
Інтерполяція за методом Лагранжа, (25 вузлів)		

**Приклад 5.5.** Знайти значення інтерполяційної функції в точці  $x=4, y=2$ , яка визначена на основі інтерполяційних вузлів (табл. 5.9).

Таблиця 5.9 – Вихідні дані

$i$	0	1	2	3
$x_i$	2	3	5	6
$y_i$	5	10	-10	-4
$z_i$	-17	-73	25	200

*Розв'язання:*

Користуючись формулою (5.13), отримуємо:

$$\begin{aligned}
 P_3(x, y) &= \sum_{j=0}^3 z_j \frac{L_j(x, y)}{L_j(x_j, y_j)} = \frac{z_0(x-x_1)(x-x_2)(x-x_3)(y-y_1)(y-y_2)(y-y_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)(y_0-y_1)(y_0-y_2)(y_0-y_3)} + \\
 &+ z_1 \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} \cdot \frac{(y-y_0)(y-y_2)(y-y_3)}{(y_1-y_0)(y_1-y_2)(y_1-y_3)} + \\
 &+ z_2 \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} \cdot \frac{(y-y_0)(y-y_1)(y-y_3)}{(y_2-y_0)(y_2-y_1)(y_2-y_3)} + \\
 &+ z_3 \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} \cdot \frac{(y-y_0)(y-y_1)(y-y_2)}{(y_3-y_0)(y_3-y_1)(y_3-y_2)} = \\
 &= -17 \cdot \frac{(x-3)(x-5)(x-6)}{(2-3)(2-5)(2-6)} \cdot \frac{(y-10)(y+10)(y+4)}{(5-10)(5+10)(5+4)} +
 \end{aligned}$$

$$\begin{aligned}
&+(-73) \cdot \frac{(x-2)(x-5)(x-6)}{(3-2)(3-5)(3-6)} \cdot \frac{(y-5)(y+10)(y+4)}{(10-5)(10+10)(10+4)} + \\
&+25 \cdot \frac{(x-2)(x-3)(x-6)}{(5-2)(5-3)(5-6)} \cdot \frac{(y-5)(y-10)(y+4)}{(-10-5)(-10-10)(-10+4)} + \\
&+200 \cdot \frac{(x-2)(x-3)(x-5)}{(6-2)(6-3)(6-5)} \cdot \frac{(y-5)(y-10)(y+10)}{(-4-5)(-4-10)(-4+10)} = \\
&=(x-5)(x-6)(y+10)(y+4) \left( \frac{-17}{8100}(x-3)(y-10) + \frac{73}{8400}(x-2)(y-5) \right) + \\
&+(x-2)(x-3)(y-5)(y-10) \left( \frac{-1}{432}(x-6)(y+4) + \frac{25}{1134}(x-5)(y+10) \right).
\end{aligned}$$

Значення інтерполяційної функції в точці  $x = 4$  і  $y = 2$ :

$$\begin{aligned}
z = P_3(4, 2) &= (4-5)(4-6)(2+10)(2+4) \left( \frac{-17}{8100}(4-3)(2-10) + \right. \\
&\quad \left. + \frac{73}{8400}(4-2)(2-5) \right) + \\
&+(4-2)(4-3)(2-5)(2-10) \left( \frac{-1}{432}(4-6)(2+4) + \frac{25}{1134}(4-5)(2+10) \right) = -4,105.
\end{aligned}$$

Програма мовою програмування C/C++ має вигляд:

```

#include <stdio.h>
#include <math.h>
#include <conio.h>
const int n = 4;
void main()
{
    float X[n] = {2, 3, 5, 6};
    float Y[n] = {5, 10, -10, -4};
    float Z[n] = {-17, -73, 25, 200};
    float x = 4, y = 2;
    float zxy = 0, Pr;
    for (int i=0; i<n; i++)
    {
        Pr = 1;
        for (int j=0; j<n; j++)
            if (i!=j)
                Pr = Pr * ((x-X[j])*(y-Y[j]))/(((X[i]-X[j])*(Y[i]-Y[j])));
        zxy = zxy + Z[i]*Pr;
    }
    printf("y = %.3f\n", zxy);
    getch();
}

```

Результат:  
Z = -4.105

### 5.1.4 Сплайн-інтерполяція

Інтерполяція многочленом Лагранжа або Ньютона на всьому відрізку з використанням великого числа вузлів інтерполяції часто призводить до неточного наближення, що пояснюється суттєвим накопиченням похибок у процесі обчислень. Крім того, через розбіжність процесу інтерполяції збільшення числа вузлів не завжди приводить до підвищення точності. Для того, щоб уникнути великих похибок, увесь відрізок розбивають на частинні відрізки і на кожному із частинних відрізків приблизно замінюється функція  $f(x)$  многочленом невисокого степеня (так звана кусково-поліноміальна інтерполяція).

Одним із способів інтерполяції на всьому відрізку є інтерполяція за допомогою сплайн-функцій. Сплайн-функцією або сплайном називають кусково-поліноміальну функцію, що визначена на відрізку і має на цьому відрізку деяке число неперервних похідних.

Слово «сплайн» (англійське «spline») означає «гнучку лінійку», що використовується для проведення гладких кривих через задані точки площини. Основною перевагою сплайнів є можливість локально змінювати форму кривої на виділеному проміжку значень.

#### Класичний кубічний сплайн

Розглянемо найбільш відомий і поширений інтерполяційний сплайн третього порядку. У машинобудівному кресленні ці сплайни широко використовуються у вигляді лекал (гнучкі лінійки), які деформуються так, щоб з їх допомогою можна було провести криву через задані точки  $(x_i, y_i)$ . Можна показати (використовуючи теорію пружно-поперечного згину бруса за малих деформацій), що сплайн – це група об'єднаних кубічних многочленів, у місцях з'єднання яких перша та друга похідні відповідних функцій многочленів неперервні. Такі функції називаються кубічними сплайнами (рис. 5.4), для побудови яких необхідно задати коефіцієнти, що однозначно визначають поліном у проміжку між двома точками.

Для математичного опису кубічних сплайнів розглянемо на відрізку  $[a; b]$  дійсної осі  $OX$  інтерполяційну сітку  $a = x_0 < x_1 < \dots < x_n = b$ , у вузлах якої визначено значення вузлів  $y_i = f(x_i)$  ( $i = 0, 1, \dots, n$ ) функції  $f(x)$ . Необхідно на відрізку  $[a; b]$  побудувати неперервну функцію – сплайн  $S(x)$ , яка задовольняє такі вимоги:

1. На кожному відрізку  $[x_{i-1}; x_i]$  сплайн  $S(x)$  є многочленом  $S_i(x)$  не вище третього степеня (рис. 5.4):

$$S_i(x) = k_{1i} + k_{2i}x + k_{3i}x^2 + k_{4i}x^3, \quad (5.18)$$

де  $k_{ij}$  – постійні коефіцієнти, які необхідно визначити.

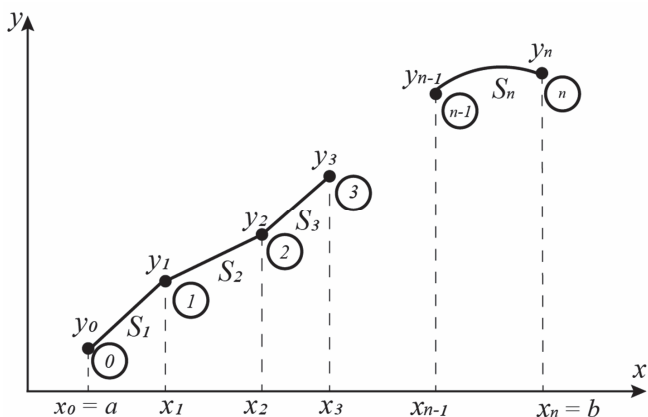


Рисунок 5.4 – Схема сплайн-інтерполяції

2. У вузлах  $x_i$  сплайн  $S_i(x)$  набуває заданих значень  $y_i = f(x_i)$  ( $i = 0, 1, \dots, n$ ) тобто:

$$\begin{cases} S_i(x_i) = y_i & (i = 1, 2, \dots, n); \\ S_{i+1}(x_i) = y_i & (i = 0, 1, \dots, n-1). \end{cases} \quad (5.19)$$

Умова (5.19) необхідна для проходження сплайнів через вузли заданої інтерполювальної сітки  $a = x_0 < x_1 < \dots < x_n = b$ ,  $y_i = f(x_i)$  ( $i = 0, 1, \dots, n$ ). Ця умова (5.19) утворює  $2n$  рівнянь.

3. У внутрішніх вузлах  $x_i$  ( $i = 1, 2, \dots, n-1$ ) сплайн має неперервну першу і другу похідні, а саме:

$$\begin{cases} S'_{i+1}(x_i) = S'_i(x_i) & (i = \overline{1, n-1}); \\ S''_{i+1}(x_i) = S''_i(x_i) & (i = \overline{1, n-1}). \end{cases}$$

Тобто будуть отримані такі вирази:

$$\begin{cases} S_i(x) = k_{4i}x^3 + k_{3i}x^2 + k_{2i}x + k_{1i} & (i = \overline{1, n}); \\ S'_i(x) = 3k_{4i}x^2 + 2k_{3i}x + c_i & (i = \overline{1, n-1}); \\ S''_i(x) = 6k_{4i}x + 2k_{3i} & (i = \overline{1, n-1}). \end{cases}$$

У точках спряження сплайнів їх перші та другі похідні мають бути однаковими між собою. Кількість таких умов має бути  $2n-2$ . Для знаходження сплайна необхідно визначити коефіцієнти  $k_{ij}$  многочленів  $S_i(x)$  ( $i = 1, 2, \dots, n-1$ ) із  $4n$  невідомими, які задовольняють систему із  $4n-2$  рівнянь.

Для отримання розв'язку системи рівнянь необхідно ще два додаткових рівняння. Їх отримують, визначивши значення кривизни графіка сплайна на

кінцях загальної інтерполювальної кривої, а саме:  $S''(x_0) = \sigma_1$ ,  $S''(x_n) = \sigma_2$ .

Алгоритм розв'язання задачі інтерполяції за допомогою полінома третього порядку подано на рисунку 5.5.

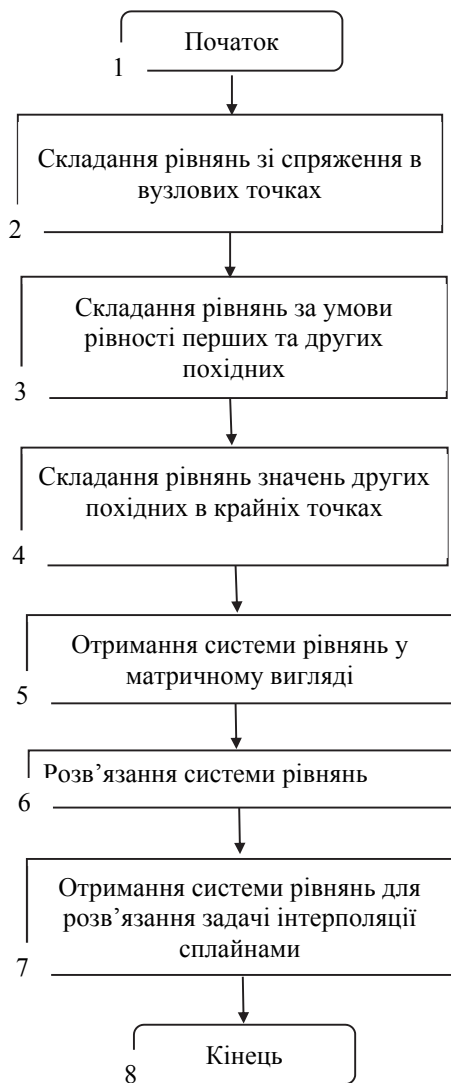


Рисунок 5.5 – Схема алгоритму розв'язання задачі інтерполяції за допомогою полінома третього порядку

Якщо  $\sigma_1 = \sigma_2 = 0$ , то такий сплайн називають природним. Якщо є додаткові відомості про поведінку функції на кінцях інтервалу інтерполяції, тоді записуються додаткові крайові умови. Таким чином, отриманий кубічний сплайн, що «склесний» із кубічних парабол, проходить через задані точки, є кусково-гладкою і неперервною функцією.

Для побудови функції кривої (5.18) необхідно визначити чотири коефіцієнти. Вираз (5.18) може бути записаний у формі, яку запропонував Чарльз Ерміт, що дозволяє зменшити кількість обчислювальних операцій. Для цього окремі кубічні рівняння можуть бути записані у вигляді:

$$S_i(x) = ty_i + \bar{t}y_{i-1} + \Delta x_i \left[ (k_{i-1} - d_i)\bar{t}^2 - (k_i - d_i)t^2\bar{t} \right] \quad (i = \overline{1, n}), \quad (5.20)$$

де  $\Delta x_i = x_i - x_{i-1}$ ,  $t = \frac{x - x_{i-1}}{\Delta x_i}$ ,  $\bar{t} = 1 - t$ ,  $\Delta y_i = y_i - y_{i-1}$ ,  $\frac{\Delta y_i}{\Delta x_i} = d_i$ ,  $\Delta x_i$ ,  $\Delta y_i$  – довжина інтервалу;  $t$  і  $\bar{t}$  – допоміжні змінні;  $x$  – проміжна точка на відрізку  $[x_{i-1}; x_i]$ .

Кожне із рівнянь  $S_i(x)$  (5.20) містить тільки два постійних невідомих коефіцієнти. Після того як перше рівняння  $S_i(x)$  записано, з кожним наступним рівнянням додається тільки один новий невідомий коефіцієнт. Тоді за  $x=x_{i-1}$ ,  $t=0$ ,  $\bar{t}=1$ , а за  $x=x_i$ ,  $t=1$ ,  $\bar{t}=0$ .

Відповідно, усі умови, окрім умов для других похідних, задовольняються. Другі похідні для внутрішніх точок виражаються співвідношеннями:

$$k_{i-1}\Delta x_{i+1} + 2k_i(\Delta x_i + \Delta x_{i+1}) + k_{i+1}\Delta x_i = 3(d_i\Delta x_{x+1} + d_{i+1}\Delta x_i),$$

а для двох зовнішніх –

$$2k_0 + k_1 = 3d_1 \quad \text{і} \quad k_{m-1} + 2k_m = 3d_m.$$

Таким чином, система рівнянь, яку необхідно розв'язати, є лінійною, а її матриця – тридіагональною (див. розд. 1):

$$\begin{pmatrix} 2 & 1 & 0 & 0 & 0 \\ \Delta x_2 & 2(\Delta x_1 + \Delta x_2) & \Delta x_1 & 0 & 0 \\ 0 & \Delta x_3 & 2(\Delta x_2 + \Delta x_3) & \Delta x_2 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 1 & 2 \end{pmatrix} \begin{pmatrix} k_0 \\ k_1 \\ k_2 \\ \dots \\ k_n \end{pmatrix} = 3 \begin{pmatrix} d_1 \\ d_1\Delta x_2 + d_2\Delta x_1 \\ d_2\Delta x_3 + d_3\Delta x_2 \\ \dots \\ d_{m-1}\Delta x_m + d_m\Delta x_{m-1} \\ d_m \end{pmatrix}.$$

У багатьох випадках метод сплайн-інтерполяції є найбільш зручним, оскільки дозволяє отримати аналітичну кусково-поліноміальну функцію. Існують сплайни вищих порядків. Застосування цього методу можливо також в інших сферах обчислювальної математики, наприклад, у чисельному інтегруванні або для розв'язання диференціальних рівнянь.



**Приклад 5.6.** Виконати сплайн-інтерполяцію за допомогою полінома третього порядку для функції  $y=f(x)$ , яка задана таблично (табл. 5.10), а також визначити наближене значення отриманої інтерполювальної функції в точці  $x=2,5$ .

Таблиця 5.10 – Вихідні дані

$i$	0	1	2	3
$x_i$	1	2	3	4
$y_i$	15	17	7	21

*Розв'язання:*

Відповідно до трьох проміжків значень аргументу  $x_i$  (див. табл. 5.10) на основі умови (5.18) буде отримано таку систему з  $m=3$  рівнянь:

$$y = \begin{cases} A_1x^3 + B_1x^2 + C_1x + D_1, & x \in [1; 2]; \\ A_2x^3 + B_2x^2 + C_2x + D_2, & x \in [2; 3]; \\ A_3x^3 + B_3x^2 + C_3x + D_3, & x \in [3; 4]. \end{cases} \quad (5.21)$$

Для того, щоб визначити невідомі коефіцієнти  $A_1, B_1, C_1, D_1, A_2, B_2, C_2, D_2, A_3, B_3, C_3, D_3$ , необхідно скласти систему кількістю  $m \cdot 4 = 12$  рівнянь і з 12 невідомими.

Перші  $2m$  рівнянь складаються відповідно до того, що сплайни мають спрягатися в заданих вузлових точках:

- 1)  $A_1x_0^3 + B_1x_0^2 + C_1x_0 + D_1 = y_0$ ;
- 2)  $A_1x_1^3 + B_1x_1^2 + C_1x_1 + D_1 = y_1$ ;
- 3)  $A_2x_1^3 + B_2x_1^2 + C_2x_1 + D_2 = y_1$ ;
- 4)  $A_2x_2^3 + B_2x_2^2 + C_2x_2 + D_2 = y_2$ ;
- 5)  $A_3x_2^3 + B_3x_2^2 + C_3x_2 + D_3 = y_2$ ;
- 6)  $A_3x_3^3 + B_3x_3^2 + C_3x_3 + D_3 = y_3$ .

Наступні  $2m-2$  рівняння складаються за умови рівності перших та других похідних у місцях спряження сплайнів:

- 7)  $3A_1x_1^2 + 2B_1x_1 + C_1 = 3A_2x_1^2 + 2B_2x_1 + C_2$ ;
- 8)  $3A_2x_2^2 + 2B_2x_2 + C_2 = 3A_3x_2^2 + 2B_3x_2 + C_3$ ;
- 9)  $6A_1x_1 + 2B_1 = 6A_2x_1 + 2B_2$ ;
- 10)  $6A_2x_2 + 2B_2 = 6A_3x_2 + 2B_3$ .

Для двох останніх рівнянь використовується додаткова умова, що значення другої похідної в крайніх точках має дорівнювати нулю, а саме:

- 11)  $6A_1x_0 + 2B_1 = 0$ ;
- 12)  $6A_3x_3 + 2B_3 = 0$ .

Підставляючи вихідні дані (див. табл. 5.10), отримаємо систему з 12-ти рівнянь із 12 невідомими:

$$\left\{ \begin{array}{l} A_1 + B_1 + C_1 + D_1 = 15; \\ 8A_1 + 4B_1 + 2C_1 + D_1 = 17; \\ 8A_2 + 4B_2 + 2C_2 + D_2 = 17; \\ 27A_2 + 9B_2 + 3C_2 + D_2 = 7; \\ 27A_3 + 9B_3 + 3C_3 + D_3 = 7; \\ 64A_3 + 16B_3 + 4C_3 + D_3 = 21; \\ 12A_1 + 4B_1 + C_1 = 12A_2 + 4B_2 + C_2; \\ 27A_2 + 6B_2 + C_2 = 27A_3 + 6B_3 + C_3; \\ 12A_1 + 2B_1 = 12A_2 + 2B_2; \\ 18A_2 + 2B_2 = 18A_3 + 2B_3; \\ 6A_1 + 2B_1 = 0; \\ 24A_3 + 2B_3 = 0. \end{array} \right. \quad (5.22)$$

Подаючи систему рівнянь (5.22) у матричному вигляді, отримаємо:

$$\begin{array}{cccccccccccc|c} & A_1 & B_1 & C_1 & D_1 & A_2 & B_2 & C_2 & D_2 & A_3 & B_3 & C_3 & D_3 & & \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 15 & \\ 2 & 8 & 4 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 17 & \\ 3 & 0 & 0 & 0 & 0 & 8 & 4 & 2 & 1 & 0 & 0 & 0 & 0 & 17 & \\ 4 & 0 & 0 & 0 & 0 & 27 & 9 & 3 & 1 & 0 & 0 & 0 & 0 & 7 & \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 27 & 9 & 3 & 1 & 7 & \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 64 & 16 & 4 & 1 & 21 & \\ 7 & 12 & 4 & 1 & 0 & -12 & -4 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 8 & 0 & 0 & 0 & 0 & 27 & 6 & 1 & 0 & -27 & -6 & -1 & 0 & 0 & \\ 9 & 12 & 2 & 0 & 0 & -12 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 10 & 0 & 0 & 0 & 0 & 18 & 2 & 0 & 0 & -18 & -2 & 0 & 0 & 0 & \\ 11 & 6 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\ 12 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 24 & 2 & 0 & 0 & 0 & \end{array} \quad (5.23)$$

Після розв'язання системи рівнянь (5.22), наприклад, за допомогою методу Гаусса, підставляємо знайдені коефіцієнти в (5.21), що дозволяє отримати розв'язок поставленої задачі.

Під час розв'язання задачі методом Гаусса попередньо необхідно перетворити систему (5.23) до вигляду, щоб у головній діагоналі не було

нульових елементів (необхідно поміняти відповідні рівняння місцями), а саме:

$$\begin{array}{cccccccccccc|c}
 & A_1 & B_1 & C_1 & D_1 & A_2 & B_2 & C_2 & D_2 & A_3 & B_3 & C_3 & D_3 & \\
 11 & 6 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 15 \\
 7 & 12 & 4 & 1 & 0 & -12 & -4 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 2 & 8 & 4 & 2 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 17 \\
 8 & 0 & 0 & 0 & 0 & 27 & 6 & 1 & 0 & -27 & -6 & -1 & 0 & 0 & 0 \\
 9 & 12 & 2 & 0 & 0 & -12 & -2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 3 & 0 & 0 & 0 & 0 & 8 & 4 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 17 \\
 4 & 0 & 0 & 0 & 0 & 27 & 9 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 7 \\
 10 & 0 & 0 & 0 & 0 & 18 & 2 & 0 & 0 & -18 & -2 & 0 & 0 & 0 & 0 \\
 12 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 24 & 2 & 0 & 0 & 0 & 0 \\
 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 27 & 9 & 3 & 1 & 0 & 7 \\
 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 64 & 16 & 4 & 1 & 0 & 21
 \end{array} \quad (5.24)$$

Розв'яжемо систему рівнянь за допомогою методу Гаусса. Програма мовою C/C++ має вигляд:

```

#include <iostream>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    int n = 12;
    float a[12][12] = {6, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                      1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                      12, 4, 1, 0, -12, -4, -1, 0, 0, 0, 0, 0, 0, 0,
                      8, 4, 2, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                      0, 0, 0, 0, 27, 6, 1, 0, -27, -6, -1, 0, 0, 0,
                      12, 2, 0, 0, -12, -2, 0, 0, 0, 0, 0, 0, 0, 0,
                      0, 0, 0, 0, 8, 4, 2, 1, 0, 0, 0, 0, 0, 0,
                      0, 0, 0, 0, 27, 9, 3, 1, 0, 0, 0, 0, 0, 0,
                      0, 0, 0, 0, 18, 2, 0, 0, -18, -2, 0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 24, 2, 0, 0, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 27, 9, 3, 1, 0, 0,
                      0, 0, 0, 0, 0, 0, 0, 0, 64, 16, 4, 1, 0, 0};
    float b[12] = {0, 15, 0, 17, 0, 0, 17, 7, 0, 0, 7, 21};
    float d;

    for (int i=0; i<n-1; i++)
        for (int j=i+1; j < n; j++)
            {
                d = a[j][i] / a [i][i];
                for (int m=0; m < n; m++)
                    a[j][m] = a[i][m] * d - a[j][m];
                b[j] = b[i]*d - b[j];
            }
}

```

```

float x[12];

for (int i=n-1; i>=0; i--)
{
    float D = 0;
    for (int j=n-1; j>i; j--)
        D = D + x[j]*a[i][j];
    x[i] = (b[i]-D) / a[i][i];
}
for (int i=0; i<n; i++)
    cout << "x[" << i << "]=" << x[i] << endl;

system("PAUSE");
return 0; }

```

В процесі розв'язання системи рівнянь (5.24), отримано такі значення коефіцієнтів:  $A_1 = -4,8$ ;  $B_1=14,4$ ;  $C_1 = -7,6$ ;  $D_1 = 13,0$ ;  $A_2 = 12,0$ ;  $B_2 = -86,4$ ;  $C_2 = 194,0$ ;  $D_2 = -121,4$ ;  $A_3 = -7,2$ ;  $B_3 = 86,4$ ;  $C_3 = -324,4$ ;  $D_3 = 397,0$ .

Відповідно, система рівнянь (5.21) матиме вигляд:

$$y = \begin{cases} -4,8x^3 + 14,4x^2 - 7,6x + 13, & x \in [1; 2]; \\ 12x^3 - 86,4x^2 + 194x - 121,4, & x \in [2; 3]; \\ -7,2x^3 + 86,4x^2 - 324,4x + 397, & x \in [3; 4]. \end{cases}$$

Тоді  $y(2,5) = 12 \cdot 2,5^3 - 86,4 \cdot 2,5^2 + 194 \cdot 2,5 - 121,4 = 11,1$ .

## 5.2 Апроксимація даних

Апроксимація (*approximation*) – це наближений опис однією функцією (апроксимувальною) заданого вигляду іншої функції (апроксимованої), яка задається у будь-якому вигляді (за апроксимації даних вона задається у вигляді масивів даних).

Існує два основних підходи до апроксимації даних. Для одного з них необхідно, щоб апроксимувальна крива (можливо, кусково-гладка) проходила через усі точки, задані таблично. Це реалізується за допомогою методів інтерполяції, які було розглянуто в попередньому підрозділі. За іншого підходу дані апроксимують простою функцією, яка використовується для всіх табличних значень, але не обов'язково, щоб вона проходила через усі точки. Такий підхід називається припасуванням кривої, яку прагнуть провести так, щоб її відхилення від табличних даних було мінімальним. Як правило, використовують метод найменших квадратів (МНК), тобто зводять до мінімуму суму квадратів різниць між значенням функції, яка визначена обраною апроксимувальною кривою і табличними даними.

На рівні із найпоширенішим МНК також використовується метод Чебишова, в якому мінімізується максимальна відстань апроксимувальної кривої від апроксимованої. Взагалі, критерієм близькості можна вибирати будь-яку обгрунтовану постановкою задачі міру, що призводить до використання різних математичних методів і моделей апроксимації, а також алгоритмів пошуку параметрів апроксимувальної функції.

Нехай у таблиці задано  $(n+1)$  точку  $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$  і необхідно визначити апроксимувальну криву  $g(x)$  в діапазоні  $x_0 \leq x \leq x_n$  (рис. 5.6). У цьому випадку похибка в кожній табличній точці:

$$\varepsilon_i = g(x_i) - y_i. \quad (5.25)$$

Тоді сума квадратів похибок визначається виразом:

$$E = \sum_{i=0}^n [g(x_i) - y_i]^2. \quad (5.26)$$

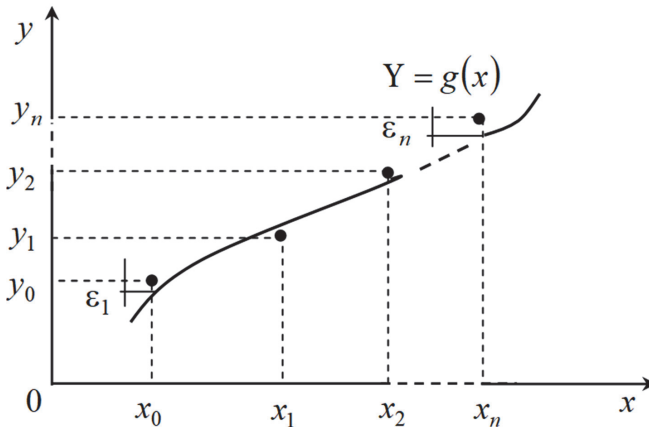


Рисунок 5.6 – Принципова схема апроксимація даних

Як правило, функція  $g(x)$  вибирається у вигляді лінійної комбінації типових функцій  $g_k(x)$ :

$$g(x) = C_1 g_1(x) + C_2 g_2(x) + \dots + C_k g_k(x). \quad (5.27)$$

Умова мінімуму функції  $E$  визначається рівняннями:

$$\frac{\partial E}{\partial C_1} = \frac{\partial E}{\partial C_2} = \dots = \frac{\partial E}{\partial C_k} = 0. \quad (5.28)$$

Оскільки

$$E = \sum_{i=0}^n [C_1 g_1(x_i) + C_2 g_2(x_i) + \dots + C_k g_k(x_i) - y_i]^2,$$



$$\begin{bmatrix} (n+1) & \sum x_i & \sum x_i^2 \\ \sum x_i & \sum x_i^2 & \sum x_i^3 \\ \sum x_i^2 & \sum x_i^3 & \sum x_i^4 \end{bmatrix} \begin{bmatrix} C_0 \\ C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} \sum y_i \\ \sum x_i y_i \\ \sum x_i^2 y_i \end{bmatrix}.$$

Іноді табличні дані розбиваються на декілька частин та добирається окрема апроксимувальна крива для кожної частини. Такий підхід задовольняє ті випадки, коли дані відповідають різним фізичним станам системи.

Залишкова середня квадратична похибка апроксимації оцінюється за допомогою такого виразу:

$$\Delta = \sqrt{E/(n+1)}. \quad (5.31)$$

Алгоритм розв'язання задачі апроксимації методом найменших квадратів набуде вигляду (рис. 5.7).

Якщо під час побудови апроксимувальної функції  $g_i(x)$  використовуються ортогональні поліноми, для яких  $\sum g_j(x_i)g_k(x_i) = 0$  ( $j \neq k$ ), то система (5.29) спрощується, і матриця стає діагональною. Зі свого боку коефіцієнти визначаються із таких співвідношень:

$$C_j = \frac{\sum_{i=0}^n g_j(x_i)y_i}{\sum_{i=0}^n g_j^2(x_i)}. \quad (5.32)$$

Такий підхід дозволяє спростити задачу, що дає змогу в багатьох стандартних програмах для припасування кривих використовувати ортогональні поліноми.

**Приклад 5.7.** Функцію  $f(x)$ , подану у табличній формі (табл. 5.11), необхідно апроксимувати функцією типу:

$$\phi(x) = C_1 x^2 + C_2 x + C_3 + C_4 \sin 2x. \quad (5.33)$$

Таблиця 5.11 – Вихідні дані

$i$	1	2	3	4	5
$x$	0	1	2	3	4
$f(x)$	-1	2	3	-2	5

*Розв'язання:*

Введемо функціональні позначення на основі апроксимувальної функції (5.33):  $g_1(x) = x^2$ ,  $g_2(x) = x$ ,  $g_3(x) = 1$ ,  $g_4(x) = \sin 2x$ .

На основі системи (5.29), записаної у матричній формі, отримаємо:

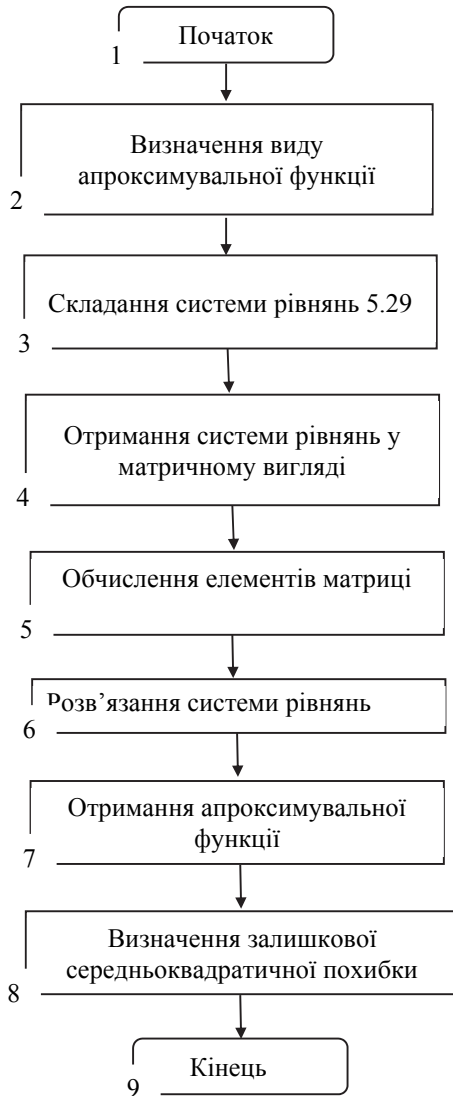


Рисунок 5.7 – Схема алгоритму розв'язання задачі апроксимації методом найменших квадратів



$$\begin{bmatrix} \sum_{i=1}^n (x_i^2)^2 & \sum_{i=1}^n x_i^2 x_i & \sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i^2 \sin 2x_i \\ \sum_{i=1}^n x_i^2 x_i & \sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i & \sum_{i=1}^n x_i \sin 2x_i \\ \sum_{i=1}^n x_i^2 & \sum_{i=1}^n x_i & 1 & \sum_{i=1}^n \sin 2x_i \\ \sum_{i=1}^n x_i^2 \sin 2x_i & \sum_{i=1}^n x_i \sin 2x_i & \sum_{i=1}^n \sin 2x_i & \sum_{i=1}^n \sin^2 2x_i \end{bmatrix} \times \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n f(x_i) x_i^2 \\ \sum_{i=1}^n f(x_i) x_i \\ \sum_{i=1}^n f(x_i) \\ \sum_{i=1}^n f(x_i) \sin 2x_i \end{bmatrix}.$$

Обчислимо всі елементи матриці:

$$\sum_{i=1}^n (x_i^2)^2 = \sum_{i=1}^5 x_i^4 = 0 + 1^4 + 2^4 + 3^4 + 4^4 = 354;$$

$$\sum_{i=1}^n x_i^2 x_i = \sum_{i=1}^5 x_i^3 = 0 + 1^3 + 2^3 + 3^3 + 4^3 = 100;$$

$$\sum_{i=1}^n x_i^2 = 0 + 1^2 + 2^2 + 3^2 + 4^2 = 30;$$

$$\sum_{i=1}^n x_i^2 \sin 2x_i = 0 + \sin 2 + 4 \sin 4 + 9 \sin 6 + 16 \sin 8 = 0,909 - 3,027 + 2,515 + 15,83 = 11,197;$$

$$\sum_{i=1}^n x_i^2 f(x_i) = -1 \cdot 0 + 2 \cdot 1^2 + 3 \cdot 2^2 - 2 \cdot 3^2 + 5 \cdot 4^2 = 2 + 12 - 18 + 80 = 76.$$

Аналогічно виконуються обчислення для решти елементів матриці:

$$\begin{bmatrix} 354,0 & 100,0 & 30,0 & 11,197 \\ 100,0 & 30,0 & 10,0 & 2,515 \\ 30,0 & 10,0 & 1,0 & 0,862 \\ 11,197 & 2,515 & 0,862 & 2,456 \end{bmatrix} \times \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ C_4 \end{bmatrix} = \begin{bmatrix} 76,0 \\ 22,0 \\ 7,0 \\ 5,054 \end{bmatrix}. \quad (5.34)$$

Розв'язуючи систему рівнянь (5.34) (див. розд. 1) буде отримано невідомі коефіцієнти  $C_{1,2,3,4}$ :  $C_1 = -0,0376$ ;  $C_2 = 0,6645$ ;  $C_3 = 0,2115$ ;  $C_4 = 1,4745$ .

Отже, апроксимувальна функція отримує такий вигляд:

$$\varphi(x) = -0,0376x^2 + 0,6645x + 0,2115 + 1,4745 \sin 2x.$$

Для повного розв'язку необхідно визначити залишкову похибку. Згідно з формулою (5.31), необхідно визначити суму квадратів відхилень отриманої функції  $\varphi(x)$  від заданої  $f(x)$ . Спочатку визначається значення

функції  $\varphi(x)$  в заданих точках, після чого обчислюється квадрат різниці між двома функціями (табл. 5.12).

Таблиця 5.12 – Результат обчислення квадратів різниць між двома функціями

$i$	0	1	2	3	4
$x$	0,000	1,000	2,000	3,000	4,000
$f(x)$	-1,000	2,000	3,000	-2,000	5,000
$\varphi(x)$	0,211	2,179	0,274	1,455	3,727
$\varphi(x) - f(x)$	1,212	0,197	-2,726	3,455	-1,273
$(\varphi(x) - f(x))^2$	1,468	0,032	7,430	11,934	1,621

Із таблиці 5.12 значення суми квадратів похибок:

$$E = \sum_{i=0}^4 [\varphi(x_i) - f(x_i)]^2 = 1,468 + 0,032 + 7,430 + 11,934 + 1,621 = 22,485.$$

Тоді значення залишкової середньоквадратичної похибки:

$$\Delta = \sqrt{\frac{E}{n+1}} = \sqrt{\frac{22,485}{5+1}} = 1,936.$$

З практичного погляду, отримана середньоквадратична похибка  $\Delta=1,936$  апроксимації функції  $f(x)$  досить велика. Щоб зменшити похибку апроксимації необхідно попередньо підібрати оптимальний тип функції  $\varphi(x)$ .

### 5.3 Статистична обробка даних

Під час обробки результатів експериментальних даних виникає необхідність оцінення характеристик випадкової величини.

Для оцінення  $\bar{X}$  невідомого значення математичного сподівання  $m_X$  випадкової величини  $X$  використовується середнє арифметичне результатів  $N$  незалежних випробувань:

$$\bar{X} = \frac{\sum_{i=1}^n x_i}{N}, \quad (5.35)$$

а для оцінення значення дисперсії  $D_x$  за достатньо великої кількості експериментальних даних ( $N \geq 30$ ) – співвідношення:

$$D_x = \sigma_x^2 = \frac{\sum_{i=1}^N (x_i - \bar{x})^2}{N-1}. \quad (5.36)$$

Розглянемо реалізацію знаходження математичного сподівання та дисперсії мовою програмування C++:

```
#include <cmath>
#include <iostream>
using namespace std;

double sum(int size, double sample[]){
    double sum = 0;
    for (int i = 0; i < size; i++){
        sum+=sample[i];
    }
    return sum;
}

double mExp(int size, double sample[]){
    return sum(size, sample)/size;
}

double variance(int size, double sample[]){
    double mean = mExp(size, sample);
    double diffSquared[size];
    for (int i=0; i<size;i++) {
        diffSquared[i] = pow(sample[i]-mean,2);
    }
    return sum(size,diffSquared)/(size-1);
}

int main(int argc, char *argv[]) {
    int size = argc-2;
    double sample [size];

    for (int i = 0; i < size; i++){
        sample[i]=stod(*(argv+2+i));
    }
    string mode(*(argv+1));
    cout << "Mode: " << mode << endl;
    if ("expectation" == mode) {
        cout << "Expectation: " << mExp(size,sample) << endl;
    } else if ("variance" == mode) {
        cout << "Variance: " << variance(size,sample) << endl;
    } else {
        cout << "Unknown mode" << endl;
    }
    return 0;
}.
```

Якщо розглядається нормальний закон розподілу величини  $X$ , тоді можна показати, що величина:

$$T = \frac{\bar{X} - m_X}{\sigma_X / \sqrt{N}},$$

має  $t$ -розподіл Стьюдента із  $k=N-1$  ступенями вільності (ступінь вільності в статистиці визначається як різниця між кількістю дослідів та кількістю коефіцієнтів моделі, які можна обчислити за результатами цих експериментів незалежно один від одного (наприклад, у нормального

розподілу два параметри, а у Пуассонівського один і т. д.). Звідси можна визначити довірчий інтервал для істинного значення  $x$ : за відомими значеннями довірчої ймовірності  $P$ , а саме з таблиці 5.13 визначається значення  $\varepsilon$ , звідки  $\Delta = \varepsilon \frac{D_x}{\sqrt{N}}$ .

Таким чином, якщо випадкова величина  $x$  розподілена за нормальним законом із математичним сподіванням  $m_x$  та дисперсією  $D_x$ , тоді істинне значення  $x$  знаходиться в інтервалі  $(m_x - \Delta, m_x + \Delta)$  з довірчою ймовірністю  $P$ .

Для оцінення виду закону розподілу за довірчого закону розподілу найпоширеніше застосування мають критерії Колмогорова та Пірсона (критерій Стьюдента застосовується тільки за нормального закону розподілу), які дозволяють на основі порівняння емпіричної функції розподілу  $f_x^*(x)$ , отриманої у вигляді гістограми внаслідок обробки експериментальних даних з гіпотетичною  $f_x(x)$ , що відповідає запропонованій гіпотезі, зробити висновок про їх збіг за рівня значущості  $\alpha$ , який визначається як ймовірність того, що буде відхилена достовірна гіпотеза.

Таблиця 5.13 – Значення для інтервалу  $-\varepsilon < t < \varepsilon$ , де величина  $t$  має розподіл Стьюдента залежно від надійної ймовірності  $P$  і числа ступенів вільності  $k$

$k$	$P=0,90$	$P=0,95$	$P=0,99$
$l$	$2$	$3$	$4$
1	6,310	12,71	63,7
2	2,920	4,30	9,92
3	2,350	3,18	5,84
4	2,130	2,77	4,60
5	2,020	2,57	4,03
6	1,943	2,45	3,71
7	1,895	2,36	3,50
8	1,860	2,31	3,36
9	1,833	2,26	3,25
10	1,812	2,23	3,17
11	1,796	2,20	3,11
12	1,782	2,18	3,06
13	1,771	2,16	3,01
14	1,761	2,14	1,98
15	1,753	2,13	2,95
16	1,746	2,12	2,92
17	1,740	2,11	2,90
18	1,734	2,10	2,86

Продовження таблиці 5.13

1	2	3	4
19	1,729	2,09	2,86
20	1,725	2,08	2,84
22	1,717	2,07	2,82
24	1,711	2,06	2,80
26	1,706	2,06	2,78
28	1,701	2,05	2,76
30	1,697	2,04	2,75
40	1,684	2,02	2,70
60	1,671	2,00	2,66
120	1,658	1,98	2,62
240	1,645	1,96	2,58

У критерії Колмогорова мірою є величина:

$$\lambda = |f_X(x) - f_X^*(x)|_{\max} \sqrt{n},$$

яку порівнюють з критичним значенням, заданим у таблиці 5.14.

Таблиця 5.14 – Критичні значення  $\lambda_0$  залежно від рівня значущості

$\alpha$	0,500	0,400	0,300	0,200	0,100	0,050	0,020	0,001	0,001
$\lambda_0$	0,828	0,895	0,974	1,073	1,224	1,358	1,520	1,627	1,950

За умови  $\lambda < \lambda_{кр}$  гіпотеза про збіг  $f_X(x)$  та  $f_X^*(x)$  приймається.

У критерії Пірсона обчислюється величина:

$$\chi^2 = \sum_{i=0}^k \frac{[f_X(x_i) - f_X^*(x_i)]^2}{f_X(x_i)}, \quad (5.37)$$

де  $k$  – число розрядів гістограми (дискретних значень  $f_X(x_i)$ ).

З таблиці 5.15 визначають критичне значення  $\chi^2$ , враховуючи значення  $\alpha$  і число ступенів вільності:

$$r = k - l - 1,$$

де  $l$  – число параметрів, що їх містить у собі закон розподілу (для нормального  $l=2$ , пуассонівського  $l=1$  і т. д.).

За  $\chi^2 < \chi^2_{кр}$  гіпотеза приймається.

Якщо порівнюють аналітично отримані закони розподілу ймовірностей, то мірою їх близькості є значення середньої квадратичної похибки.

Для оцінення взаємозалежності випадкових величин, між якими існує стохастичний зв'язок, використовується коефіцієнт кореляції (*correlation coefficient*):

$$r_{xy} = \frac{1}{n-1} \frac{\sum_{i=1}^n (x_i - m_x)(y_i - m_y)}{\sigma_x \sigma_y}, \quad (5.38)$$

де  $n$  – об'єм вибірки.

Під час визначення взаємозалежності значень випадкових величин у різні моменти часу коефіцієнт кореляції оцінюється за формулою:

$$r_x(\tau) = \frac{1}{n-m-1} \frac{\sum_{i=1}^{n-m} [x(t_i) - m_x][x(t_i + \tau) - m_x]}{D_x}, \quad (5.39)$$

де  $x(t_i)$  – значення випадкової величини  $X$  в момент часу  $t_i$ , а  $x(t_i + \tau)$  – в момент часу, який відрізняється від  $t_i$  на інтервалі часу  $\tau$ . Таким чином,  $x(t_i) = x_i$ ,  $x(t_i + \tau) = x_j$ ,  $\tau$  – інтервал часу між  $i$  та  $j$  значеннями  $x$  ( $i-j = m$ ).

Таблиця 5.15 – Критичні точки розподілу  $x$  – випадкова величина, яка розподілена за законом  $\chi^2$  із  $k$  ступенями вільності

Число ступенів свободи $k$	$\alpha=0,010$	$\alpha=0; 0,025$	$\alpha=0,050$	$\alpha=0,950$	$\alpha=0,975$	$\alpha=0,990$
1	6,6	6,0	3,8	0,0039	0,00098	0,00016
2	9,2	7,4	6,0	0,103	0,051	0,020
3	11,3	9,4	7,8	0,352	0,216	0,115
4	13,3	11,1	9,5	0,711	0,484	0,297
5	15,1	12,8	11,1	1,15	0,831	0,554
6	16,8	14,4	12,6	1,64	1,24	0,872
7	18,5	16,0	14,1	2,17	1,69	1,24
8	20,1	17,5	15,5	2,73	2,18	1,65
9	21,7	19,0	16,9	3,33	2,70	2,09
10	23,2	20,5	18,3	3,94	3,25	2,56
11	24,7	21,9	19,7	4,57	3,82	3,05
12	26,2	23,3	21,0	5,23	4,40	3,57
13	27,7	24,7	22,4	5,89	5,01	4,11
14	29,1	26,1	23,7	6,57	5,63	4,66
15	30,6	27,5	25,0	7,26	6,26	5,23
16	32,0	28,8	26,3	7,96	6,91	5,81

Інтервал кореляції визначається як відрізок часу, за який кореляційна функція зменшується на 95%.

Визначення коефіцієнта кореляції (нормованої кореляційної функції) та кореляційної функції за відомими масивами даних  $x$  та  $y$  на основі наведених формул не викликає труднощів, а апроксимація вигляду кореляційної

функції типовими кореляційними функціями (табл. 5.16) може здійснюватися за методом найменших квадратів.

Таблиця 5.16 – Типові кореляційні функції

Вигляд	Параметри
$R_x(\tau) = \sigma_x^2(1 - \alpha \tau ),$ $\tau < 1/\alpha$	$\alpha = (\sigma_x^2 - R_x(\tau^*)) / \sigma_x^2 \tau^*,$ $R_x(\tau^*)$ – відоме значення кореляційної функції
$R_x(\tau) = \sigma_x^2 e^{-\alpha \tau }$	$\alpha = \frac{1}{\tau^*} \ln \frac{\sigma_x^2}{R_x(\tau^*)}$
$R_x(\tau) = \sigma_x^2 e^{-\alpha^2 \tau^2}$	$\alpha = \frac{1}{\tau^*} \sqrt{\ln \frac{\sigma_x^2}{R_x(\tau^*)}}$
$R_x(\tau) = \sigma_x^2 e^{-\alpha \tau } (1 + \alpha \tau )$	$\alpha \approx 4,5 / \tau_k^{max}$
$R_x(\tau) = \sigma_x^2 e^{-\alpha \tau } \cos \beta \tau$	$\alpha = \frac{\ln \sigma_x^2 \cos \frac{\pi \tau_2^*}{2 \tau_1^*}}{\tau_2^* * R_x(\tau_2^*)}, \beta = \pi / 2 \tau_1^*$ за двох відомих значень кореляційної функції $R_x(\tau)$ , причому $R_x(\tau_1^*) = 0$ .

### Висновки щодо застосування методів розв'язання задач обробки даних

Задачі обробки даних об'єднують низку питань, що виникають у дослідників під час обробки експериментальних даних чи в процесі випробувань з досліджуваними об'єктами. Задачі наближення невідомих функцій загалом – це задачі апроксимації. Якщо отримання функції заданого вигляду є кінцевим результатом, то сам процес розв'язання задачі визначається вибором критерія близькості апроксимованої та апроксимувальної функцій. Найчастіше таким методом виступає метод найменших квадратів, де міра близькості – сума квадратів відхилень цих функцій, але бувають й інші постановки задачі. У загальному випадку критерій близькості вибирає сам дослідник. Коли наближення використовується для оцінення значення невідомої функції у певній точці, то розглядається задача інтерполяції, що розв'язується відомими методами Лагранжа, різницевиими, сплайнами тощо. Вибір ефективного алгоритму залежить від відповідей на питання: наскільки точним є вибраний метод, які витрати машиночасу йдуть на його використання, наскільки гладкою є інтерполяційна функція, яку кількість точок даних вона потребує тощо.

Похибки методів інтерполяції залежать від того, чи всі точки, отримані з експерименту, використано. Поліном Лагранжа явно містить значення функцій у вузлах інтерполяції, тому він корисний, коли значення функцій змінюються, а вузли інтерполяції незмінні. Ефективність застосування різницевих методів інтерполяції залежить від розташування точки, в якій виконується пошук значення функції. У разі її розташування на початку інтервалу зручніше користуватися першою інтерполяційною формулою Ньютона, в кінці – другою, а в середині – формулами, що базуються на центральних різницях. Сплайни ефективно використовуються за необхідності багаторазових інтерполяційних обчислень і у разі великої незмінної бази даних. Вони також ефективні у випадку багатомірної інтерполяції і широко використовуються (просторова інтерполяція на основі кривих Без'є) для відновлення та обробки просторових зображень у комп'ютерній графіці. Екстраполяцією називають процедуру інтерполяції за межами заданого інтервалу. Під час статистичної обробки даних необхідно виконувати оцінювання основних характеристик випадкової величини. Саме ці процедури входять у набір класичних задач обробки експериментальних даних в поєднанні із алгоритмами та методами їх розв'язання, що обов'язково мають бути у арсеналі інженера та дослідника.

### Контрольні запитання та завдання

1. Сформулювати задачу інтерполяції. В яких випадках інтерполяція неможлива?
2. Які обмеження накладаються на набір базових точок під час розв'язання задачі інтерполяції?
3. Скласти алгоритм та програму для розв'язання задачі інтерполяції за методом Лагранжа.
4. Скласти алгоритм та програму для розв'язання задачі інтерполяції першою інтерполяційною формулою Ньютона.
5. Скласти алгоритм та програму для розв'язання задачі інтерполяції другою інтерполяційною формулою Ньютона.
6. Скласти алгоритм та програму для розв'язання задачі інтерполяції з використанням інтерполяційних формул із центральними різницями.
7. Вивести інтерполяційну формулу Ньютона для рівновіддалених вузлів.
8. Що таке сплайн-інтерполяція? Як визначаються коефіцієнти сплайнів?
9. Що таке екстраполяція?
10. Що є більш узагальненим поняттям – екстраполяція чи інтерполяція?



11. Що таке апроксимація? Чим відрізняється апроксимація від інтерполяції?

12. Якими шляхами розв'язується задача апроксимації?

13. Наведіть головну формулу для методу найменших квадратів.

14. Виведіть систему рівнянь для визначення коефіцієнтів апроксимувального полінома в методі найменших квадратів.

15. Які поліноми називаються ортогональними? Наведіть приклади.

16. Побудуйте інтерполяційний многочлен Лагранжа  $n$ -го ступеня для функції  $y(x)$  на заданому проміжку та обчисліть його значення в заданій точці за розбиття інтервалу на 10 та 20 точок, порівняйте похибки.

а)  $y = \frac{2x}{3x^2 + 4x + 1}$  на інтервалі  $x \in [2; 4]$ ,  $x=3,0$ ;

б)  $y = \frac{1}{3\sqrt{x} + 2}$  на інтервалі  $x \in [1; 3]$ ,  $x=2,0$ ;

в)  $y = \frac{1}{2x^2 + 3x + 5}$  на інтервалі  $x \in [2; 3]$ ,  $x=2,2$ ;

г)  $y = \frac{x}{\lg(2 + x)}$  на інтервалі  $x \in [4; 5]$ ,  $x=4,3$ .

17. Розв'яжіть задачу інтерполяції за методом Лагранжа для функції, заданої таблично та обчисліть значення функції в точці  $x=2,2$ :

а)

$i$	0	1	2	3
$x_i$	2	4	5	8
$y_i$	10	15	9	25

б)

$i$	0	1	2	3
$x_i$	1	3	5	8
$y_i$	11	5	9	12

в)

$i$	0	1	2	3
$x_i$	-5	-3	0	3
$y_i$	-24	-12	4	22

18. Знайти значення інтерполяційної функції в точці  $x=2$ ,  $y=5$  методом Лагранжа для функцій двох змінних, заданої таблично:

$i$	0	1	2	3	4
$x_i$	-2	0	1	3	4
$y_i$	4	2	-5	12	22
$z_i$	17	14	22	30	12

19. Побудуйте діагональну таблицю скінченних різниць для заданої функції, використовуючи першу інтерполяційну формулу Ньютона та обчисліть значення інтерполяції в заданій точці за розбиття інтервалу на 10 та 20 точок, порівняйте похибки обчислення.

а)  $y(x) = \frac{1}{2x^3 + 3x^2 + 1}$  на інтервалі  $x \in [4; 5]$ ,  $x=4,2$ ;

б)  $y(x) = \frac{1}{\sqrt[3]{3x^2 + 5}}$  на інтервалі  $x \in [6; 7]$ ,  $x=6,4$ ;

в)  $y(x) = \frac{1}{\sqrt{x^3 + \sqrt{3x}}}$  на інтервалі  $x \in [1; 3]$ ,  $x=1,2$ .

20. Розв'яжіть задачу інтерполяції, використовуючи першу інтерполяційну формулу Ньютона та обчисліть значення функції в точці  $x=1,5$ .

а)

$i$	0	1	2	3
$x_i$	0	2	4	6
$y_i$	-2	15	7	24

б)

$i$	0	1	2	3
$x_i$	-2	0	2	4
$y_i$	10	5	7	12

в)

$i$	0	1	2	3
$x_i$	-5	0	5	10
$y_i$	-24	-12	4	22

21. Побудуйте діагональну таблицю скінченних різниць для заданої функції, використовуючи другу інтерполяційну формулу Ньютона, та обчисліть значення інтерполяції в заданій точці за розбиття інтервалу на 10 та 20 точок, порівняйте похибки обчислення.

а)  $y(x) = \frac{1}{x^4 + 3x^3 + x}$  на інтервалі  $x \in [1; 3]$ ,  $x=1,5$ ;

б)  $y(x) = \frac{1}{5x\sqrt{x + x^5}}$  на інтервалі  $x \in [3; 5]$ ,  $x=4,0$ ;

в)  $y(x) = \frac{1}{4x^3 + 5x + \sqrt{x}}$  на інтервалі  $x \in [6; 8]$ ,  $x=7,6$ .

22. Розв'яжіть задачу інтерполяції, використовуючи другу інтерполяційну формулу Ньютона та обчисліть значення функції в точці  $x=6,2$ .

а)

$i$	0	1	2	3
$x_i$	2	4	6	8
$y_i$	10	-2	9	5

б)

$i$	0	1	2	3
$x_i$	1	3	5	7
$y_i$	11	2	9	10

в)

$i$	0	1	2	3
$x_i$	-4	0	4	8
$y_i$	-24	-12	4	22

23. Побудуйте діагональну таблицю скінченних різниць для заданої функції, використовуючи формулу Стірлінга, та обчисліть значення інтерполяції в заданій точці за розбиття інтервалу на 10 та 20 точок, порівняйте похибки обчислення.

а)  $y(x) = \frac{1}{2x^3 + 3x^2 + 1}$  на інтервалі  $x \in [4; 6]$ ,  $x=5,0$ ;

б)  $y = \sin(x) + x^2 - x^3$  на інтервалі  $x \in [6; 7]$ ,  $x=6,8$ ;

в)  $y(x) = \frac{1}{\sqrt{x^3 + \sqrt{3x}}}$  на інтервалі  $x \in [1; 3]$ ,  $x=1,2$ .

24. Розв'яжіть задачу інтерполяції, використовуючи формулу Стірлінга та обчисліть значення функції в точці  $x=2,2$ .

а)

$i$	0	1	2	3
$x_i$	0	2	4	6
$y_i$	-2	15	7	24

б)

$i$	0	1	2	3
$x_i$	-3	0	3	6
$y_i$	10	5	7	12

в)

$i$	0	1	2	3
$x_i$	-4	0	4	8
$y_i$	-15	-12	4	10

25. Побудуйте сплайни третього порядку для функції  $y(x_i)$ , заданої таблично:

а)

$i$	0	1	2	3
$x_i$	2	4	5	8
$y_i$	10	15	9	25

б)

$i$	0	1	2	3
$x_i$	1	3	5	8
$y_i$	11	5	9	12

в)

$i$	0	1	2	3
$x_i$	-5	-3	0	3
$y_i$	-24	-12	4	22

26. Використовуючи метод найменших квадратів, апроксимуйте задані табличні дані функцією такого вигляду:

а)  $\varphi(x) = C_1 + C_2x$ ;

б)  $\varphi(x) = C_1 + C_2x + C_3x^2$ ;

в)  $\varphi(x) = C_1 + C_2x + C_3 \sin(3x)$ .

Визначіть відповідні коефіцієнти  $C_1, C_2, C_3$  та залишкову похибку.

$i$	0	1	2	3	4
$x$	-4,0	1,0	3,0	5,0	12,0
$f(x)$	2,4	-5,0	1,2	7,0	4,0

27. Розробіть алгоритм та складіть програму для оцінення статистичних характеристик результатів вимірювання випадкової величини  $X$ .

28. Складіть алгоритм та програму для оцінення коефіцієнта кореляції двох випадкових величин.

29. Складіть алгоритм та програму для побудови гістограми випадкової величини, а також побудуйте гістограму для випадкової величини, отриманої зі стандартного генератора випадкових чисел.

30. Складіть алгоритм та програму для оцінення коефіцієнта автокореляції випадкової величини та апроксимуйте отриману функцію однією з типових кореляційних функцій з таблиці 5.16.

## РОЗДІЛ 6 ЧИСЕЛЬНЕ ІНТЕГРУВАННЯ ТА ДИФЕРЕНЦЮВАННЯ

У багатьох задачах, пов'язаних з розробкою, аналізом, ідентифікацією й оцінюванням якості різних методів і засобів математичного моделювання, а також інформаційних технологій, виникає необхідність обчислення певних інтегралів.

Функція  $F(x)$  на заданому проміжку  $D$  називається первісною функцією для функції  $f(x)$  або інтегралом від  $f(x)$ , якщо у всьому цьому проміжку  $f(x)$  є похідною для функції  $F(x)$ , або те саме, що і  $f(x)dx$  слугує для  $F(x)$  диференціалом:

$$F'(x) = f(x) \text{ або } dF(x) = f(x)dx.$$

Якщо функція  $f(x)$  неперервна на відрізку  $[a; b]$  й відома її первісна функція  $F(x)$ , то визначений інтеграл від  $a$  до  $b$  може бути обчислений за допомогою формули Ньютона-Лейбніца:

$$I = \int_a^b f(x)dx = F(b) - F(a). \quad (6.1)$$

Графічна інтерпретація інтеграла – це площа криволінійної трапеції, обмежена кривою  $y = f(x)$ , двома ординатами  $x_1 = a$  і  $x_2 = b$  та відрізком осі  $x$ .

Дуже часто обчислення значення інтеграла є не те що складним процесом (велика складність аналітичних перетворень), а й узагалі неможливим (наявність невластних інтегралів), коли підінтегральна функція задана набором числових даних (експериментальні дані).

Тому задача чисельного інтегрування (*numerical integration*) функції полягає в обчисленні значення визначеного інтеграла на основі ряду значень підінтегральної функції (заміна вихідної підінтегральної функції певною апроксимувальною функцією). Формули чисельного інтегрування часто називають квадратурними.

Найбільш відомими методами знаходження визначених інтегралів є:

- формули прямокутників;
- методи Ньютона-Котеса, Гаусса, Чебишова, в основі яких покладено використання так званих квадратурних формул, отриманих заміною  $f(x)$  інтерполяційними многочленами;
- методи Монте-Карло, які базуються на використанні статистичних моделей.

### 6.1 Метод прямокутників

Нехай потрібно визначити значення інтеграла функції  $f(x)$  на відрізку  $[a; b]$ . Ідея методу прямокутників полягає в розбитті відрізка інтегрування  $[a; b]$  на елементарні відрізки  $[x_{i-1}; x_i]$  точками  $a = x_0 < x_1 < \dots < x_n = b$ , на основі яких будуються прямокутники, що мають висоту  $f(\xi_i)$ . За

рівномірного розбиття відрізка  $x_i = a + i \cdot h$  ( $h$  – крок), тому  $h = \frac{(a-b)}{n}$ .

Значення інтеграла функції  $f(x)$  приблизно виражається сумою площ побудованих прямокутників. Узагальнена квадратурна формула прямокутників має вигляд:

$$I = \int_a^b f(x) dx \approx \sum_{i=1}^n f(\xi_i)(x_i - x_{i-1}), \quad (6.2)$$

де точка  $\xi_i \in [x_i; x_{i-1}]$ .

Залежно від вибору положення точки  $\xi_i$  розрізняють формули лівих, правих й середніх прямокутників.

За  $\xi_i = x_{i-1}$  формула лівих прямокутників із першим порядком точності –  $O(h)$ :

– для нерівновіддалених вузлів

$$I \approx \sum_{i=1}^n f(\xi_i)(x_i - x_{i-1}); \quad (6.3)$$

– для рівновіддалених вузлів

$$I \approx h \sum_{i=1}^n f(x_{i-1}). \quad (6.4)$$

Геометричну інтерпретацію наведено на рисунку 6.1, а).

За  $\xi_i = x_i$  формула правих прямокутників із першим порядком точності –  $O(h)$ :

– для нерівновіддалених вузлів

$$I \approx \sum_{i=1}^n f(x_i)(x_i - x_{i-1}); \quad (6.5)$$

– для рівновіддалених вузлів

$$I \approx h \sum_{i=1}^n f(x_i). \quad (6.6)$$

Геометричну інтерпретацію наведено на рис. 6.1, б).

У випадку  $\xi_i = \frac{x_{i-1} + x_i}{2}$ , формула середніх прямокутників із другим порядком точності –  $O(h^2)$ :

– для нерівновіддалених вузлів

$$I \approx \sum_{i=1}^n f\left(\frac{x_i + x_{i-1}}{2}\right)(x_i - x_{i-1}); \quad (6.7)$$

– для рівновіддалених вузлів

$$I \approx h \sum_{i=1}^n f\left(x_{i-1} + \frac{h}{2}\right). \quad (6.8)$$

Геометричну інтерпретацію наведено на рисунку 6.1, в).

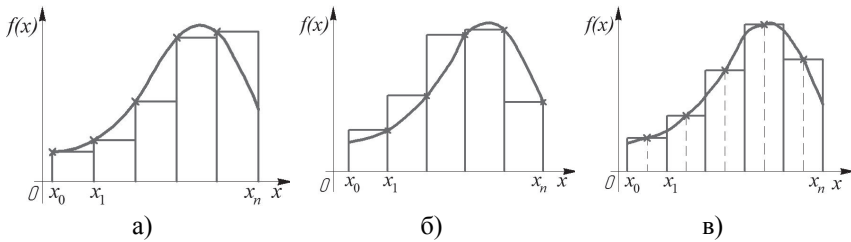


Рисунок 6.1 – Схема чисельного інтегрування методом прямокутників:  
а) лівих; б) правих; в) середніх

Формули лівих та правих прямокутників можуть бути використані як для аналітично заданих функцій, так і для функцій, заданих таблично. Метод середніх прямокутників може використовуватись для пошуку інтегралів тільки від аналітично заданих функцій.

**Приклад 6.1** Визначити значення інтеграла  $I = \int_0^1 \frac{dx}{1+x^2}$  методом лівих, правих та середніх прямокутників для значення обчислювального кроку  $h = 0,2$ .

*Розв'язання:*

За формулою лівих прямокутників (6.4) отримаємо:

$$I_{\text{лн}} = h \sum_{i=1}^5 f(x_{i-1}) = h \cdot (f(0) + f(0,2) + f(0,4) + f(0,6) + f(0,8)) =$$

$$= 0,2 \left( \frac{1}{1,0+0^2} + \frac{1}{1,0+0,2^2} + \frac{1}{1,0+0,4^2} + \frac{1}{1,0+0,6^2} + \frac{1}{1,0+0,8^2} \right) = 0,833732.$$

За формулою правих прямокутників (6.6) отримаємо:

$$I_{\text{пн}} = h \sum_{i=1}^5 f(x_i) = h \cdot (f(0,2) + f(0,4) + f(0,6) + f(0,8) + f(1,0)) =$$

$$= 0,2 \left( \frac{1}{1,0+0,2^2} + \frac{1}{1,0+0,4^2} + \frac{1}{1,0+0,6^2} + \frac{1}{1,0+0,8^2} + \frac{1}{1,0+1,0^2} \right) = 0,733732.$$

За формулою середніх прямокутників (6.8) отримаємо:

$$I_{cn} = h \sum_{i=1}^5 f\left(x_{i-1} + \frac{h}{2}\right) = h \left[ f\left(0 + \frac{0,2}{2}\right) + f\left(0,2 + \frac{0,2}{2}\right) + f\left(0,4 + \frac{0,2}{2}\right) + f\left(0,6 + \frac{0,2}{2}\right) + f\left(0,8 + \frac{0,2}{2}\right) \right] =$$

$$= 0,2 \left( \frac{1}{1,0 + 0,1^2} + \frac{1}{1,0 + 0,3^2} + \frac{1}{1,0 + 0,5^2} + \frac{1}{1,0 + 0,7^2} + \frac{1}{1,0 + 0,9^2} \right) = 0,786826.$$

Точне значення інтеграла на основі аналітичного розв'язку:

$$I = \int_0^1 \frac{dx}{1+x^2} = \arctg(x) \Big|_0^1 = \frac{\pi}{4} = 0,785398.$$

Відносна похибка обчислення шляхом порівняння між значеннями аналітичного розв'язку інтеграла і чисельним методом за формулою:

– лівих прямокутників

$$\varepsilon_n = \left| \frac{I - I_{ln}}{I} \right| \cdot 100\% = \left| \frac{0,785398 - 0,833732}{0,785398} \right| \cdot 100\% = 6,15\%,$$

– правих прямокутників

$$\varepsilon_n = \left| \frac{I - I_{rn}}{I} \right| \cdot 100\% = \left| \frac{0,785398 - 0,733732}{0,785398} \right| \cdot 100\% = 6,58\%,$$

– середніх прямокутників

$$\varepsilon_c = \left| \frac{I - I_{cn}}{I} \right| \cdot 100\% = \left| \frac{0,785398 - 0,786826}{0,785398} \right| \cdot 100\% = 0,18\%.$$

Метод лівих та правих трикутників для заданого рівняння дають суттєву похибку обчислення. Найбільш точний результат дозволяє отримати метод середніх прямокутників. У разі збільшення кількості інтервалів (зменшенні значення  $h$ ) точність обчислення інтеграла буде збільшуватись.

## 6.2 Формули Ньютона – Котеса

Для виведення формул Ньютона-Котеса інтеграл записується у вигляді:

$$\int_a^b f(x) dx = \sum_{i=0}^n A_i f(x_i) + \Delta, \quad (6.9)$$

де  $x_i$  – вузли інтерполяції;

$A$  – коефіцієнти, які залежать від виду формули;

$\Delta$  – похибка квадратурної формули.



Замінюючи в рівнянні (6.9) підінтегральну функцію відповідним інтерполяційним поліномом Лагранжа для  $n$  рівновіддалених вузлів із кроком  $h = \frac{b-a}{n}$ , можна отримати таку формулу для розрахунку коефіцієнтів  $A_i$  за довільної кількості вузлів:

$$A_i = \left( \frac{b-a}{n} \right) \frac{(-1)^{n-1}}{i!(n-1)!} \int_0^m \frac{q(q-1)\dots(q-n)}{(q-i)} dq \quad (i=0, 1, 2, \dots, n), \quad (6.10)$$

де  $q = \frac{x-a}{h}$  – приведена змінна.

Зазвичай коефіцієнти  $H_i = \frac{A_i}{b-a}$  називаються коефіцієнтами Котеса. Формула (6.9) набуде вигляду:

$$\int_a^b f(x) dx = (b-a) \sum_{i=0}^n H_i f(x_i), \quad (6.11)$$

та має такі властивості  $\sum_{i=0}^n H_i = 1$  і  $H_i = H_{n-i}$ .

За  $n = 1$  і  $n = 2$  із (6.10) та (6.11) отримуємо формули трапецій і Сімпсона. У таблиці 6.1 наведено значення коефіцієнтів Котеса для  $n = 1, 2, \dots, 8$ . Оскільки коефіцієнти Котеса за великого числа ординат складні, практично для приблизного обчислення визначених інтегралів проміжок інтегрування розбивається на велике число дрібних проміжків і до кожного з них застосовується квадратурна формула Ньютона-Котеса із малим числом ординат. Після чого буде отримано формули більш простої структури, точність яких може бути достатньо високою.

Таблиця 6.1 – Значення коефіцієнтів формули Ньютона-Котеса

$\hat{H}_i = H_i$	$\hat{H}_0$	$\hat{H}_1$	$\hat{H}_2$	$\hat{H}_3$	$\hat{H}_4$	$\hat{H}_5$	$\hat{H}_6$	$\hat{H}_7$	$\hat{H}_8$	Загальний знаменник $N$
1	1	1								2
2	1	4	1							6
3	1	3	3	1						8
4	7	32	12	32	7					90
5	19	75	50	50	75	19				288
6	41	216	27	272	27	216	41			840
7	751	3577	1223	2989	2989	1223	3577	751		17280
8	989	5888	-928	10496	-4540	10496	-928	5888	989	28350

Наприклад, отримані таким чином формули трапецій і Сімпсона (парабол) мають такий вигляд:

$$I = \frac{h}{2} [f(x_0) + 2f(x_1) + 2f(x_2) + \dots + 2f(x_{n-1}) + f(x_n)], \quad (6.12)$$

$$I = \frac{h}{3} [f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \dots + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)]. \quad (6.13)$$

Причому похибки складових формул становлять, відповідно:

$$\Delta = -n \frac{h^3}{12} M_2 \quad \text{та} \quad \Delta = -n \frac{h^5}{180} M_4.$$

Аналогічно можна отримати складові формули Ньютона – Котеса для більш високих порядків.

Для оцінення похибки обчислення на практиці використовується метод Рунге (екстраполяції Річардсона) аналогічно використанню для однокрокових чисельних методів розв'язання задачі Коші (див. розд. 3).

**Приклад 6.2.** Визначіть значення інтеграла  $I = \int_0^1 \frac{dx}{1+x^2}$  за допомогою чисельного методу Ньютона-Котеса, а саме за допомогою формули трапецій із кроком  $h = 0,2$  і формули Сімпсона із кроком  $h = 0,25$ .

*Розв'язання:*

Оскільки значення підінтегральної функції  $f(x) = \frac{1}{1+x^2}$ , то за  $h = 0,2$  буде отримано розбиття інтервалу  $x \in [0; 1,0]$  на п'ять відрізків. Використовуючи формулу трапецій (6.13), маємо:

$$\begin{aligned} I_m &= \frac{h}{2} [f(x_0) + 2f(x_1) + 2f(x_2) + 2f(x_3) + 2f(x_4) + f(x_5)] = \\ &= \frac{h}{2} [f(0) + 2 \cdot f(0,2) + 2 \cdot f(0,4) + 2 \cdot f(0,6) + 2 \cdot f(0,8) + f(1,0)] = \\ &= \frac{0,2}{2} \left[ \frac{1}{1,0+0^2} + 2 \cdot \frac{1}{1,0+0,2^2} + 2 \cdot \frac{1}{1,0+0,4^2} + 2 \cdot \frac{1}{1,0+0,6^2} + 2 \cdot \frac{1}{1,0+0,8^2} + \right. \\ &\quad \left. + \frac{1}{1,0+1,0^2} \right] = 0,783730. \end{aligned}$$

Також, використовуючи формулу Сімпсона (6.13) із кроком  $h = 0,25$ , отримаємо:

$$\begin{aligned}
 I_c &= \frac{h}{3} [f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + f(x_4)] = \\
 &= \frac{h}{3} [f(0) + 4f(0,25) + 2f(0,5) + 4f(0,75) + f(1,0)] = \\
 &= \frac{0,2}{3} \left[ \frac{1}{1,0+0^2} + 4 \cdot \frac{1}{1,0+0,25^2} + 2 \cdot \frac{1}{1,0+0,5^2} + 4 \cdot \frac{1}{1,0+0,75^2} + \frac{1}{1,0+1,0^2} \right] = \\
 &= 0,785392.
 \end{aligned}$$

Відносна похибка обчислення шляхом порівняння між значеннями аналітичного розв'язку інтеграла (див. приклад 6.1) і чисельним методом за формулою:

– трапецій

$$\varepsilon_m = \left| \frac{I - I_m}{I} \right| \cdot 100\% = \left| \frac{0,785398 - 0,783730}{0,785398} \right| \cdot 100\% = 0,21\%,$$

– Сімпсона (парабол)

$$\varepsilon_c = \left| \frac{I - I_c}{I} \right| \cdot 100\% = \left| \frac{0,785398 - 0,785392}{0,785398} \right| \cdot 100\% = 7,63 \cdot 10^{-4}\%.$$

Найточніший результат розв'язання дозволяє отримати використання методу Сімпсона (парабол) з тим, що крок обчислення значно більший, ніж у формули трапецій для обчислення інтеграла.

### 6.3 Формула Чебишова

Якщо у виразі (6.9) виконати заміну  $x_i = \frac{a+b}{2} + \frac{b-a}{2} t_i$ , тоді відповідний вираз буде зведений до такого вигляду:

$$\int_{-1}^1 f(t) dt = \sum_{i=1}^n A_i f(t_i). \quad (6.14)$$

Під час виведення формули Чебишова використовуються такі умови: коефіцієнти  $A_i$  дорівнюють один одному; квадратурна формула (6.14) має високий ступінь точності для всіх поліномів до  $n$ -го степеня включно.

Враховуючи, що  $A_1=A_2=\dots=A_n=A$  і  $f(t)=1$ , то  $\sum_{i=1}^n A_i = nA = 2$ , звідки  $A=2/n$ .

За цих умов формула (6.14) буде мати такий вигляд:

$$\int_{-1}^1 f(t) dt = \frac{2}{n} \sum_{i=1}^n f(t_i). \quad (6.15)$$

Для визначення  $t_i$  використовується друга умова, згідно з якою потрібно, щоб формула (6.15) мала високий ступінь точності обчислення для функції вигляду:

$$f(t) = t^k \quad (k=1, 2, \dots, n). \quad (6.16)$$

Після підстановки цих функцій в (6.15) отримаємо систему рівнянь:

$$\begin{cases} t_1 + t_2 + \dots + t_n = 0; \\ t_1^2 + t_2^2 + \dots + t_n^2 = \frac{n}{3}; \\ \dots\dots\dots; \\ t_1^n + t_2^n + \dots + t_n^n = \frac{n[1 - (-1)^{n+1}]}{2(n+1)}. \end{cases} \quad (6.17)$$

Система рівнянь (6.17) має розв'язок за  $n < 8$  та  $n = 9$ , що і накладає обмеження в точності обчислення, як недолік використання формули Чебишова. Значення абсцис  $t_i$  у формулі Чебишова для різних значень  $n$  наведено в таблиці 6.2.

Для довільного інтервалу  $[a; b]$  формула (6.15) набуває вигляду:

$$I = \frac{b-a}{n} \sum_{i=1}^n f(x_i), \quad (6.18)$$

де  $x_i = \frac{a+b}{2} + \frac{b-a}{2} t_i$ .

Таблиця 6.2 – Значення абсцис  $t_i$  у формулі Чебишова

$n$	$i$	$t_i$	$n$	$i$	$t_i$	
2	1; 2	$\mp 0,5773500$	6	1; 6	$\mp 0,866247$	
	3	1; 3		$\mp 0,7071070$	2; 5	$\mp 0,422519$
2		0,0		3; 4	$\mp 0,266635$	
4	1; 4	$\mp 0,7946540$		7	1; 7	$\mp 0,883862$
	2; 3	$\mp 0,1875920$			2; 6	$\mp 0,529657$
5	1; 5	$\mp 0,8324980$			3; 5	$\mp 0,323912$
	2; 4	$\mp 0,3745413$	4		0,0	
	3	0,0				

Похибка обчислень за методом Чебишова:

$$\Delta = \int_a^b \frac{\left(x - \frac{a+b}{2}\right)^{n+1}}{(n+1)!} f^{(n+1)}(x) dx = \frac{b-a}{n(n+1)!} \sum_{i=1}^n \left(x_i - \frac{a+b}{2}\right)^{n+1} f^{(n+1)}(x). \quad (6.19)$$

**Приклад 6.3.** Визначити значення інтеграла  $I = \int_0^1 \frac{dx}{1+x^2}$  чисельним методом Чебишова (порядок методу –  $n = 3$ ).

*Розв'язання:*

Для порядку  $n = 3$  із таблиці 6.2 отримасмо значення абсцис:  $t_1 = -0,707107$ ;  $t_2 = 0$ ;  $t_3 = 0,707107$ .

Якщо отримані значення абсцис підставити у формулу 6.19, то отримаємо:

$$I_v = \frac{b-a}{n} \sum_{i=1}^n f(x_i) = \frac{b-a}{n} [f(x_1) + f(x_2) + f(x_3)] = \frac{1,0-0}{3} [0,9790 + 0,80 + 0,57852] = 0,78584,$$

$$\begin{aligned} \text{де } x_1 &= \frac{a+b}{2} + \frac{b-a}{2} \cdot t_1 = \frac{0+1,0}{2} + \frac{1,0-0}{2} \cdot (-0,707107) = 0,14645; \quad x_2 = \frac{a+b}{2} + \\ &+ \frac{b-a}{2} \cdot t_2 = \frac{0+1,0}{2} + \frac{1,0-0}{2} \cdot 0 = 0,5; \quad x_3 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_3 = \frac{0+1,0}{2} + \frac{1,0-0}{2} \times \\ &\times 0,707107 = 0,85355; \quad f(x_1) = \frac{1}{1+x_1^2} = \frac{1}{1,0+0,14645^2} = 0,9790; \quad f(x_2) = \\ &= \frac{1}{1+x_2^2} = \frac{1}{1+0,5^2} = 0,8; \quad f(x_3) = \frac{1}{1+x_3^2} = \frac{1}{1+0,85355^2} = 0,57852. \end{aligned}$$

**Приклад 6.4.** Визначити значення інтеграла  $I = \int_0^1 \frac{dx}{1+x^2}$  методом криволінійних трапецій із кроком обчислення  $h = 0,5$  (порядок чисельного методу Чебишова –  $n = 3$ ).

*Розв'язання:*

Для кроку обчислення  $h = 0,5$  заданий інтервал  $x \in [0; 1,0]$  розбивається на два однакових проміжки, для яких відповідно  $I = I_1 + I_2$ .

Значення інтегралів  $I_1$  та  $I_2$  для кожного із проміжків визначаються за формулою Чебишова, тому з таблиці 6.2 отримасмо значення абсцис:  $t_1 = -0,707107$ ;  $t_2 = 0$ ;  $t_3 = 0,707107$ .

Значення інтеграла  $I_1$  для інтервалу  $[0; 0,5]$ :

$$I_1 = \frac{b-a}{n} \sum_{i=1}^n f(x_i) = \frac{b-a}{n} [f(x_1) + f(x_2) + f(x_3)] = \frac{0,5-0}{3} [0,99467 + 0,94118 + 0,84592] = 0,46363,$$

$$\begin{aligned} \text{де } x_1 &= \frac{a+b}{2} + \frac{b-a}{2} \cdot t_1 = \frac{0+0,5}{2} + \frac{0,5-0}{2} \cdot (-0,707107) = 0,07322; x_2 = \frac{a+b}{2} \\ &+ \frac{b-a}{2} \cdot t_2 = \frac{0+0,5}{2} + \frac{0,5-0}{2} \cdot 0 = 0,250; \quad x_3 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_3 = \frac{0+0,5}{2} + \\ &+ \frac{0,5-0}{2} \cdot (0,707107) = 0,42678; \quad f(x_1) = \frac{1}{1+x_1^2} = \frac{1}{1+0,07322^2} = 0,99467; \\ f(x_2) &= \frac{1}{1+x_2^2} = \frac{1}{1+0,25^2} = 0,94118; \quad f(x_3) = \frac{1}{1+x_3^2} = \frac{1}{1+0,42678^2} = 0,84592. \end{aligned}$$

Значення інтеграла  $I_2$  для інтервалу  $[0,5; 1,0]$ :

$$I_2 = \frac{b-a}{n} \sum_{i=1}^n f(x_i) = \frac{b-a}{n} [f(x_4) + f(x_5) + f(x_6)] = \frac{1,0-0,5}{3} [0,75268 + 0,64 + 0,53795] = 0,32177,$$

$$\begin{aligned} \text{де } x_4 &= \frac{a+b}{2} + \frac{b-a}{2} \cdot t_1 = \frac{0,5+1}{2} + \frac{1-0,5}{2} \cdot (-0,707107) = 0,57322; \quad x_5 = \frac{a+b}{2} \\ &+ \frac{b-a}{2} \cdot t_2 = \frac{0,5+1}{2} + \frac{1-0,5}{2} \cdot 0 = 0,750; \quad x_6 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_3 = \frac{0+0,5}{2} + \\ &+ \frac{0,5-0}{2} \cdot (0,707107) = 0,92678; \quad f(x_4) = \frac{1}{1+x_4^2} = \frac{1}{1+0,57322^2} = 0,75268; \\ f(x_5) &= \frac{1}{1+x_5^2} = \frac{1}{1+0,75^2} = 0,640; \quad f(x_6) = \frac{1}{1+x_6^2} = \frac{1}{1+0,92678^2} = 0,53795. \end{aligned}$$

Тоді повне значення інтеграла:

$$I_{\text{км}} = \int_0^1 \frac{dx}{1+x^2} = I_1 + I_2 = 0,46363 + 0,32177 = 0,78540.$$

Відносна похибка обчислення шляхом порівняння між значеннями аналітичного розв'язку інтеграла (див. прикл. 6.1) і формулою Чебишова:

– для усього інтервалу (див. прикл. 6.3)

$$\varepsilon_{ni} = \left| \frac{I - I_q}{I} \right| \cdot 100\% = \left| \frac{0,785398 - 0,785840}{0,785398} \right| \cdot 100\% = 0,056\%,$$

– методом криволінійних трапецій (див. прикл. 6.4)

$$\varepsilon_{\text{км}} = \left| \frac{I - I_{\text{км}}}{I} \right| \cdot 100\% = \left| \frac{0,785398 - 0,785400}{0,785398} \right| \cdot 100\% = 2,55 \cdot 10^{-4} \%.$$

Використання формули Чебишова, порівняно з формулами трапецій і Сімпсона, є найбільш продуктивним, оскільки дозволяє розглядати увесь проміжок інтегрування загалом, водночас забезпечуючи більш високу точність обчислення. З метою подальшого підвищення точності розрахунку, на основі формули Чебишова, доцільно зменшувати інтервали обчислень у рамках загального інтервалу інтегрування.

## 6.4 Формула Гаусса

Формула Гаусса називається формулою найвищої алгебраїчної точності. Для формули виду (6.14) найвища точність може бути досягнута для поліномів степеня  $(2n-1)$ , які визначаються  $2n$  постійними  $t_i$  і  $A_i$  ( $i = 1, 2, \dots, n$ )

Для забезпечення цієї умови необхідно і достатньо, щоб вона виконувалась для функцій типу:

$$f(t) = t^k \quad (k = 0, 1, \dots, 2n - 1).$$

Враховуючи, що функція  $f(t)$  може бути апроксимована поліномами степеня  $(2n-1)$ , а саме  $f(t) = \sum_{k=0}^{2n-1} c_k t^k$ , можна отримати:

$$\int_{-1}^1 f(t) dt = \sum_{k=0}^{2n-1} C_k \int_{-1}^1 t^k dt = \sum_{k=0}^{2n-1} C_k \sum_{i=1}^n A_i t_i^k = \sum_{i=1}^n A_i \sum_{k=0}^{2n-1} C_k t_i^k A_i f(t_i).$$

Завдання полягає у визначенні коефіцієнтів  $A_i$  і абсцис точок  $t_i$ . Для визначення цих сталих необхідно розглянути виконання формули (6.14) для функцій вигляду  $f(t) = t^k$  ( $k = 0, 1, \dots, 2n-1$ ).

Враховуючи, що

$$\int_{-1}^1 t^k dt = \begin{cases} 2 / (k + 1) \\ 0 \end{cases},$$

отримуємо систему рівнянь:

$$\begin{cases} \sum_{i=1}^n A_i = 2; \\ \sum_{i=1}^n A_i t_i = 0; \\ \sum_{i=1}^n A_i t_i^2 = 1; \\ \sum_{i=1}^n A_i t_i^{2n-2} = \frac{2}{2n-1}; \\ \sum_{i=1}^n A_i t_i^{2n-1} = 0. \end{cases} \quad (6.20)$$

Система рівнянь (6.20) нелінійна, розв'язання її пов'язано зі значними обчислювальними труднощами. Але якщо використовувати систему для поліномів вигляду:

$$f(t) = t^k P_n(t) \quad (k = 0, 1, \dots, n-1), \quad (6.21)$$

де  $P_n(t)$  – поліном Лежандра, то її можна звести до лінійної системи відносно коефіцієнтів  $A_i$  із заданими точками  $t_i$ .

Поліноми Лежандра називаються поліномами виду

$$P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} [(x^2 - 1)^n],$$

які мають такі основні властивості:

- 1)  $P_n(1) = 1$ ,  $P_n(-1) = (-1)^n$  для будь-якого цілого  $n$ ;
- 2) ортогональність

$$\int_{-1}^1 P_n(x) Q_k(x) dx = 0, \quad (6.22)$$

де  $Q_k(x)$  – будь-який поліном степеня  $k < n$ ;

- 3) наявність  $n$  дійсних коренів на інтервалі  $[-1; 1]$ .

Оскільки степені поліномів у співвідношенні (6.21) не перевищують значення  $2n-1$ , то має виконуватися система (6.20), і формула (6.14) набуває такого вигляду:

$$\int_{-1}^1 t^k P_n(t) dt = \sum_{i=1}^n A_i t_i^k P_n(t_i). \quad (6.23)$$

Як результат властивості ортогональності (6.22) – ліва частина виразу (6.23) дорівнює нулю, тоді:

$$\sum_{i=1}^n A_i t_i^k P_n(t_i) = 0, \quad (6.24)$$

що завжди забезпечується за будь-яких значень  $A_i$  в точках  $t_i$ , які відповідають кореням відповідних поліномів Лежандра.

Підставивши ці значення  $t_i$  в систему (6.20) і враховуючи перші  $n$  рівнянь, можна визначити коефіцієнти  $A_i$ .

Формула (6.14), де  $t_i$  – нулі полінома Лежандра  $P_n(t)$ , а  $A_i$  ( $i = 1, 2, \dots, n$ ) визначаються із системи (6.20), називається формулою Гаусса.

У таблиці 6.3 наведено значення  $t_i$  і  $A_i$  у формулі Гаусса для різних  $n$  від 1 до 8.

Для довільного інтервалу  $[a; b]$  формула для методу Гаусса набуває вигляду:

$$I = \frac{b-a}{2} \sum_{i=1}^n A_i f(x_i), \quad (6.25)$$

де  $x_i = \frac{a+b}{2} + \frac{b-a}{2} t_i$ .

Оцінка похибки формули Гаусса з  $n$  вузлами визначається із співвідношення:

$$\Delta \leq \frac{(b-a)^{2n+1} (n!)^4 M_{2n}}{[(2n)!]^3 (2n+1)}, \quad (6.26)$$

де  $M_{2n}$  – максимальне значення  $2n$ -ої похідної на проміжку  $[a; b]$ .



Таблиця 6.3 – Елементи формули Гаусса

$n$	$i$	$t_i$	$A_i$
1	1	0,0	2,0
2	1; 2	$\mp 0,57735027$	1,0
3	1; 3	$\mp 0,77459667$	$\frac{5}{9}=0,55555556$
	2	0,0	$\frac{8}{9}=0,88888889$
4	1; 4	$\mp 0,86113631$	0,34785484
	2; 3	$\mp 0,33998104$	0,65214516
5	1; 5	$\mp 0,90617985$	0,23692689
	2; 4	$\mp 0,53846931$	0,47862867
	3	0,0	0,56888889
6	1; 6	$\mp 0,93246951$	0,17132450
	2; 5	$\mp 0,66120939$	0,36076158
	3; 4	$\mp 0,238619119$	0,46791394
7	1; 7	$\mp 0,94910791$	0,12948496
	2; 6	$\mp 0,74153119$	0,27970540
	3; 5	$\mp 0,40584515$	0,38183006
	4	0,0	0,41795918
8	1; 8	$\mp 0,96028986$	0,10122854
	2; 7	$\mp 0,79666648$	0,22238104
	3; 6	$\mp 0,52553142$	0,31370664
	4; 5	$\mp 0,18343464$	0,36268378

**Приклад 6.5.** Визначити значення інтеграла  $I = \int_0^1 (1+x^2)dx$  методом Гаусса (порядок методу –  $n = 3$ ).

*Розв'язання:*

Для порядку  $n = 3$  із таблиці 6.3 обираємо значення абсцис:  $t_1 = -0,77459667$ ;  $A_1 = 0,55555556$ ;  $t_2 = 0$ ;  $A_2 = 0,88888889$ ;  $t_3 = 0,77459667$ ;  $A_3 = 0,55555556$ .

Якщо отримані значення абсцис підставити у формулу (6.25), матимемо:

$$I_1 = \frac{b-a}{2} (A_1 f(x_1) + A_2 f(x_2) + A_3 f(x_3)) = \frac{1,0-0}{2} (1,012702 \cdot 0,55555556 + 1,250 \cdot 0,88888889 + 1,787298 \cdot 0,55555556) = 1,333333,$$

$$\text{де } x_1 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_1 = \frac{0+1,0}{2} + \frac{1,0-0}{2} \cdot (-0,77459667) = 0,112702;$$

$$x_2 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_2 = \frac{0+1,0}{2} + \frac{1,0-0}{2} \cdot 0 = 0,50;$$

$$x_3 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_3 = \frac{0+1,0}{2} + \frac{1,0-0}{2} \cdot (0,77459667) = 0,887298;$$

$$f(x_1) = 1 + x_1^2 = 1 + 0,112702^2 = 1,012702; \quad f(x_2) = 1 + x_2^2 = 1 + 0,5^2 = 1,250;$$

$$f(x_3) = 1 + x_3^2 = 1 + 0,887298^2 = 1,787298.$$

Точне значення інтеграла на основі аналітичного розв'язку:

$$I = \int_0^1 (1+x^2) dx = \frac{1}{3} x(x^2+3) \Big|_0^1 = \frac{4}{3} = 1,333333. \quad (6.27)$$

**Приклад 6.6.** Визначити значення інтеграла  $I = \int_0^1 (1+x^2) dx$  методом криволінійних трапецій із кроком обчислення  $h = 0,5$  (порядок чисельного методу Гаусса –  $n = 3$ ).

*Розв'язання:*

Для кроку обчислення  $h = 0.5$  заданий інтервал  $x \in [0; 1]$  розбивається на два однакових проміжки, для яких відповідно  $I = I_1 + I_2$ .

Значення інтегралів  $I_1$  та  $I_2$  для кожного із проміжків визначаються за формулою Чебишова, тому з таблиці 6.3 отримаємо значення абсцис:  $t_1 = -0,77459667$ ;  $A_1 = 0,55555556$ ;  $t_2 = 0$ ;  $A_2 = 0,88888889$ ;  $t_3 = 0,77459667$ ;  $A_3 = 0,55555556$ .

Значення інтеграла  $I_1$  для інтервалу  $[0; 0,5]$ :

$$I_1 = \frac{b-a}{2} [A_1 f(x_1) + A_2 f(x_2) + A_3 f(x_3)] = \frac{0,5-0}{2} (1,003175 \cdot 0,55555556 + 1,0625 \cdot 0,88888889 + 0,196825 \cdot 0,55555556) = 0,541667,$$

$$\text{де } x_1 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_1 = \frac{0+0,5}{2} + \frac{0,5-0}{2} \cdot (-0,77459667) = 0,056351;$$

$$x_2 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_2 = \frac{0+0,5}{2} + \frac{0,5-0}{2} \cdot 0 = 0,250; \quad x_3 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_3 =$$

$$= \frac{0+0,5}{2} + \frac{0,5-0}{2} \cdot (0,77459667) = 0,443649; \quad f(x_1) = 1 + x_1^2 = 1 + 0,056351^2 =$$

$$= 1,003175; \quad f(x_2) = 1 + x_2^2 = 1 + 0,25^2 = 1,06250; \quad f(x_3) = 1 + x_3^2 = 1 +$$

$$+ 0,443649^2 = 1,196825.$$

Значення інтеграла  $I_2$  для інтервалу  $[0,5; 1,0]$ :

$$I_2 = \frac{b-a}{2} (A_1 f(x_4) + A_2 f(x_5) + A_3 f(x_6)) = \frac{0,5-0}{2} (1,309526 \cdot 0,55555556 + 1,5625 \cdot 0,88888889 + 1,890474 \cdot 0,55555556) = 0,791667,$$

$$\text{де } x_1 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_1 = \frac{0,5+1,0}{2} + \frac{1,0-0,5}{2} \cdot (-0,77459667) = 0,556351;$$

$$x_2 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_2 = \frac{0,5+1,0}{2} + \frac{1,0-0,5}{2} \cdot 0 = 0,750;$$

$$x_3 = \frac{a+b}{2} + \frac{b-a}{2} \cdot t_3 = \frac{0,5+1,0}{2} + \frac{1,0-0,5}{2} \cdot (0,77459667) = 0,943649;$$

$$f(x_4) = 1 + x_4^2 = 1 + 0,556351^2 = 1,309526; \quad f(x_5) = 1 + x_5^2 = 1 + 0,750^2 = 1,5625$$

;

$$f(x_6) = 1 + x_6^2 = 1 + 0,943649^2 = 1,890474.$$

Тоді повне значення інтеграла:

$$I = \int_0^1 (1+x^2) dx = I_1 + I_2 = 0,541677 + 0,791667 = 1,333333.$$

Порівнюючи результати чисельного й аналітичного розв'язання, можна відзначити високу точність обчислення значення інтеграла.

**Приклад 6.7.** Визначити значення інтеграла  $I = \int_0^1 \frac{dx}{1+x^2}$  методом Гаусса (порядок методу –  $n = 5$ ).

*Розв'язання:*

Зробимо заміну змінної  $x = \frac{a+b}{2} + \frac{b-a}{2} \xi = \frac{1}{2} + \frac{1}{2} \xi$ , де  $a=0$  і  $b=1,0$ .

Також  $dx = \frac{d\left[\frac{1}{2} + \frac{1}{2}\xi\right]}{d\xi} = \frac{1}{2}$ . На основі формули (6.25) нове значення інтервалу інтегрування для  $x \in [x_1; x_2] = [0; 1]$ :  $\xi_1 = 2t_1 - 1 = 2 \cdot 0 - 1,0 = -1,0$ ;  $\xi_2 = 2t_2 - 1 = 2 \cdot 1,0 - 1,0 = 1,0$ .

Також функція  $f(x) = \frac{1}{1+x^2}$  після підстановки набуде такого вигляду:

$$\varphi(\xi) = \frac{4}{4 + (\xi + 1)^2}.$$

Після підстановки і заміни усіх складових, новий інтегральний вираз матиме такий вигляд:

$$I = 2 \int_{-1}^1 \frac{d\xi}{4 + (\xi + 1)^2}.$$

За формулою Гаусса (6.25):

$$I = 2 \int_{-1}^1 \varphi(\xi) d\xi = \frac{\xi_2 - \xi_1}{2} \sum_{i=1}^n A_i f(\varepsilon_i) = 2 \left( \frac{\xi_2 - \xi_1}{2} \right) [A_1 \varphi(\varepsilon_1) + A_2 \varphi(\varepsilon_2) + A_3 \varphi(\varepsilon_3) + A_4 \varphi(\varepsilon_4) + A_5 \varphi(\varepsilon_5)] = 2 \left( \frac{1,0 - (-1,0)}{2} \right) [0,236926885 \cdot \varphi(-0,9061179846) + 0,478628670 \cdot f(-0,538469310) + 0,568888889 \cdot f(0) + 0,478628670 \times f(0,538469310) + 0,236926885 \cdot f(0,906179846)] = 0,78539816,$$

де  $\varepsilon_i = \frac{\xi_1 + \xi_2}{2} + \frac{\xi_2 - \xi_1}{2} t_i = \frac{1,0 + (-1,0)}{2} + \frac{1,0 - (-1,0)}{2} t_i = t_i$  й відповідно значення параметрів  $A_i$  і  $f(t_i)$  подано в таблиці 6.4.

Таблиця 6.4 – Результати обчислення інтеграла функції

$i$	$t_i$	$f(t_i)$	$A_i$
1	-0,9061179846	0,24945107	0,236926885
2	-0,538469310	0,23735995	0,478628670
3	0,0	0,20000000	0,568888889
4	0,538469310	0,15706211	0,478628670
5	0,906179846	0,13100114	0,236926885

Порівнюючи результати чисельного й аналітичного (див. прикл. 6.1) розв'язання, можна відмітити високу точність обчислення значення інтеграла.

## 6.5 Алгоритми застосування чисельних методів

Послідовність застосування формул Ньютона – Котеса:

1. Вибір формули і визначення (див. табл. 6.1) коефіцієнтів  $H_i$ .
2. Складання алгоритму та програми, причому:
  - у випадку задання дискретних значень  $y_i = f(x_i)$  через крок  $h$  ці значення підставляються у вибрану формулу, наприклад, у (6.12) або (6.13);
  - у випадку задання функції  $y = f(x)$  значення  $y_i = f(x_i)$  визначається, причому  $x_i = x_0 + ih = a + ih$  ( $a \leq x \leq b$ ).
3. Оцінення похибки.

Алгоритм розв'язання інтеграла методом трапецій подано на рисунку 6.5.

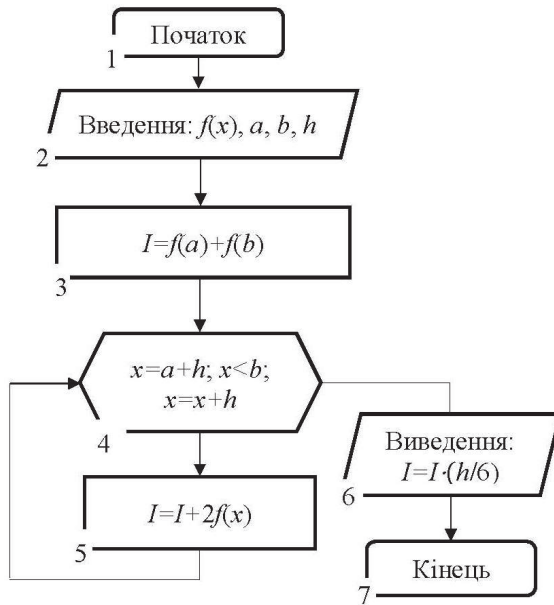


Рисунок 6.5 – Схема алгоритму розв’язання інтегралів методом трапецій

Розглянемо реалізацію методу трапецій мовою програмування C++:

```

#include <stdio.h>
#include <conio.h>
#include <math.h>

FILE *fp;
const int n_h = 2;

float f(float x)
{ return (sin(x)/(x*x + 1)); }

float Metod_Trapets (float a, float b, float h)
{
    float I, x = a + h;
    fprintf (fp, «\n--- Metod trapets - h = %.3f ---\n», h);
    I = f(a) + f(b);
    for (x; x<b; x+=h)
        I = I + 2*f(x);

    I = h*I/2;
    fprintf (fp, «I = %9.7f\n», I);
    return 0;
}

int main()

```

```

{
    float a = 0, b = 1;
    float h[n_h] = {0.1, 0.05};
    if ( (fp = fopen(«Data.txt», «w»)) == NULL)
    {
        printf(«Error opening file\n»);
        return 0;
    }
    for (int i=0; i<n_h; i++)
    {
        Metod_Trapets (a, b, h[i]);
    }
    printf(«End!»);
    getch();
    fclose(fp);
    return 0;
}.

```

Під час розв'язання задачі методом Сімпсона за непарної кількості інтервалів пропонується додатково розбивати кожен проміжок навпіл. Алгоритм методу Сімпсона за непарної кількості інтервалів подано на рисунку 6.6.

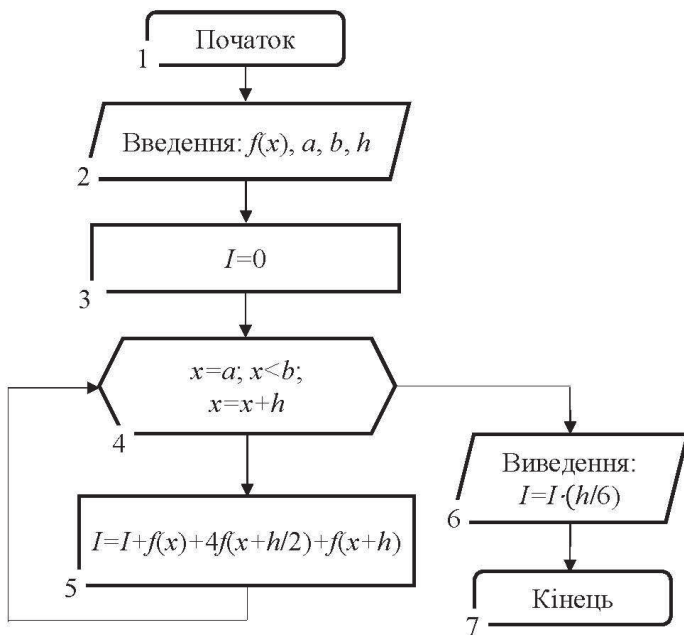


Рисунок 6.6 – Схема алгоритму розв'язання інтегралів методом Сімпсона для непарної кількості інтервалів

Послідовність застосування методу Гаусса:

1. Вибір порядку методу і визначення (див. табл. 6.3) коефіцієнтів  $A_i$  і значень  $t_i$  ( $-1 \leq t_i \leq 1$ ).
2. Розбиття інтервалу  $a \leq x \leq b$  на  $l$  відрізків (рис. 6.7).
3. Визначення значень інтеграла для кожного інтервалу ( $j = 1, 2, \dots, l$ )

$$I = \sum_{j=1}^l I_j.$$

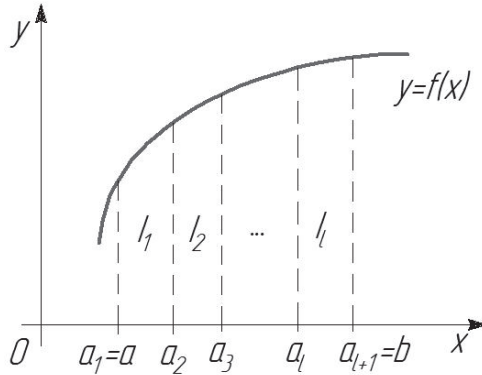


Рисунок 6.7 – Схема поділу інтервалу інтегрування на відрізки у методі Гаусса

У такому випадку значення абсцис  $x_i$  усередині кожного інтервалу  $j$  визначаються за формулою:

$$x_i = \frac{a_{j+1} + a_i}{2} + \frac{a_{j+1} - a_i}{2} t_i, \quad (6.28)$$

де  $a_{j+1} = a_j + h$ , причому  $a_1 = a$ ,  $a_{l+1} = b$ ,  $h = \frac{b-a}{n}$  ( $j = 1, 2, \dots, l$ ).

Значення інтеграла  $I_j$  визначаються за формулою:

$$I_j = \int_{a_j}^{a_{j+1}} f(x) dx = \frac{(a_{j+1} - a_j)}{2} \sum_{i=1}^n A_i f(x_i). \quad (6.29)$$

Алгоритм методу Гаусса подано на рисунку 6.8, б).

Розглянемо реалізацію функції методу Гаусса мовою програмування C++:

```

const int n_G = 4; //порядок методу для методу Гаусса
float t_G[n_G]= {-0.86113631, -0.33998104, 0.33998104, 0.86113631}; //
    коефіцієнти t [i] для методу Гаусса n=4
float A_G[n_G]= {0.34785484, 0.65214516, 0.65214516, 0.34785484}; //
    коефіцієнти A[i] для методу Гаусса n=4

float Method_Gaussa (float a, float b)
{
    float I;
    fprintf (fp, "\n--- Metod Gaussa_simple \n");
    float F = 0;
    for (int i=0; i<n_G; i++)
    {
        float x = (a+b)/2 + (b-a)*t_G[i]/2;
        F = F + A_G[i]*f(x);
    }
    I = (b-a)*F/2;
    fprintf (fp, "I = %9.7f\n", I);
    return 0;
}

```

До головної функції відповідно потрібно додати:  
 method\_Gaussa (a, b);

У методі Чебишова послідовність дій аналогічна методу Гаусса, але в пункті 1 коефіцієнти  $t_i$  беруться із таблиці 6.2, а в пункті 3 для визначення  $j$ -го інтеграла використовується формула:

$$I_j = \int_{a_j}^{a_{j+1}} f(x)dx = \frac{(a_{j+1} - a_j)}{n} \sum_{i=1}^n f(x_i), \quad (6.30)$$

де  $x_i$  оцінюється аналогічним способом, що і в методі Гаусса, за формулою (6.25).

Алгоритм методу Чебишова подано на рисунку 6.8, а).

Розглянемо реалізацію функції методу Чебишова мовою програмування C++:

```

const int n = 4; //порядок методу для методу чебишева
float t[n]= {-0.794654, -0.187592, 0.187592, 0.794654};

float Method_Chebisheva (float a, float b)
{
    float I;
    fprintf (fp, "\n--- Metod Chebisheva_simple \n");
    float F = 0;
    for (int i=0; i<n; i++)
    {
        float x = (a+b)/2 + (b-a)*t[i]/2;
        F = F + f(x);
    }
    I = (b-a)*F/n;
    fprintf (fp, "I = %9.7f\n", I);
    return 0;
}

```

До головної функції потрібно додати, відповідно:  
 method\_Chebisheva (a, b);



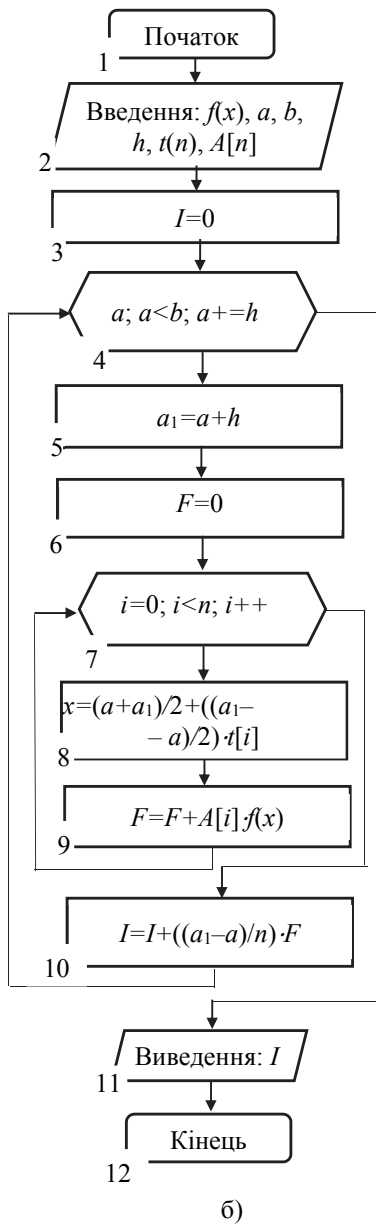
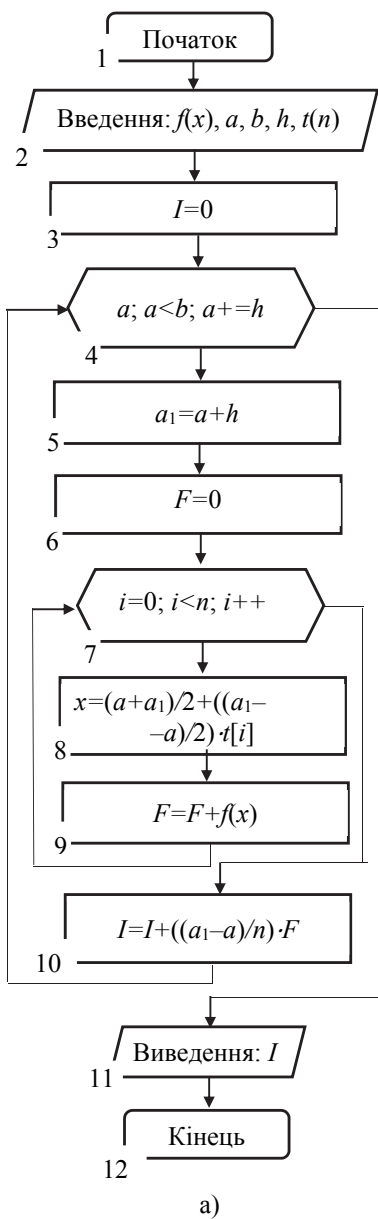


Рисунок 6.8 – Схеми алгоритмів розв’язання задач методами:  
а) Чебишова; б) Гаусса

## 6.6 Метод Монте-Карло

Метод чисельного інтегрування Монте-Карло – є найвідомішим застосуванням статистичного моделювання для розв'язання прикладних математичних задач.

Якщо з послідовністю випадкових чисел  $\{x_i\} \in X$  із законом розподілу ймовірностей  $f_x(x)$  провести функціональне перетворення  $y_i = \phi(x_i)$ , тоді математичне сподівання отриманої послідовності випадкових чисел  $\{y_i\} \in Y$ :

$$m_y = \int_{-\infty}^{\infty} \phi(x) f_x(x) dx \quad (6.31)$$

за обсягу вибірки ( $n > 1000$ ) з достатньо високою точністю ( $\Delta_{\text{вiдн}} < 0,1$ ) може бути оцінено за формулою:

$$m_y = \frac{1}{n} \sum_{i=1}^n y_i. \quad (6.32)$$

Якщо у вирази (6.31) і (6.32) ввести функцію індикатора області  $[a; b]$ :

$$\mathbf{1}[a, b, x] = \begin{cases} 1, & a \leq x \leq b; \\ 0, & x < a, x > b, \end{cases}$$

і обрати  $\phi(x) = \frac{f(x)}{f_x(x)}$ , де  $f(x)$  – підінтегральна функція, а значення  $a$  і  $b$  – межі інтегрування із формули (6.14), тоді кінцевий вираз буде мати вигляд:

$$I = m_y = \int_a^b f(x) dx = \frac{1}{n} \sum_{i=1}^n \frac{f(x_i)}{f_x(x_i)} \mathbf{1}[a, b, x_i].$$

Схему алгоритму чисельного інтегрування методом Монте-Карло подано на рисунку 6.9.

Похибка методу Монте-Карло визначається похибкою генерації псевдовипадкової послідовності чисел, згенерованих за допомогою комп'ютерної системи, та обсягом вибірки. Вона може бути оцінена із такого співвідношення:

$$\Delta = \frac{1}{2\sqrt{n(1-P)}}. \quad (6.33)$$

де  $P$  – гарантована ймовірність влучання похибки в інтервал  $[-\Delta; +\Delta]$ .

Кількість випробувань не залежить від розмірності інтеграла  $I$ , тому метод Монте-Карло вигідно використовувати для обчислення багатократних інтегралів, де застосування інших методів чисельного інтегрування трудомістке (наприклад, обчислення десятикратного інтеграла в одиничному об'ємі із кроком  $h = 0,1$  потребує обчислення суми приблизно кількості  $10^{10}$  складових).

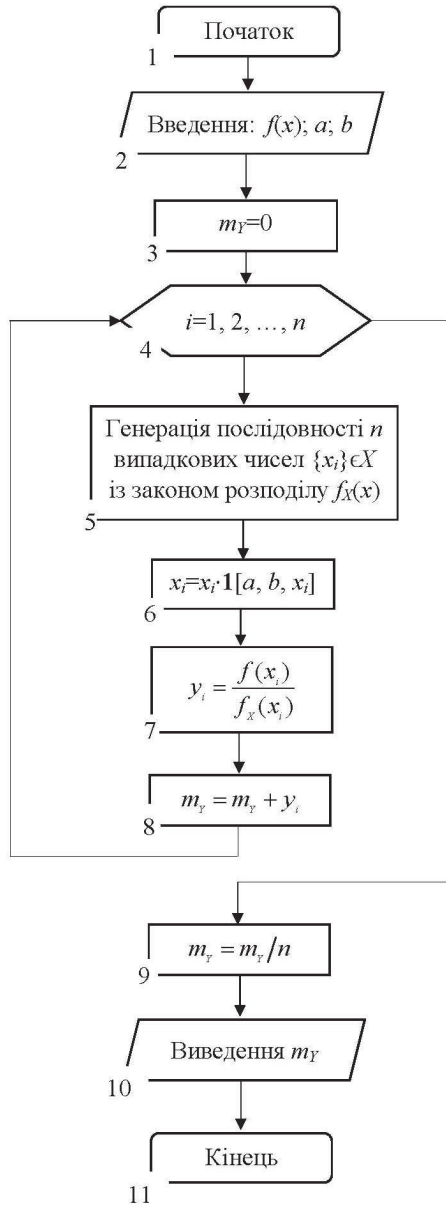


Рисунок 6.9 – Схема алгоритму метода Монте-Карло

## 6.7 Оцінення похибки за чисельного інтегрування

У випадку, коли підінтегральна функція задана аналітично, може бути поставлена задача про визначення інтеграла із наперед заданою точністю. Оскільки точність розглянутих вище квадратурних формул залежить від кроку обчислень  $h$ , тоді точність результату можна підвищити, зменшуючи крок. Наприклад, за допомогою поділу обчислювального кроку навпіл.

Для визначення значення інтеграла із заданою точністю  $\varepsilon$  необхідно вибрати відповідну квадратурну формулу й початковий крок  $h=h_0$ . Після чого обчислити інтеграл з кроком  $h/2$ . Зменшувати крок обчислення шляхом половинного ділення необхідно до тих пір, поки не виконається умова:

$$|I_{h/2} - I_h| < \varepsilon.$$

За виконання цієї умови значення інтеграла дорівнює  $I_{h/2}$ .

Також можна користуватися методом Рунге (для того, щоб не вносити додаткові похибки в обчислення, необхідно вибрати близькі за значенням кроки в першому і другому розрахунках):

$$c = \frac{I_{h_1} - I_{h_2}}{h_2^{p+1} - h_1^{p+1}},$$

$$\text{де } I^* = I_{h_1} + ch_1^{p+1} = I_{h_2} + ch_2^{p+1}.$$

Також похибки обчислення інтегралів чисельними методами можна розрахувати за такими формулами:

– метод трапецій

$$\Delta = \frac{nh^3}{12} M_2,$$

де  $M_2$  – максимальне значення другої похідної  $f(x)$  в обчислювальній області  $x \in [a; b]$ ;

– метод Сімпсона

$$\Delta = \frac{nh^5}{180} M_4,$$

де  $M_4$  – максимальне значення четвертої похідної  $f(x)$  в обчислювальній області  $x \in [a; b]$ ;

– метод Чебишова

$$\Delta = \frac{b-a}{n(n+1)!} \sum_{i=1}^n \left( x_i - \frac{a+b}{2} \right)^{n+1} f^{(n+1)}(x);$$

– метод Гаусса

$$\Delta \leq \frac{(b-a)^{2n+1} (n!)^4 M_{2n}}{[(2n)!]^3 (2n+1)};$$

– метод прямокутників (ліві, праві)

$$\Delta = \frac{nh^2}{2} M_1,$$

де  $M_1$  – максимальне значення першої похідної  $f(x)$  в обчислювальній області  $x \in [a; b]$ ;

– метод прямокутників (середні)

$$\Delta = \frac{nh^3}{24} M_2.$$

## 6.8 Чисельне диференціювання

Задача полягає в тому, щоб для заданої диференційованої функції  $f(x)$ , яка задана табличними даними або аналітичним виразом, знайти похідну в точці  $x_0$ .

### 6.8.1 Чисельне диференціювання аналітично заданих функцій

Наближене диференціювання аналітично заданих функцій необхідне під час розробки універсальної процедури пошуку похідної для великої кількості різних функцій, або у випадку, коли аналітичний вигляд похідної функції надто громіздкий і призводить до втрати точності.

В основі чисельного диференціювання (*numerical differentiation*) аналітично заданих функцій покладено визначення похідної:

$$f'(x_0) = \lim_{h \rightarrow 0} \frac{f(x_0 + h) - f(x_0)}{h}.$$

Оскільки невідомо, яке значення  $h$  взяти, то необхідно побудувати послідовність  $\{h_k\}$  такою, щоб  $h_k \rightarrow 0$  (наприклад,  $h_k = 0,5^k$ ) й, відповідно, формується послідовність  $\{D_k\}$ , де:

$$D_k = \frac{f(x + h_k) - f(x)}{h_k} \quad (k=1, 2, \dots, n). \quad (6.34)$$

Розрахунок елементів послідовності виконується до тих пір, поки виконується умова:

$$|D_{n+1} - D_n| < |D_n - D_{n-1}|.$$

Якщо відома точність  $\varepsilon$ , з якою необхідно визначити похідну, то умова завершення обчислення похідної цим методом із порядком точності  $h$ :

$$|D_{n+1} - D_n| < \varepsilon.$$

Нехай  $f \in C^3[a; b]$ , тоді порядок точності попереднього методу можна підвищити, використовуючи замість формули (6.34) інші вирази.

Формула другого порядку точності  $O(h^2)$  для обчислення похідної:

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}. \quad (6.35)$$

Формулу (6.35) можна отримати, розклавши функцію  $f(x)$  в ряд Тейлора:

$$f(x+h) = f(x) + f'(x) \cdot h + \frac{f^{(2)}(x) \cdot h^2}{2!} + \frac{f^{(3)}(C_1) \cdot h^3}{3!}; \quad (6.36)$$

$$f(x-h) = f(x) - f'(x) \cdot h + \frac{f^{(2)}(x) \cdot h^2}{2!} - \frac{f^{(3)}(C_1) \cdot h^3}{3!}. \quad (6.37)$$

Якщо від рівняння (6.36) відняти вираз (6.37), матимемо:

$$f(x+h) - f(x-h) = 2f'(x) \cdot h + \frac{(f^{(3)}(C_1) + f^{(3)}(C_2)) \cdot h^3}{3!},$$

або

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} + O(h^2). \quad (6.38)$$

Аналогічно можна отримати подібні формули для старших похідних порядку точності  $O(h^2)$ :

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}; \quad (6.39)$$

$$f'''(x) \approx \frac{f(x+2h) - 2f(x+h) + 2f(x-h) - f(x-2h)}{2h^3}; \quad (6.40)$$

$$f^{(4)}(x) \approx \frac{f(x+2h) - 4f(x+h) + 6f(x) - 4f(x-h) + f(x-2h)}{h^4}. \quad (6.41)$$

Сам алгоритм обчислення похідної залишається незмінним, а саме: формується послідовність  $\{h_k\}$  так, щоб значення  $h_k \rightarrow 0$  й відповідно обчислюються елементи послідовності  $\{D_k\}$ , для розрахунку яких замість виразу (6.34) використовується одна з отриманих формул (6.38)–(6.41).

Також існують формули більш високих порядків точності, які можна знайти в спеціальній літературі.

## 6.8.2 Чисельне диференціювання експериментальних даних

Під час розв'язання багатьох практичних задач виникає необхідність визначення похідних від функцій, які задаються масивами експериментально отриманих даних. У цьому випадку безпосереднє диференціювання може дати хибні результати через невідокремлений «шум», який дуже сильно спотворює значення похідної. Приклад зображено на рисунку 6.10, де  $f(x)$  – істинний сигнал,  $\tilde{f}(x)$  – виміряний сигнал (із «шумом»).

Зрозуміло, що значення похідної від функції  $f(x)$  буде містити дуже суттєву випадкову складову величини, що вносить велике значення похибки у визначення дійсної похідної. Практичним шляхом розв'язання цієї проблеми є використання згладжуваного сигналу, що був отриманий із експерименту, шляхом інтерполяції чи апроксимації з наступним диференціюванням інтерполяційного полінома  $P(x)$  (чи апроксимувальної функції). Водночас, для отримання похідних вищих порядків необхідно ставити умову їх збіжності до сигналу та його інтерполяційного полінома. Значення похибки похідної інтерполяційної функції  $\Delta^*(x)$  дорівнює значенню похідної від похибки цієї функції  $\Delta'(x)$ :

$$\Delta(x) = \tilde{f}(x) - P(x); \quad \Delta^*(x) = \tilde{f}'(x) - P'(x),$$

тобто внаслідок лінійності операцій диференціювання та віднімання отримуємо  $\Delta^*(x) = \Delta'(x)$ .

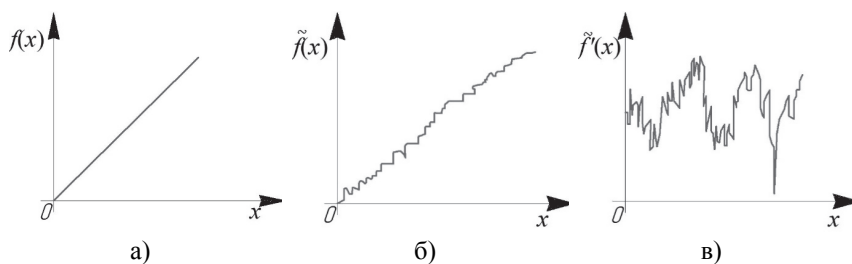


Рисунок 6.10 – Діаграми функцій сигналу:  
а) – дійсний; б) – виміряний; в) – похідна

У довідниках існують спеціальні таблиці, що дозволяють визначити із застосуванням різних різницевих інтерполяційних формул значення похідних. Такі формули нескладно отримати для будь-яких методів інтерполяції шляхом диференціювання інтерполяційних формул у загальному вигляді.

**Приклад 6.8.** Визначити значення похідної функції  $f(x)$ , яка задана табличними даними (табл. 6.5).

Таблиця 6.5 – Вихідні дані

$i$	0	1	2	3	4	5
$x$	0	1,0	2,0	3,0	4,0	5,0
$y$	3,0	6,0	9,0	10,0	11,0	12,0

*Розв'язання:*

Для наближеного оцінення похідних функції  $f(x)$  використовуються ліві кінцеві різниці, а саме значення кроку диференціювання:

$$h = x_i - x_{i-1} = x_1 - x_0 = 1,0 - 0,0 = x_2 - x_1 = x_3 - x_2 = x_4 - x_3 = x_5 - x_4 = 1,0.$$

Значення першої похідної на основі формули (6.34):

$$\left\{ \begin{array}{l} y'_0 = \frac{y_1 - y_0}{h} = \frac{6,0 - 3,0}{1,0} = 3,0; \\ y'_1 = \frac{y_2 - y_1}{h} = \frac{9,0 - 6,0}{1,0} = 3,0; \\ \dots\dots\dots; \\ y'_4 = \frac{y_5 - y_4}{h} = \frac{12,0 - 11,0}{1,0} = 1,0. \end{array} \right.$$

Значення другої і третьої похідних на основі формули (6.34):

$$\left\{ \begin{array}{l} y''_0 = \frac{y'_1 - y'_0}{h} = \frac{3,0 - 3,0}{1,0} = 0; \\ y''_1 = \frac{y'_2 - y'_1}{h} = \frac{1,0 - 3,0}{1,0} = -2,0; \\ \dots\dots\dots; \\ y'_3 = \frac{y'_5 - y'_4}{h} = \frac{1,0 - 1,0}{1,0} = 0. \end{array} \right.$$

$$\left\{ \begin{array}{l} y'''_0 = \frac{y''_1 - y''_0}{h} = \frac{(-2,0) - 0,0}{1,0} = -2,0; \\ y'''_1 = \frac{y''_2 - y''_1}{h} = \frac{0,0 - (-2,0)}{1,0} = 2,0; \\ y'''_2 = \frac{y''_3 - y''_2}{h} = \frac{0,0 - 0,0}{1,0} = 0. \end{array} \right.$$



Результати обчислення похідних заносяться в таблицю 6.6.

Таблиця 6.6 – Результати обчислення похідних функції  $f(x)$

$i$	0	1	2	3	4	5
$x$	0,0	1,0	2,0	3,0	4,0	5,0
$y$	3,0	6,0	9,0	10,0	11,0	12,0
$y'$	3,0	3,0	1,0	1,0	1,0	
$y''$	0,0	-2,0	0,0	0,0		
$y'''$	-2,0	2,0	0,0			

### **Висновки щодо застосування методів чисельного інтегрування та диференціювання**

Чисельне інтегрування функцій, що задаються масивом експериментальних даних, здійснюється за методами Ньютона-Котеса. У випадку аналітичного задання підінтегральної функції можливе застосування більш точних методів Гаусса і Чебишова або методу Монте-Карло. Методи Гаусса і Чебишова, які ще називають методами найвищої алгебраїчної точності, потребують забезпечення можливості обчислення підінтегральної функції у будь-якій точці інтервалу інтегрування, яка буде обчислена на основі порядку методу та інтервалів інтегрування. Це можливо тільки у разі її аналітичного завдання, бо будь-які оціночні підходи вносять додаткові похибки. Метод Монте-Карло, що базується на генерації та обробці випадкових чисел, використовує те, що математичне сподівання випадкової величини оцінюється середнім значенням послідовності випадкових чисел, а це надає можливість побудувати алгоритм оцінення інтеграла як послідовності операцій із згенерованою множиною випадкових (чи псевдовипадкових) чисел. Переваги цього методу особливо відчутні у разі його застосування для обчислення кратних інтегралів. Оцінювання похибок застосування чисельних методів інтегрування на практиці через складність аналітичного обчислення найчастіше здійснюється шляхом застосування кількох обчислень із різними кроками. В процесі чисельного диференціювання необхідно розуміння природи даних, що диференціюються, і за наявності суттєвої випадкової похибки виконувати попереднє згладжування даних.

## Контрольні запитання та завдання

1. Що називають інтегруванням функції? У чому полягає задача чисельного інтегрування функції?

2. Які існують методи чисельного інтегрування? Дайте їх порівняльний аналіз.

3. У чому полягає суть методу прямокутників для чисельного інтегрування функцій? Яка особливість використання методів лівих, правих і середніх прямокутників для чисельного інтегрування функцій?

4. Як впливає на точність чисельного інтегрування функцій використання методів лівих, правих і середніх прямокутників? Дайте визначення похибки обчислення за застосування відповідних чисельних методів.

5. Як визначаються коефіцієнти Котеса? Які властивості мають коефіцієнти Котеса за використання їх у визначенні значення інтеграла функції?

6. Яка послідовність застосування методів Ньютона-Котеса? Розробіть відповідні алгоритми. Чим відрізняються прості формули Ньютона-Котеса від складних?

7. Як отримують формули Ньютона-Котеса? Виведіть прості формули трапецій та Сімпсона для чисельного інтегрування, а також складові формул.

8. Яка особливість застосування формули Сімпсона для парної і непарної кількості інтервалів на усьому проміжку інтегрування?

9. Яким чином оцінюються похибки за застосування методів Ньютона-Котеса чисельного інтегрування функцій?

10. Обчислити інтеграл відповідним чисельним методом за вихідними даними, поданими в таблиці 6.7. Скласти алгоритм і програму обчислення. Оцінити похибку обчислення.

Таблиця 6.7 – Вихідні дані до завдання

Варіант	Функція інтегралу ( $I$ )	Крок обчислення ( $h$ )	Тип обчислювального методу
1	$\int_{1,0}^{10,0} x^2 \ln x dx$	0,20	Трапецій
2	$\int_{1,0}^{10,0} \frac{x}{\sqrt{1+x^2}} dx$	0,25	Лівих та правих прямокутників

Продовження таблиці 6.7

3	$\int_{1,0}^{10,0} \frac{x \sin x}{\ln x} dx$	0,20	Сімпсона
4	$\int_{1,0}^{10,0} \frac{\arctg x}{x} dx$	0,25	Лівих та правих прямокутників
5	$\int_{1,0}^{10,0} \sin x \lg x dx$	0,50	Трапецій
6	$\int_{1,0}^{10,0} \frac{\sin x}{x} dx$	0,30	Сімпсона
7	$\int_{1,0}^{10,0} x^3 \ln x dx$	0,50	Трапецій
8	$\int_{1,0}^{10,0} \frac{x}{\ln x} dx$	0,30	Сімпсона
9	$\int_{1,0}^{10,0} \cos x \ln x dx$	0,50	Лівих та правих прямокутників
10	$\int_{1,0}^{10,0} \frac{\sqrt{x^3 + 4}}{x} dx$	0,60	Сімпсона
11	$\int_{1,0}^{10,0} \frac{\cos x}{x^2} dx$	0,20	Лівих та правих прямокутників
12	$\int_0^{1,0} \frac{\cos x}{\ln x} dx$	0,20	Сімпсона
13	$\int_{1,0}^{10,0} x^4 \lg x dx$	0,25	Трапецій

11. Виведіть формулу Чебишова для чисельного інтегрування функції, заданої у трьох точках.

12. У чому полягає принциповий недолік формули Чебишова для чисельного інтегрування функції?

13. Що таке поліноми Лежандра і які основні властивості вони мають? Отримайте залежності для перших п'яти поліномів Лежандра.

14. Виведіть формулу Гаусса для чисельного інтегрування функції.

15. Як визначаються коефіцієнти у формулі Гаусса для чисельного інтегрування? Чому її називають формулою найвищої алгебраїчної точності?

16. Яким чином застосовують методи Гаусса і Чебишова та яка принципова відмінність між ними? Опишіть методіку, послідовність дій та розробіть алгоритм.

17. Обчислити інтеграл відповідним чисельним методом за вихідними даними, поданими в таблиці 6.8. Скласти алгоритм і програму обчислення. Оцінити похибку обчислення.

Таблиця 6.8 – Вихідні дані до завдання

Варіант	Функція інтегралу ( $I$ )	Порядок методу ( $n$ )	Тип обчислювального методу
14	$\int_{1,0}^{10,0} \frac{x \ln x}{\sqrt{1+x^2}} dx$	3	Чебишова
15	$\int_{1,0}^{10,0} \frac{x^2}{\ln x} dx$	5	Гаусса
16	$\int_{1,0}^{10,0} \frac{\lg x}{x^4} dx$	4	Чебишова
17	$\int_{1,0}^{10,0} \ln  x \sin x  dx$	6	Гаусса
18	$\int_{1,0}^{10,0} \frac{x^2}{\sqrt{x+1}} dx$	3	Чебишова
19	$\int_{1,0}^{10,0} \frac{\ln x}{x^2} dx$	5	Гаусса
20	$\int_{1,0}^{10,0} x \cdot \operatorname{arctg}(x) dx$	4	Чебишова
21	$\int_{1,0}^{10,0} \frac{\operatorname{arcctg} x}{x+1} dx$	3	Гаусса
22	$\int_{1,0}^{10,0} \sin 3x \lg(x+1) dx$	6	Чебишова
23	$\int_{1,0}^{10,0} \frac{\sin x}{x} dx$	7	Гаусса
24	$\int_{1,0}^{10,0} x^x \ln(x+1) dx$	3	Чебишова
25	$\int_{1,0}^{10,0} \frac{x+1}{\ln x^2} dx$	5	Гаусса
26	$\int_{1,0}^{10,0} \frac{\cos x}{\ln x} dx$	4	Чебишова
27	$\int_{1,0}^{10,0} \frac{\sqrt{x^2+1}}{x-2} dx$	6	Гаусса

Продовження таблиці 6.8

28	$\int_{1,0}^{10,0} \frac{\cos 2x}{x^x} dx$	5	Чебишова
29	$\int_0^{1,0} x \frac{\cos x}{\ln(x+2)} dx$	3	Гаусса
30	$\int_{1,0}^{10,0} x^2 \lg(x+2) \sin x dx$	3	Чебишова

18. Дайте порівняльний аналіз методів Чебишова, Гаусса і Ньютона-Котеса.

19. У чому полягає використання методу Монте-Карло для чисельного інтегрування функцій? Як визначається похибка обчислення? Яким чином можна підвищити точність методу?

20. Визначіть значення інтеграла  $I = \int_0^{10,0} x^2 dx$  чисельним методом Монте-Карло. Порівняйте отримане значення із значенням аналітичного виразу за різних значень обсягу вибірки послідовності випадкових чисел.

21. Визначіть значення інтеграла  $I = \int_{1,0}^{5,0} \ln(x) dx$  чисельним методом Монте-Карло. Порівняйте отримане значення із значенням аналітичного виразу за використання різних видів функцій закону розподілу ймовірностей.

22. Розробіть програму чисельного інтегрування за методом Монте-Карло для подвійного інтеграла  $I = \iint_{(\sigma)} (x^2 + y^2) dx dy$ , де область інтегрування  $\sigma$  визначається такими нерівностями, як:  $0,5 \geq x \geq 1,0$  і  $0 \geq y \geq 2x - 1$ . Чи потраплять у цю область інтегрування точки із координатами (0,55; 0,75), (0,25; 0,75), (0,25; 0,25), (0,99; 0,70)?

23. Яким чином оцінюються похибки у разі застосування методів Чебишова і Гаусса чисельного інтегрування функцій?

24. Яким чином здійснюється чисельне диференціювання аналітично заданих функцій? Яким чином можна підвищити точність чисельного диференціювання функцій?

25. Як визначається похибка під час чисельного диференціювання аналітично заданих функцій?

26. Продиференціюйте функцію  $f(x) = x^2 \sin(x)$  на проміжку  $x \in [0; 10,0]$ . Оцініть похибку обчислення за допомогою порівняння між собою значень похідної функції обчисленої аналітичним і чисельним методами.

27. Визначіть другу й третю похідні функції  $f(x)=x\ln(x)$  на проміжку  $x \in [0; 5,0]$ . Оцініть похибку обчислення за допомогою порівняння між собою значень похідної функції, обчисленої аналітичним і чисельним методами.

28. Яким чином здійснюється чисельне диференціювання функцій, визначених за допомогою експериментальних даних?

29. Визначіть значення першої і другої похідної функції  $f(x)$ , яка задана табличними даними (табл. 6.9).

Таблиця 6.9 – Вихідні дані до завдання

$i$	0	1	2	3	4	5	6	8
$x$	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8
$y$	2,5	3,4	5,2	5,8	5,9	6,3	6,8	7,1

30. Яким чином визначається похибка чисельного диференціювання експериментально визначеної функції?

## ЛІТЕРАТУРА

1. Задачин В. М. Чисельні методи : навчальний посібник / В. М. Задачин, І. Г. Коношенко. – Х. : Вид. ХНЕУ ім. С. Кузнеця, 2014. – 180 с.
2. Різницеві методи та сплайни в задачах багатовимірної інтерполяції / [Р. Н. Кветний, В. Ю. Дементьев, М. О. Машницький, О. О. Юдін]. – Вінниця : УНІВЕРСУМ-Вінниця, 2009. – 87 с.
3. Кветний Р. Н. Методи комп'ютерних обчислень: навчальний посібник / Р. Н. Кветний – Вінниця : ВНТУ, 2001. – 218 с.
4. Комп'ютерне моделювання систем та процесів. Методи обчислень / за заг. ред. Р. Н. Кветного – Вінниця : ВНТУ, 2012. – Ч. 1 – 196 с.; Ч. 2 – 230 с.
5. Ляшенко М. Я. Чисельні методи : підручник / М. Я. Ляшенко, М. С. Головань. – К. : Либідь, 1996. – 288 с.
6. Моделювання та оптимізація систем : [підручник] / В. М. Дубовой, Р. Н. Кветний, О. І. Михальов, А. В. Усов. – Вінниця : ПП «ТД«Едельвейс», 2017. – 804 с.
7. Усов А. В. Чисельні методи та їх реалізація у середовищі SCILAB : навчальний посібник / А. В. Усов, О. А. Шпинковський, М. І. Шпинковська – Київ : Освіта України, 2013. – 192 с.
8. Чабан В. Чисельні методи : навчальний посібник / В. Чабан – Львів : Вид. Нац. ун-ту «Львівська політехніка», 2001. – 186 с.
9. Фельдман Л. П. Чисельні методи в інформатиці : підручник / Л. П. Фельдман, А. І. Петренко, О. А. Дмитрієва – К. : Вид. група ВНУ, 2006. – 480 с.
10. Abramovitz M. Handbook of mathematical functions. With Formulas, Graphs, and Mathematical Tables/ Abramowitz M., Stegun I. A. – National Bureau of Standards, N.Y., 1972–1044 p.
11. Collatz L. Functional Analysis and Numerical Mathematics. / L. Collatz – Academic Press, New York, 1966.– 494 p.
12. Forsythe G. E. Computer Methods for Mathematical Computations. Englewood Cliffs / G. E. Forsythe, M. A. Malcolm, C. B. Moler – New Jersey 07632. Prentice Hall, Inc., 1977. XI. – 259 p.
13. Kvyetnyy R. Basics of Modelling and Computational Methods / R. Kvyetnyy. – Вінниця : ВДТУ, 2007. – 147 с.

*Навчальне видання*

**Кветний Роман Наумович  
Іванчук Ярослав Володимирович  
Богач Ілона Віталіївна  
Софіна Ольга Юріївна  
Барабан Марія Володимирівна**

**Методи та алгоритми  
комп'ютерних обчислень.  
Теорія і практика**

Підручник

Редактор *Т. Старичек*

Оригінал-макет підготовлено в *редакційно-видавничому відділі ВНТУ*

Підписано до друку 21.03.2022. Формат 29,7×42¼.

Гарнітура Times New Roman, Lucida Console.

Папір офсетний. Ум. друк. арк. 16,17.

Наклад 50 пр. Зам. № 2023-004.

Видавець та виготовлювач –  
Вінницький національний технічний університет,  
редакційно-видавничий відділ.

21021, м. Вінниця, Хмельницьке шосе, 95,

ВНТУ, ГНК, к. 114.

**press.vntu.edu.ua**

*email: irvc.vntu@gmail.com*

Свідоцтво суб'єкта видавничої справи  
серія ДК № 3516 від 01.07.2009 р.