МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЦЕНТРАЛЬНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

Практикум: “Чисельні методи”

Вказівки до виконання практичних завдань
для студентів денної та заочної форми навчання за спеціальністю
123 “Комп’ютерна інженерія ”,125 “Кібербезпека”

ЗАТВЕРДЖЕНО
на засіданні кафедри кібербезпеки та програмного забезпечення,
протокол від 29 травня 2019 року № 14

КРОПИВНИЦЬКИЙ
2019

Елементи векторної комп’ютерної графіки: метод. вказівки до виконання лабораторних робіт для студентів денної та заочної форми навчання за спеціальністю 123 “Комп’ютерна інженерія”, 125 “Кібербезпека” / уклад. Дрєєва Г.М., Дрєєв О.М., Якименко Н.М., Денисенко О.О. — Кропивницький: ЦНТУ, 2019. — 90 с.

Укладачі: Дрєєва Г.М., Дрєєв О.М., Якименко Н.М., Денисенко О.О.

Рецензенти: Якименко М.С. – кандидат фізико-математичних наук, доцент кафедри кібербезпеки та програмного забезпечення.

Зміст

1. НАБЛИЖЕНІ ЧИСЛА....4
1.1. Види похибок....4
1.2. Сумнівні та вірні цифри числа....7
1.3. Додавання та віднімання наближених чисел....9
1.4. Множення та ділення наближених чисел....10
1.5. Відшукання похибки результату обчислень за допомогою диференціального числення....12
Завдання до першого розділу....15
2. ІНТЕРПОЛЯЦІЯ....19
2.1. Інтерполяція степеневим поліномом....19
2.2. Інтерполяція многочленами інших типів....24
Завдання для другого розділу....27
3. АПРОКСИМАЦІЯ ФУНКЦІЙ....28
3.1. Рівномірне наближення поліномами Бернштейна....28
3.2. Формула Тейлора....31
3.3. Метод найменших квадратів....37
Завдання до третього розділу....46
4. РОЗВ'ЯЗАННЯ РІВНЯНЬ З ОДНІЄЮ НЕВІДОМОЮ....47
4.1. Методи послідовного звуження інтервалу....48
4.2. Методи послідовного наближення....52
4.3. Методи, які використовують особливості поведінки функції....55
Завдання до четвертого розділу....57
5. СИСТЕМИ ЛІНІЙНИХ РІВНЯНЬ....59
5.1. Умови єдиності та існування розв’язків....60
5.2. Метод Крамера....60
5.3. Метод підстановки....62
5.4. Метод Гауса....63
5.5. Матричний метод....65
Завдання до пя'того розділу....66
6. МЕТОД ІТЕРАЦІЙ....67
6.1. Нелінійні рівняння з однією змінною....67
6.2. Система лінійних рівнянь....70
6.3. Система нелінійних рівнянь....72
Завдання до шостого розділу....74
7. ЧИСЕЛЬНЕ ІНТЕГРУВАННЯ....75
7.1. Метод прямокутників....75
7.2. Метод трапецій....79
7.3. Метод Симпсона....81
Завдання до сьомого розділу....84
8. ДИФЕРЕНЦІАЛЬНІ РІВНЯННЯ, ЗАДАЧА КОШІ....85
8.1. Метод Ейлера для задач Коші першого порядку....86
8.2. Метод Ейлера для задач Коші другого порядку....88
Завдання до восьмого розділу....90

1. НАБЛИЖЕНІ ЧИСЛА

Електронна обчислювальна машина (ЕОМ) насамперед створена для ефективного проведення математичних розрахунків. Всі схеми та їх поєднання в єдину систему розраховані на ефективне, швидке та якісне проведення арифметичних дій із числами різного формату. Та для кожної цифрової системи обчислення існує обмеження пам'яті для зберігання числа, тобто будь яке ірраціональне число та переважна більшість раціональних чисел дозволеного діапазону записані не повністю, а з обмеженою кількістю цифр у числі. Такі числа називають наближеними, а відмінність записаного числа від реальної величини — похибкою. Похибки можуть бути незначними і не впливати на результат обчислень, так і можуть бути значно більшими за величину, яку вимірювали. При практичних обчисленнях знання про похибку результату обчислення є дуже важливим, і в деяких випадках можуть застерегти від неправильного результату, бо іноді незнання відповіді є менш шкідливим ніж вважати за выдповыдь неправильне значення.

Практичні обчислення в більшості випадках проводяться з наближеними числами. І це зрозуміло, бо калькулятори та комп’ютери для представлення чисел використовують обмежену кількість пам’яті. Виявляється, що не завжди можна так вільно оперувати наближеними числами як точними, можливі ситуації коли результат обчислення буде мати дуже великі відхилення від дійсних значень. Для того щоб передбачити такі ситуації та вміти поставити обчислення найвигіднішим, щодо похибок, шляхом і вводиться цей розділ.

1.1. Види похибок

Розв'яжімо гіпотетичну задачу: древньогрецький філософ-мандрівник спостерігав за рухом Місяця на фоні зірок та тіні від Сонця. Це дало йому

можливість зробити висновок про форму та розміри Землі і також розрахувати відстань до Місяця. Припустимо, що він отримав результат 400000км з можливим відхиленням від дійсного результату ±100000км. В той самий час філософ-лунатик на Місяці міркував ро свій розмір, сили тяжіння, кількість води на Сонці та хутра на бізонах, прийшов до висновку що зріст людей на Землі складає 2м ±1м. Питання задачі полягає у порівнянні якості проведених числових оцінок – який філософ визначив величину точніше.

Для розв'язання цієї задачі є необхідним визначитися як порівнювати похибки і коли це можливо. Тому ведемо поняття похибки.

Означення: **Абсолютна похибка** – модуль різниці між точним значенням числа та його наближенням.

Наприклад, для рівності π=3,14 модуль різниці між даним наближенням π та більш точним його значенням:

$$|3,1415926-3,14|=0,0015926,$$

дає абсолютну похибку рівності π=3,14.

Означення: **Відносна похибка** – відношення абсолютної похибки до модуля точного значення числа. Найчастіше виражається в процентах.

На практиці майже завжди неможливо визначити точні значення похибок і тому використовують поняття **граничної** похибки, тобто такої похибки, яка завжди більша або рівна реальному відхиленню. Надалі без додаткових вказівок всі похибки вважатимемо граничними.

Введемо позначення:

A – точне значення числа;

a – наближення числа A;

Δ – абсолютна похибка;

ε – відносна похибка.

Тоді у вигляді формул:

$\Delta_a = |A - a|$, $\varepsilon_a = \frac{\Delta_a}{|A|}$, або в процентах $\varepsilon_a = \frac{\Delta_a}{|A|} \cdot 100\%$.

Тепер повернемося до питання про філософів, хто з них точніше визначив величину? Для цього знайдемо відносні похибки відстані до Місяця:

$\varepsilon_М = 100\% \cdot 100000км / 400000км = 25\%$,

та зросту людини:

$\varepsilon_л = 100\% \cdot 1м / 2м = 50\%$.

В результаті порівняння відносних похибок бачимо, що похибка відстані до Місяця в два рази менша, і тому вважатимемо земного філософа більш точним.

Приклади:

1) Задане наближене число a=3,142, та його абсолютна гранична похибка $\Delta_a = \pm 0{,}001$. Знайти відносну граничну похибку.

Розв'язання:

Так як ми не знаємо точного значення числа, то $\varepsilon_a = \Delta_a / a$:

$$\varepsilon_a = 0{,}001/3{,}142 = 0{,}000318 = 0{,}0318\%$$

2) Відоме число a=12,4 та його відносна похибка ε_a=1%. Знайти абсолютну похибку Δ_a.

Розв'язання:

$$\Delta_a = a\varepsilon_a = 12{,}4 * 0{,}01 = \pm 0{,}124$$

3) Є дві рівності: 2/3=0,66 та 11/7=1,57. Визначити яке з них точніше.

Розв'язання:

Щоб визначити відповідь, достатньо порівняти відносні похибки цих рівностей.

Для першої рівності:

$\varepsilon_1=|2/3-0,66|/(2/3)=0,01=1\%$.

Для другої рівності:

$\varepsilon_2=|11/7-1,57|/(11/7)=0,000909=0,0909\%$.

$\varepsilon_1>\varepsilon_2$, тобто похибка першої рівності більша, тому друга рівність точніше.

4) На мові паскаль для запису чисел single з плавучою комою використовується 4-байтний запис (32 біти). В результаті для запису мантиси відводиться вісім десяткових цифр. Максимальне значення мантиси сягає 9.9999999, а мінімальне 1.0000000 з абсолютними похибками не більші за ±0.00000005. Тепер легко визначити відносні похибки:

$\varepsilon_1=100\%\cdot 0.00000005/9.9999999=0.0000005\%$

$\varepsilon_2=100\%\cdot 0.00000005/1.0000000=0.000005\%$

1.2. Сумнівні та вірні цифри числа

При вимірюваннях реальних величин за допомогою приладів, отримаємо числа які є наближеними. Наприклад, шкільна лінійка має мінімальну поділку в 1мм. Це обмежує точність визначення розмірів об’єкту одним міліметром. Також, коли виміряємо об’єкт різними лінійками, майже завжди отримаємо різні результати, бо лінійки зроблені з різного матеріалу по різному оброблялися та зазнали різноманітних деформацій. Але кожна лінійка має своє максимальне відхилення від точного значення і загалом не повинне перевищувати зазначеної величини. Для приладів, як правило, максимальне відхилення не перевищує половини ціни поділки. Тому провівши виміри тією ж лінійкою, можемо отримати таке значення: (12,4±0,05)см.

Зрозуміло, що точне значення повинне лежати між 12,35 та 12,45, і якою б лінійкою ми не вимірювали (що відповідає стандартам), завжди матимемо 12см., а от кількість міліметрів буде то 3, то 4. Тому, в даному

випадку називатимемо цифри, що виражають кількість сантиметрів вірними, а цифри, що виражають кількість сантиметрів – сумнівними.

Вірні цифри наближеного числа – ті цифри, які співпадають в точному та наближеному числі.

Сумнівні цифри – ті цифри, що не співпадають з цифрами точного числа.

Також можна сказати і по іншому:

Вірними цифрами наближеного числа називають цифри, що не змінюють свого значення, якщо похибка не перевищуватиме максимально вказану (граничну). Всі інші цифри називатимемо **сумнівними**.

На практиці наближені числа записують так, щоб лише остання значуща цифра могла містити похибку, тобто була сумнівною. Наприклад, запис 1,012 означає, що гранична абсолютна похибка рівна ±0,0005, а запис 1,0000 визначає похибку ±0,00005. Тому в наближених числах при запису відкидати “зайві” нулі не варто.

Приклади:

1) Відоме число a=5,2351 та його абсолютна похибка 0,003. Округлити число залишивши лише вірні цифри.

Розв’язання:

Проведемо округлення поетапно. Так як абсолютна похибка впливає на третій знак після коми, то можна округлити число зразу до другого знаку та перевірити, чи всі цифри в округленому числі вірні:

a=5,2351≈5,235≈5,24.

Знайдемо похибку, що дає округлення:

|5,2351-5,24|=0,0049

та додамо до неї “рідну” похибку:

0,0049+0,003=0,0079.

В результаті 5,24-0,0079≤a≤5,24+0,0079; 5,2321≤a≤5,2479; 5,23≤a≤5,25.

Тобто при округленні до тих самих двох знаків після коми, остання цифра може змінити своє значення, тому число a потрібно округлити ще на один знак (при необхідності цю операцію повторюють не один раз):

$a \approx 5,2$.

Знайдемо похибку, що дає округлення:

$|5,2351-5,2|=0,0351$

та додамо до неї "рідну" похибку:

$0,0351+0,003=0,0381$.

В результаті $5,2-0,0381 \leq a \leq 5,2+0,0381$; $5,1679 \leq a \leq 5,2381$; $5,2 \leq a \leq 5,2$.

В результаті число 5,2 має лише вірні цифри.

1.3. Додавання та віднімання наближених чисел

Звісно, при використанні наближених значень величин в розрахунках ми також отримаємо наближені значення і для нього також необхідно знати величну похибки. Як далі буде показано, після неправильно поставлених розрахунках, відносна похибка після обчислень може зростати в багато разів зводячи нанівець всі зусилля обчислень.

Нехай ми маємо два наближених числа a та b. Маємо також їх точні значення A,B, та абсолютні похибки Δ_a, Δ_b. Тоді:

$$A+B=a+\Delta_a+b+\Delta_b=(a+b)+(\Delta_a+\Delta_b).$$

Тобто при складанні двох наближених чисел щоб знайти абсолютну похибку результату, потрібно скласти абсолютні похибки доданків. При відніманні картина не змінюється – похибки теж додаються, бо взявши число $b=-c$ отримаємо ті самі викладки.

Оцінити ж відносну похибку після додавання чи віднімання можна буде за такими співвідношеннями:

$$\varepsilon_{a+b}=\frac{\Delta_a+\Delta_b}{a+b} \quad , \quad \varepsilon_{a-b}=\frac{\Delta_a+\Delta_b}{a-b} \quad .$$

Приклад:

Оцінити відносну та абсолютну похибки після додавання та віднімання двох чисел: 4,12±0,0042 та 3,98±0,0037.

Розв'язання:

Додавання:

Δ_1=0,0042+0,0037=0,0079;

ε_1=(0,0042+0,0037)/(4,12+3,98)=0,000975=0,0975%

Віднімання:

Δ_2=0,0042+0,0037=0,0079;

ε_2=(0,0042+0,0037)/(4,12-3,98)=0,056=5,6%

Відносна похибка відрізняється більш як в 50 разів!

1.4. Множення та ділення наближених чисел

Повернемось до позначень попереднього пункту 1.3. та проведемо перетворення для дії множення:

$$A\cdot B=(a+\Delta_a)\cdot(b+\Delta_b)=ab+a\Delta_b+b\Delta_a+\Delta_a\Delta_b\approx ab+(a\Delta_b+b\Delta_a).$$

Тобто

$$\Delta_{a\cdot b}= a\Delta_b+b\Delta_a.$$

Для відносної похибки:

$$\varepsilon_{a\cdot b}=\frac{a\Delta_b+b\Delta_a}{ab}=\frac{\Delta_b}{b}+\frac{\Delta_a}{a}=\varepsilon_a+\varepsilon_b \quad .$$

Тепер повторимо перетворення для ділення:

$$\Delta_{a/b}=\frac{a+\Delta_a}{b+\Delta_b}-\frac{a}{b}=\frac{ab+b\Delta_a-ab+a\Delta_b}{b(b+\Delta_b)}=\frac{b\Delta_a+a\Delta_b}{b(b+\Delta_b)} \quad ,$$

$$\Delta_{a/b}=\frac{b\Delta_a+a\Delta_b}{b(b+\Delta_b)}=a\cdot b\frac{\Delta_a/a+\Delta_b/b}{b(b+\Delta_b)}=\frac{a}{b+\Delta_b}\cdot(\Delta_a/a+\Delta_b/b) \quad ,$$

$$\frac{\Delta_{a/b}}{a/b}\approx\frac{\Delta_a}{a}+\frac{\Delta_b}{b} \quad .$$

Або $\varepsilon_{a/b}=\varepsilon_a+\varepsilon_b$.

Як показано в попередніх викладках, що при множенні, що при діленні, для знаходження відносної похибки результату потрібно скласти відносні похибки аргументів дії.

Приклад:

Дані числа: $a=2{,}122\pm0{,}013$, $b=5{,}5\pm0{,}06$. Знайти відносну та абсолютну похибку результату обчислень за формулою: $c=ab-a/b$.

Розв’язання:

$ab=11{,}671$,

$a/b=0{,}3858$,

$\varepsilon_{a/b}=\varepsilon_{ab}=\varepsilon_a+\varepsilon_b=\Delta_a/a+\Delta_b/b=0{,}013/2{,}122+0{,}06/5{,}5\approx0{,}017$,

$\Delta_{ab}=ab\cdot\varepsilon_{ab}=11{,}671\cdot0{,}017=0{,}198\approx0{,}2$,

$\Delta_{a/b}=a/b\cdot\varepsilon_{a/b}=0{,}3858\cdot0{,}017=0{,}00656$,

$\Delta_{ab-a/b}=\Delta_{ab}+\Delta_{a/b}=0{,}2+0{,}00656\approx0{,}2$

Відповідь: $ab-a/b=11{,}3\pm0{,}2$, $\varepsilon_{ab-a/b}=0{,}2/11{,}3\approx0{,}0177=1{,}77\%$.

1.5. Відшукання похибки результату обчислень за допомогою диференціального числення

Ідея використання диференціального числення для визначення похибок показана на малюнку 1. Лише покажемо як цей принцип використати для функцій з декількома змінними.

Нехай є функція $y=f(x_1,x_2,\ldots,x_n)$, тоді знаючи відхилення i-тої змінної (Δx_i), можна оцінити максимальне відхилення значення функції, що залежить саме від i-тої змінної:

$$\Delta y_i \approx f'_{x_i}(x_1, x_2, \ldots, x_n)\cdot \Delta x_i \quad .$$

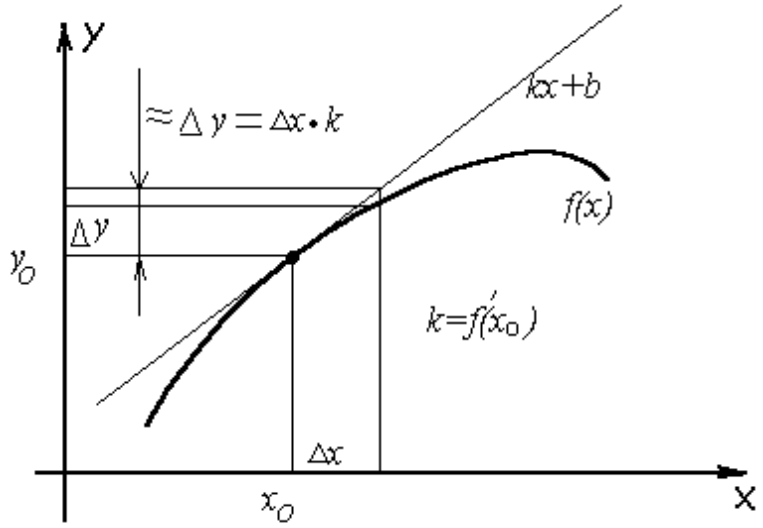

Рис. 1.1

Звісно, щоб взнати повне максимальне відхилення необхідно просумувати всі можливі похибки, а так як ми не знаємо знак відхилення, кожний доданок потрібно брати за модулем, отримуючи, таким чином оцінку саме максимального відхилення результату:

$$\Delta y \approx \sum_{i=1}^{n} \left| f'_{x_i}(x_1, x_2, \ldots, x_n)\cdot \Delta x_i \right| \quad .$$

Приклади:

1) Знайти відносну похибку результату обчислень за формулою $f(x)=\sin(3x)\cdot x^2$ при $x_1=0$ та $\Delta x_1=0{,}01$, при $x_2=1/3$ та $\Delta x_2=0{,}01$.

Розв'язання:

Знайдемо похідну: $f'(x)=3\cos(3x)x^2+2\sin(3x)x$, та її значення в зазначених точках:

$$f'(x_1)=0 \quad , \quad f'(x_2)=0{,}741 \quad .$$

Також нам потрібні значення функції:

$$f(x_1)=0 \quad , \quad f(x_2)=0{,}0935 \quad .$$

Тепер знайдемо оцінки максимальних похибок:

$$\Delta y_1 \approx f'(x_1) \cdot \Delta x_1 = 0 \cdot 0{,}01 = 0 \quad ,$$

$$\Delta y_2 \approx f'(x_2) \cdot \Delta x_2 = 0{,}741 \cdot 0{,}01 = 0{,}00741.$$

Відносну похибку при x_1 порахувати неможливо, так як значення функції рівне нулю, а на нуль ділити не можна, тому знайдемо відносну похибку лише при x_2:

$\varepsilon_2=\Delta y_2/y_2=0{,}00741/0{,}0935\approx 0{,}0793=7{,}93\%$.

Висновок: при досить гладкій функції (мала перша похідна) абсолютна похибка менша ніж при швидкозмінній функції.

2) Оцінити відносну похибку що дає вираз $\dfrac{a^{1/2}b^3}{c^{2/3}}$. Знайти відповідь в загальному вигляді.

Розв'язання:

Знайдемо всі часткові похідні даної залежності $f(a,b,c)$:

$$f'_a = \frac{1}{2}\frac{b^3}{a^{1/2}c^{2/3}},$$
$$f'_b = 3\frac{a^{1/2}b^2}{c^{2/3}},$$
$$f'_c = -\frac{2}{3}\frac{a^{1/2}b^3}{c^{5/3}}.$$

Просумуємо похибки знайшовши абсолютну похибку:

$$\Delta f \approx \Delta a \cdot \left|\frac{1}{2}\frac{b^3}{a^{1/2}c^{2/3}}\right| + \Delta b \cdot \left|3\frac{a^{1/2}b^2}{c^{2/3}}\right| + \Delta c \cdot \left|-\frac{2}{3}\frac{a^{1/2}b^3}{c^{5/3}}\right| \quad .$$

Поділимо абсолютну похибку на значення виразу:

$$\varepsilon = \frac{\Delta a \cdot \left|\frac{1}{2}\frac{b^3}{a^{1/2}c^{2/3}}\right| + \Delta b \cdot \left|3\frac{a^{1/2}b^2}{c^{2/3}}\right| + \Delta c \cdot \left|-\frac{2}{3}\frac{a^{1/2}b^3}{c^{5/3}}\right|}{\left|\frac{a^{1/2}b^3}{c^{2/3}}\right|} = \frac{\Delta a}{2a} + 3\frac{\Delta b}{b} + \frac{2\Delta c}{3c} \quad .$$

Остаточна відповідь:

$$\varepsilon = \tfrac{1}{2}\varepsilon_a + 3\varepsilon_b + \tfrac{2}{3}\varepsilon_c \quad .$$

Завдання до першого розділу

1) Знайти відносну похибку числа a, якщо відома його абсолютна похибка Δ_a:

№ варіанту	a	Δ_a	№ варіанту	a	Δ_a
1	3746 13 23,343	±1,8 ±0,02 ±0,0023	2	1345 13,4643 8,346	±2,724 ±1,25 ±0,6384
3	345 34,21 1,356	±2,345 ±1,245 ±0,0452	4	354 51,467 4,5436	±2,457 ±1,543 ±0,25
5	8756 65,34 3,567	±2,356 ±1,723 ±0,034	6	324 45,547 2,783	±2,758 ±1,534 ±0,856
7	2674 45,65 3,864	±2,2356 ±1,857 ±0,1324	8	865 34,363 3,4364	±2,257 ±1,364 ±0,8045
9	576 78,56 0,3563	±1,865 ±2,6325 ±0,748	(0) 10	467 2,343 54,244	±2,523 ±1,784 ±0,787
11	356 3,576 98,895	±2,236 ±1,65 ±0,74	12	334 9,33 6,6666	±2,087 ±1,235 ±0,689
13	963 45,75 8,32354	±2,093 ±1,06 ±0,2930	14	23 12,343 3,21	±2,438 ±1,063 ±0,683
15	65 4,6746 0,673	±2,64 ±1,2374 ±0,123	16	1477 71,45 3,950	±2,683 ±1,073 ±0,873
17	36 4,457 34,6875	±2,2398 ±1,932 ±0,23	18	134 23,5757 75,431	±2,944 ±1,3296 ±0,0934

№ варіанту	a	Δ_a	№ варіанту	a	Δ_a
19	753 1,34 456,467	±2,32 ±1,532 ±0,52	20	392 87,4368 7,12674	±2,955 ±1,5834 ±0,634
21	214 2,436 45,5674	±2,923 ±1,38 ±0,2398	22	96 10,625 3,904	±2,6873 ±1,6673 ±0,4673
23	975 7,534 34,34	±2,3296 ±1,36 ±0,39	24	8643 3,74553 54,5376	±2,4673 ±1,734 ±0,34567
25	3245 356,78 1,11	±2,34 ±1,39 ±0,37	26	3832 34,4326 1,465	±2,743 ±1,5763 ±0,734
27	235 56,56 5,565	±2,374 ±1,93 ±0,309	28	554 34,3256 1,7854	±2,734 ±1,744 ±0,437
29	94 3,8765 12,3546	±2,938 ±1,323 ±0,36	30	357 2,476 65,326	±2,34457 ±1,3647 ±0,4667
31	724 78,45 7,8424	±2,94 ±1,2354 ±0,38	32	854 67,6587 4,6437	±2,74443 ±1,684 ±0,74476

2) Визначити відносну похибку чисел записаних у ЕОМ форматі:

single (float, 32 bit з плавучою комою);

real (48 bit з плавучою комою);

double (64 bit з плавучою комою);

extended (long double 80 bit з плавучою комою).

3) Знаючі числа та їх похибки зробити округлення

№ варіанту	*a*	№ варіанту	*a*
1	935,32±0,012 487,98 ε=0,1%	2	743,83±0,021 749,83 ε=0,8%
3	623,87±0,022 242,88 ε=0,2%	4	316,84±0,023 745,73 ε=0,7%
5	603,47±0,032 283,83 ε=0,3%	6	846,65±0,034 827,74 ε=0,6%
7	383,38±0,042 187,23 ε=0,4%	8	843,48±0,054 628,57 ε=0,5%
9	494,29±0,052 946,27 ε=0,5%	10	749,94±0,0654 872,48 ε=0,4%
11	321,23±0,062 946,38 ε=0,6%	12	826,75±0,065 374,85 ε=0,3%
13	373,38±0,073 932,27 ε=0,7%	14	737,75±0,076 873,85 ε=0,2%
15	622,19±0,053 832,13 ε=0,8%	16	749,84±0,087 846,85 ε=0,1%
17	272,47±0,064 821,48 ε=0,9%	18	927,85±0,090 612,73 ε=0,2%
19	276,37±0,075 272,86 ε=0,1%	20	867,93±0,078 749,94 ε=0,3%
21	825,74±0,086 162,47 ε=0,2%	22	739,92±0,054 774,82 ε=0,4%
23	387,57±0,096 273,54 ε=0,3%	24	927,84±0,056 824,22 ε=0,5%
25	753,39±0,063 429,64 ε=0,4%	26	843,84±0,034 826,84 ε=0,6%
27	272,84±0,042 312,47 ε=0,5%	28	912,61±0,053 859,74 ε=0,7%
29	373,85±0,053 373,58 ε=0,6%	30	162,74±0,075 842,47 ε=0,8%
31	465,74±0,064 646,57 ε=0,7%	32	823,84±0,018 836,75 ε=0,3%

4) Використати тип з плавучою комою довжиною 32 біти (float, single) для знаходження середнього арифметичного *f*(N+30) (N – останнє двозначне число в заліковій книжці) 10000 значень. Свій варіант *f*(*x*) визначте з таблиці:

№ варіанту	*f*(*x*)=	№ варіанту	*f*(*x*)=	№ варіанту	*f*(*x*)=
1	sin(x)cos(x)	2	cos(x)/exp(x)	3	cos(x)/exp(3x)
4	cos(x)cos(3x)	5	sin(2x)cos(x)	6	sin(2x)cos(3x)
7	tg(x)sin(x)	8	cos(2x)cos(x)	9	cos(2x)cos(3x)
10	exp(x)	11	tg(2x)sin(x)	12	tg(2x)sin(3x)
13	cos(x)	14	exp(2x)	15	exp(3x)
16	sin(x)	17	cos(2x)	18	cos(3x)
19	exp(x)cos(x)	20	sin(2x)	21	sin(3x)
22	sin(x)exp(x)	23	exp(2x)cos(x)	24	exp(2x)cos(3x)
25	exp(x)/cos(x)	26	sin(2x)exp(x)	27	sin(2x)exp(3x)
28	exp(x)/sin(x)	29	exp(2x)/cos(x)	30	exp(2x)/cos(3x)
31	sin(x)/exp(x)	32	exp(2x)/sin(x)	33	exp(2x)/sin(3x)

Для обчислення створіть програму на довільній мові програмування та проведіть розрахунки:

а) Скласти в циклі 10000 раз значення *f*(N+1), результат поділити на кількість доданків 10000, записати отриманий результат.

б) Скласти в циклі 10000 раз значення *f*(N+1)/10000, результат записати.

в) Знаючи точний результат середнього арифметичного (складаються однакові числа 10000 раз, і результат ділиться на 10000, тому результат повинен бути рівним значенню *f*(N+1)), знайти відносну та абсолютну похибки обчислень.

г) Спробуйте теоретично передбачити максимальну похибку при знаходженні середнього арифметичного двома методами.

д) Додаткове завдання: розробити більш точний алгоритм для знаходження середнього арифметичного значення.

е) Зробіть висновки: де значення точніше й чому.

2. ІНТЕРПОЛЯЦІЯ

Інтерполяція являє собою дію знаходження функції певного виду, графік якої проходить через всі задані точки. Якщо такі точки отримані як значення заданої функції, то в вказаних точках значення заданої функції та функції що отрималася в результаті інтерполяції співпадають. Відхилення від значень заданої функції в інших точках може бути, в загальному випадку, довільним, але, прийнявши умову гладкості заданої функції, зазвичай вважаються близькими.

Використовують інтерполяцію при пошуках проміжних значень функцій заданих таблично, при наближених пошуках значень визначених інтегралів, чисельне диференціювання та ін.

2.1. Інтерполяція степеневим поліномом

2.1.1. Метод невизначених коефіцієнтів один з самих трудомістких, але є найбільш легким для зрозуміння. Нехай $Q_n(x)$ є многочленом степеня n:

$$Q(x)=a_0+a_1x+a_2x^2+\ldots+a_nx^n \quad . \tag{2.1}$$

Тоді для знаходження коефіцієнтів a_i побудуємо систему рівнянь:

$$\begin{cases} y_0=a_0+a_1x_0+a_2x_0^2+\ldots+a_nx_o^n \\ y_1=a_0+a_1x_1+a_2x_1^2+\ldots+a_nx_1^n \\ \ldots \\ y_m=a_0+a_1x_m+a_2x_m^2+\ldots+a_nx_m^n \end{cases} . \tag{2.2}$$

Відповідно, щоб система 2.2 мала єдиний розв’язок необхідно щоб $m=n$. Тобто, якщо маємо $m+1$ точок, то отримується многочлен степеня m.

Розв’язавши систему 2.2 отримаємо всі невідомі коефіцієнти.

Приклад:

Точки задані в таблиці. Провести інтерполяцію степеневим многочленом методом невизначених коефіцієнтів.

x	1	2	3	4
y	2	7	14	23

Розв’язання:

Запишемо систему рівнянь за зразком 2.2:

$$\begin{cases} 2=a_0+a_1 1+a_2 1^2+a_3 1^3 \\ 7=a_0+a_1 2+a_2 2^2+a_3 2^3 \\ 14=a_0+a_1 3+a_2 3^2+a_3 3^3 \\ 23=a_0+a_1 4+a_2 4^2+a_3 4^3 \end{cases}.$$

Розв’язавши цю систему будь-яким з методів отримаємо результат:

a_0=-1, a_1=2, a_2=1, a_3=0.

Остаточна відповідь: $y=x^2+2x-1$ – многочлен другого степеня.

2.1.2. Інтерполяція за формулами Лагранжа теж дає степеневий поліном степінь якого, в загальному випадку, менше на одиницю кількості точок інтерполяції, але представлений в іншому вигляді. При цьому відпадає необхідність розв’язувати систему лінійних рівнянь, але, при необхідності можна отримати класичний вигляд многочлену, лише потрібно розкрити дужки та звести подібні.

Будується многочлен таким чином:

Для кожної *i*-тої точки будується дріб, які потім сумуються в єдиний вираз придатний для подальших розрахунків. Позначимо літерою *i* номер точки, тоді:

$$D_i(x)=y_i\cdot\frac{(x-x_0)(x-x_1)\dots(x-x_{i-1})(x-x_{i+1})\dots(x-x_{n-1})(x-x_n)}{(x_i-x_0)(x_i-x_1)\dots(x_i-x_{i-1})(x_i-x_{i+1})\dots(x_i-x_{n-1})(x_i-x_n)} \quad . \qquad (2.3)$$

Як видно з виразу в чисельнику та знаменнику пропускаються множники за номером i. Завдяки цьому сам дріб при x_i приймає значення $y_i\cdot 1$, а при інших заданих x_k маємо нуль. Звісно сума

$$\sum_{i=o}^{n} D_i(x) \qquad (2.4)$$

при всіх $x=x_k$ буде рівна y_k. Тобто умова інтерполяції виконується.

Приклад:

Використовуючи умову прикладу з пункту 2.1.1 провести інтерполяцію методом Лагранжа.

Розв’язання:

За формулами Лагранжа 2.3 та 2.4 запишемо:

$$y(x)=y_0\frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)}+y_1\frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)}+$$
$$+y_2\frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)}+y_3\frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)},$$

або переходячи до конкретних значень

$$y(x)=2\frac{(x-2)(x-3)(x-4)}{(1-2)(1-3)(1-4)}+7\frac{(x-1)(x-3)(x-4)}{(2-1)(2-3)(2-4)}+$$
$$+14\frac{(x-1)(x-2)(x-4)}{(3-1)(3-2)(3-4)}+23\frac{(x-1)(x-2)(x-3)}{(4-1)(4-2)(4-3)},$$

$$y(x)=\frac{x^3-9x^2+26x-24}{-3}+7\frac{x^3-8x^2+19x-12}{2}+$$
$$+14\frac{x^3-7x^2+14x-8}{-2}+23\frac{x^3-6x^2+11x-6}{6},$$

$$y(x)=x^2+2x-1 \quad .$$

Відповідь: $y(x)=x^2+2x-1$.

2.1.3. Інтерполяція многочленом Ньютона можлива, коли задані точки розбивають проміжок на якому проводиться інтерполяція на рівні відрізки. Тоді всі x можна записати так:

$$x_i=x_0+i\cdot h, \text{ де } h=x_{i+1}-x_i.$$

Сам інтерполяційний многочлен шукатимемо у вигляді:

$$P_n(x)=b_0+b_1(x-x_0)+b_2(x-x_0)(x-x_1)+\ldots+b_n(x-x_0)(x-x_1)\cdot\ldots\cdot(x-x_{n-2})(x-x_{n-1}). \qquad (2.5)$$

Тепер знайдемо невідомі коефіцієнти b_i. Для початку візьмемо $x=x_0$ і підставимо його в рівняння 2.5:

$$y_0=b_0.$$

Всі інші доданки рівні нулю, бо до них входить множник $(x_0-x_0)=0$. Тепер приймемо що $x=x_1$:

$$y_1=b_0+b_1(x_1-x_0)+0+0+\ldots=b_0+b_1h,$$
$$b_1=(y_1-y_0)/h=\Delta y_0/h.$$

Таким же чином знайдемо b_2:

$$b_2=(y_2-2y_1+y_0)/(2h^2),$$
$$b_2=\Delta^2 y_0/(2h^2).$$

Тут Δ позначає різницю (дискретний еквівалент похідної). Загальний вираз для пошуку значення коефіцієнтів записується так:

$$b_k=(\Delta^k y_0)/(k!h^k),$$

і загальний запис інтерполяційного многочлену:

$$P_n(x)=\sum_{k=0}^{n}\frac{\Delta^k y_0}{k!h^k}(x-x_0)(x-x_1)\dots(x-x_{k-1}) \quad . \tag{2.6}$$

Приклад:

Точки задані в таблиці. Провести інтерполяцію степеневим многочленом Ньютона.

x	1	2	3	4
y	2	7	14	23

Розв'язання:

В даній таблиці x_{i+1}-x_i завжди рівне 1, тому h=1. Знаходження різниць для зручності запишемо в таблицю:

y	**2**	*7*	*14*	*23*
$\Delta y_i=(y_{i+1}-y_i)/h$	**7-2=5**	*14-7=7*	*23-14=9*	--
$\Delta^2 y_i=(\Delta y_{i+1}-\Delta y_i)/h$	**7-5=2**	*9-7=2*	--	--
$\Delta^3 y_i=(\Delta^2 y_{i+1}-\Delta^2 y_i)/h$	**2-2=0**	--	--	--

В результаті отримаємо многочлен:

$$f(x)=\frac{2}{0!1^0}+\frac{5}{1!1^1}(x-1)+\frac{2}{2!1^2}(x-1)(x-2)+\frac{0}{3!1^2}(x-1)(x-2)(x-3),$$

$$f(x)=2+5(x-1)+1(x-1)(x-2),$$

$$f(x)=2+5x-5+x^2-3x+2,$$

$$f(x)=x^2+2x-1.$$

Відповіді співпадають у всіх трьох прикладах. Це вказує на єдині основи пошуку степеневого многочлена. Також можна зробити **висновок**:

Якщо всі різниці n-го порядку рівні, то дана послідовність описується єдиним многочленом n-го степеню.

2.2. Інтерполяція многочленами інших типів

2.2.1. Використання тригонометричних функцій при інтерполяції теж можливо і, в більшості ситуаціях, дає значно кращі наближення коливних функцій. В даному випадку припустимо що основні циклічні частоти нам вже відомі $\omega_1, \omega_2, \ldots, \omega_{n-1}, \omega_n$. Тоді інтерполяційний многочлен запишеться так:

$$f(x)\approx a_0+\sum_{k=1}^{n}\left(a_k\cos(\omega_k x)+b_k\sin(\omega_k x)\right) \quad . \tag{2.7}$$

В цьому многочлені маємо $n\cdot 2+1$ коефіцієнтів, тому для однозначного їх визначення повинно бути стільки ж рівнянь в системі та точок в таблиці точок для інтерполяції. Аналогічно пп. 2.1.1. запишемо систему рівнянь:

$$\begin{cases} y_0=a_0+\sum_{k=1}^{n}\left(a_k\cos(\omega_k x_0)+b_k\sin(\omega_k x_0)\right), \\ y_1=a_0+\sum_{k=1}^{n}\left(a_k\cos(\omega_k x_1)+b_k\sin(\omega_k x_1)\right), \\ \ldots \\ y_{2n+1}=a_0+\sum_{k=1}^{n}\left(a_k\cos(\omega_k x_{2n+1})+b_k\sin(\omega_k x_{2n+1})\right). \end{cases} \tag{2.8}$$

Ця система є лінійною, тому розв'язується одним з стандартних методів для систем лінійних рівнянь. Але більш зручними тут будуть інші методи, які будуть показані нижче, як і оцінка похибки такого інтерполяційного многочлену.

2.2.2. Використання гіперболічних функцій при інтерполяції дає значно краще наближення для швидкозмінних функцій. Тут висновки повністю ідентичні попередньому пункту, лише вирази 2.7 та 2.8 перепишуться таким чином:

$$f(x) \approx a_0 + \sum_{k=1}^{n} \left(a_k \operatorname{ch}(\omega_k x) + b_k \operatorname{sh}(\omega_k x) \right) \quad , \tag{2.9}$$

$$\begin{cases} y_0 = a_0 + \sum_{k=1}^{n} \left(a_k \operatorname{ch}(\omega_k x_0) + b_k \operatorname{sh}(\omega_k x_0) \right), \\ y_1 = a_0 + \sum_{k=1}^{n} \left(a_k \operatorname{ch}(\omega_k x_1) + b_k \operatorname{sh}(\omega_k x_1) \right), \\ \dots \\ y_{2n+1} = a_0 + \sum_{k=1}^{n} \left(a_k \operatorname{ch}(\omega_k x_{2n+1}) + b_k \operatorname{sh}(\omega_k x_{2n+1}) \right). \end{cases} \tag{2.10}$$

Тут $\operatorname{ch}(x) = \frac{e^x + e^{-x}}{2}, \operatorname{sh}(x) = \frac{e^x - e^{-x}}{2}$ - гіперболічні косинус та синус.

2.2.3. Інтерполяційний многочлен загального вигляду складається з лінійної комбінації довільних відомих функцій:

$$f_0(x), f_1(x), \dots, f_k(x), \dots, f_n(x) \quad ,$$

$$y(x) \approx \sum_{k=0}^{n} a_k \cdot f_k(x) \quad .$$

Тут коефіцієнти визначаються теж системою лінійних рівнянь 2.11, але в загальному випадку оцінити максимальну похибку інтерполяції дуже ускладнено.

$$\begin{cases} y_0 = a_0 f_0(x_0) + a_1 f_1(x_0) + \ldots + a_n f_n(x_0), \\ y_1 = a_0 f_0(x_1) + a_1 f_1(x_1) + \ldots + a_n f_n(x_1), \\ \ldots, \\ y_n = a_0 f_0(x_i) + a_1 f_1(x_i) + \ldots + a_n f_n(x_i). \end{cases} \quad (2.11)$$

Завдання для другого розділу

Написати програму для побудови інтерполяційного поліному (якщо остання цифра номера заліковки N парна, то поліном Лагранжа, інакше – Ньютона) за заданими точками (задати або константами або вводяться вручну, набір точок визначається за N з наступних таблиць). Додати можливість обчислити значення інтерполяційного поліному в будь-якій точці. Програма повинна мати функцію для отримання коефіцієнтів поліному та функції для обчислення значення поліному.

Додаткове завдання: побудувати на екрані базові точки та графік інтерполяційного поліному.

Таблиця 1

x	*y*	№ варианту
0,43	1,80866	1, 25
0,48	0,89492	2, 27
0,55	1,02964	11
0,62	2,20966	12
0,70	1,34087	17
0,75	1,52368	19

Таблиця 2

x	*y*	№ варианту
1,01	8,05421	3, 26
1,05	6,61659	4, 28
1,09	4,69170	10, (0)
1,13	3,35106	13
1,17	2,73951	18
1,20	2,36522	20

Таблиця 3

x	*y*	№ варианту
0,35	2,80866	5, 29
0,41	2,89492	6
0,47	1,02964	14
0,51	1,20966	16
0,56	1,34087	21
0,64	1,52368	23

Таблиця 4

x	*y*	№ варианту
0,11	9,07421	7, 30
0,15	6,62659	8
0,21	4,56170	9
0,29	3,37106	15
0,35	2,78951	22
0,40	2,45522	24

3. АПРОКСИМАЦІЯ ФУНКЦІЙ

Апроксимація (тобто наближення) функцій має дуже важливе значення в обчислювальній математиці, бо на цій задачі та на схожій задачі інтерполяції ґрунтується дуже багато інших методів обчислень. Задачу апроксимації можна сформулювати таким чином:

Підібрати комбінацію відомих функцій таким чином, щоб значення результуючої функції якнайменше відрізнялися від значень заданої функції (як правило вхідна функція задається таблично, але якщо потрібно просто спростити відому функцію, то вона може бути задана й аналітично чи якимось іншим чином).

Існує два види апроксимації *рівномірна* та *в середньому*. При рівномірній апроксимації проводиться наближення так, щоб максимальне відхилення від функції-еталону було найменшим (3.1), а при апроксимації в середньому, намагаються зробити мінімальним середнє відхилення (як приклад можна мінімізувати вираз 3.2 або вираз 3.3, що більш складно).

$$|f(x)-g(x)|=\min. \tag{3.1}$$

$$\sum_i (f(x_i)-g(x_i))^2=\min. \tag{3.2}$$

$$\sum_i |f(x_i)-g(x_i)|=\min. \tag{3.3}$$

Тут *f(x)* – функція-еталон, а *g(x)* – функція-наближення, $x\in[a;b]$.

3.1. Рівномірне наближення поліномами Бернштейна

3.1.1. Поліноми Бернштейна використовуються для рівномірної апроксимації функцій (критерій наближення визначається за умовою 3.1).

Поліноми Бернштейна наближають функцію лише на проміжку від нуля до одиниці використовуючи рівномірний поділ цього проміжку на *n* частин, але за допомогою підстановок легко домогтися апроксимації на довільному іншому проміжку, і мають такий вигляд:

$$B_n(f,x)=\sum_{k=0}^{n} f\left(\frac{k}{n}\right) C_n^k x^k (1-x)^{n-k} \quad . \tag{3.4}$$

Тут $f(x)$ – функція до якої шукаємо наближення, $C_n^k=\frac{n!}{k!(n-k)!}$ – біноміальний коефіцієнт.

Для демонстрації ефективності використання поліному проведемо наближення поліномом Бернштейна досить просту функцію $f(x)=\sin(4x)-1/2$. Для цього складемо програму для розрахунку виразу:

$$B_n(f,x)=\sum_{k=0}^{n} f(k/n)\cdot C_n^k x^k (1-x)^{(n-k)} \quad .$$

Приклад створення програми для обчислення напишимо на мові C++.

Для проведення розрахунків потрібно створити функцію для обчислення $f(x)=\sin(4x)-1/2$:

```
float f(float x)
{
        return sin(4.0*x)-0.5;
}
```

Функція є досить простою, тому зупинятися на пояснені не варто, а перейдемо до наступної функції розрахунку біноміального коефіцієнту:

```
float C(int k,int n)
{
        float C=1.0;
        float Z=1.0;
    int i;
    for(i=k+1;i<=n;i++) C*=i;
    for(i=2;i<=(n-k);i++) Z*=i;
    return C/Z;
}
```

Для пояснення роботи програми потрібно спростити формулу розрахунку біноміального коефіцієнту:

$$C_n^k = \frac{1 \cdot 2 \cdot \ldots \cdot n}{(1 \cdot 2 \cdot \ldots \cdot k) \cdot (1 \cdot 2 \cdot \ldots \cdot (n-k))} = \frac{(k+1) \cdot (k+2) \cdot \ldots \cdot n}{1 \cdot 2 \cdot \ldots \cdot (n-k)} \quad .$$

Завдяки такому скороченю дробу, зменшується кількість множень для кожного коефіцієнту на k в чисельнику та k множень в знаменнику. Також виключено множення на одиницю. Після цього можна написати функцію, яка рахує значення поліному Бернштейна:

```
float Bernshtane(float x, int n)
{
        float B=0.0;
    int k;
    for(k=0;k<=n;k++)
    {
      B+=C(k,n)*f(k/n)*pow(x,k)*pow(x,n-k);
    }
    return B;
}
```

При написанні функцій використовувалися стандартні функції модуля math.h.

3.1.2. Оцінка максимального відхилення R_n значення поліному Бернштейна степеня n на проміжку [0;1] можна використати нерівність:

$$|R_n| \le \frac{max_{0 \le x \le 1} |f''(x)|}{4n} \quad .$$

Для функції sin(4*x*) максимальний модуль похідної в дійсних числах є 16, тому максимальне відхилення наближення від функції не перевищить $16/(4n) = 4/n$. Також ця формула дає знання, що для зменшення максимального відхилення в двічі, потрібно збільшити n теж вдвічі.

3.2. Формула Тейлора

3.2.1. Наближення поліномом Тейлора здійснюється при справдженні припущення, що функція яка наближається має неперервні похідні в точці *a* до *n*+1 порядку. Нехай маємо поліном *n*-ного степеню:

$$f(x) \approx b_0 + b_1(x-a) + b_2(x-a)^2 + \ldots + b_{n-1}(x-a)^{n-1} + b_n(x-a) \quad . \qquad (3.5)$$

Спробуємо знайти b_0, при цьому приймемо, що $x=a$:

$$b_0 = f(a) \quad ,$$

тобто $b_0 = f(a)$. Візьмемо від обох частин рівняння 3.5 похідну по *x*:

$$f'(x) = b_1 + b_2(x-a) \cdot 2 + \ldots + b_{n-1}(x-a)^{n-2} \cdot (n-1) + b_n(x-a)^{n-1} \cdot n$$

Тепер другу похідну:

$$f''(x)=b_2+b_3(x-a)\cdot(1\cdot2\cdot3)+...+b_{n-1}(x-a)^{n-3}\cdot((n-2)(n-1))+b_n(x-a)^{n-2}\cdot((n-1)n) \quad .$$

Повторивши дії n раз, отримаємо для n-1 та n:

$$f^{(n-1)}(x)=b_{n-1}\cdot(1\cdot2\cdot\ldots\cdot(n-2)(n-1))+b_n(x-a)\cdot(1\cdot2\cdot\ldots\cdot(n-2)(n-1)) \quad ,$$

$$f^{(n)}(x)=b_n(1\cdot2\cdot3\cdot\ldots\cdot(n-1)\cdot n) \quad .$$

В результаті маємо просте співвідношення при умові, що $x=a$:

$$b_i=f^{(i)}(a)/i! \quad . \tag{3.6}$$

Тоді вираз 3.5, якщо врахувати співвідношення 3.6 ,перепишеться так:

$$f(x)\approx f(a)+f'(a)(x-a)+...+f^{(i)}(a)(x-a)^i/i!+...+f^{(n)}(a)(x-a)^n/n! \quad , \tag{3.7}$$

це і є знаменитою формулою Тейлора. Найцікавіше в ній є те, що знаючи “всі” похідні в одній точці функції, ми маємо можливість описати її на всьому проміжку визначення. Як приклад цього вражаючого факту, покажемо перетворення функції синусу в многочлен Тейлора.

Нехай маємо функцію $\sin(x)$ яку потрібно представити у вигляді многочлену. Проведемо перетворення за формулою Тейлора (3.7) прийнявши a=0. Тут маємо: $\sin'(x)=\cos(x)$, $\sin''(x)=\cos'(x)=-\sin(x)$, $\sin'''(x)=-\sin'(x)=-\cos(x)$ …, але при x=0: $\sin(0)=0$, $\sin'(0)=1$,

$\sin''(0)=0$, $\sin'''(0)=-1$, …, $\sin^{(2i)}(0)=(-1)^{i+1}$, та $\sin^{(2i+1)}(0)=0$. Таким чином остаточно з (3.7) маємо наступний вираз:

$$\sin(x)\approx x-\frac{x^3}{3!}+\frac{x^5}{5!}-\ldots+(-1)^{(2i)}\frac{x^{2i+1}}{(2i+1)!}+\ldots \quad .$$

При цьому *i* це ціле число, яке від нуля прямує до нескінченності. Для впевненості зазначимо, що в отриманий многочлен входять одночлени лише непарного степеня, що в сумі дає непарну функцію, якою і є функція синусу. Також, знаючи що факторіал зростає значно швидше ніж показникова функція легко показати, що доданки цього многочлену поступово прямують до нуля, а сума збігається при будь-якому обмеженому *x*.

3.2.2. Оцінку похибки наближення формулою Тейлора проведемо припустивши, що функція *f(x)* має *n*+1 неперервних похідних.

Запишемо рівність:

$$f(x)=f(a)+f'(a)(x-a)+\ldots+f^{(n)}(a)(x-a)^n/n!+R_n(x) \quad , \tag{3.8}$$

де $R_n(x)$ вважатимемо залишковим членом. Для його знаходження запишемо таке співвідношення:

$$\int_a^x f'(t)\,dt=f(x)-f(a) \quad ,$$

яке можна записати й так:

$$f(x)=f(a)+\int_a^x f'(t)\,dt \quad .$$

Тепер скористаємось формулою інтегрування за частинами:

$$u=f'(t);\, du=f''(t)dt;\, dv=dt;\, v=t-x.$$

$$f(x)=f(a)+f'(t)(t-x)\Big|_a^x+\int_a^x f''(t)(t-x)\,dt \quad ,$$

$$f(x)=f(a)+f'(a)(x-a)-\int_a^x f''(t)(t-x)\,dt \quad .$$

Повторимо інтегрування за частинами ще раз:

$$\begin{aligned} &u=f''(t);\\ &du=f'''(t)dt;\\ &dv=(t-x)dt;\\ &v=(t-x)^2/2. \end{aligned}$$

$$f(x)=f(a)+f'(a)(a-x)-f''(t)(t-x)^2/2\Big|_a^x+\int_a^x f'''(t)(t-x)^2/2\,dt \quad ,$$

$$f(x)=f(a)+f'(a)(a-x)+f''(a)(x-a)^2/2+\int_a^x f'''(t)(t-x)^2/2dt \quad .$$

І продовжуючи такі перетворення n раз остаточно отримаємо:

$$\begin{aligned} &f(x)=f(a)+f'(a)(x-a)+\frac{1}{2}f''(a)(x-a)^2+\dots\\ &\dots+\frac{1}{n!}f^{(n)}(a)(x-a)^n+\frac{(-1)^{n+1}}{n!}\int_a^x f^{(n+1)}(t)(t-x)^n\,dt. \end{aligned}$$

З останньої рівності, ми отримали формулу Тейлора плюс залишок-поправка, яка називається остаточним членом формули Тейлора:

$$R_n=\frac{(-1)^{n+1}}{n!}\int_a^x f^{(n+1)}(t)(t-x)^n\,dt \quad .$$

Для оцінки похибки наближення замість похідної в остаточному члені підставимо число M, таке щоб $|f^{(n+1)}(t)|\le M$ (тут $t\in[a;x]$). Тоді

$$|R_n| \le \frac{M}{n!} \left| \int_a^x (t-x)^n \, dt \right| \quad . \tag{3.9}$$

Приклад:

Для функції $f(x)=x\cdot\cos(x)$ побудувати наближені поліноми за формулою Тейлора другого та четвертого степенів, a взяти рівним 0. Оцінити похибку для поліному четвертого степеня, якщо модуль x не перевищує 1. Побудувати графіки наближень та оригінальної функції.

Розв'язання:

Спочатку знайдемо похідні до п'ятого порядку та їх значення при $x=a$:

$$f'(x) = -x\cdot\sin(x) + \cos(x),$$

$$f''(x) = -x\cdot\cos(x) - 2\sin(x),$$

$$f'''(x) = x\cdot\sin(x) - 3\cos(x),$$

$$f^{(4)}(x) = x\cdot\cos(x) + 4\sin(x),$$

$$f^{(5)}(x) = -x\cdot\sin(x) + 5\cos(x).$$

$$f(0)=0, \qquad f'(0)=1, \qquad f''(0)=0, \qquad f'''(0)=-3, \qquad f^{(4)}(0)=0.$$

За виразом (3.7) маємо:

$$f(x) \approx f(0) + f'(0)(x-0) + f''(0)(x-0)^2/2 \quad ,$$

$$f(x) \approx f(0) + f'(0)(x-0) + f''(0)(x-0)^2/2 + f'''(0)(x-0)^3/6 + f^{(4)}(0)(x-0)^4/24 \quad .$$

Підставляючи конкретні значення:

$$f(x) \approx 0 + 1\cdot x + 0\cdot x^2/2 = x \quad ,$$

$$f(x)\approx 0+1\cdot x+0\cdot x^2/2-3\cdot x^3/6+0\cdot x^4/24=x-x^3/2 \quad .$$

Для оцінки похибки оцінимо п'яту похідну. П'ята похідна при $x\in[-1;1]$ не перевищуватиме 6, тому при оцінці похибки M візьмемо рівним 6, і за виразом (3.9) маємо:

$$|R_4|\le\frac{6}{24}\left|\int_0^1 (t-1)^4\,dt\right|=\frac{1}{4}\left|\frac{(t-1)^5}{5}\right|_0^1|=\frac{1}{20} \quad .$$

Графіки можна побудувати на ЕОМ використовуючи будь-який математичний пакет або електронну таблицю EXCEL. Графіки приведені на рис. 3.1.

Створимо функцію для розрахунку наближеного значення f(x)=cos(x) за допомогою многочлену Тейлора степеню *n*:

$$\cos(x)=1-\frac{x^2}{2!}+\frac{x^4}{4!}-\ldots+(-1)^n\frac{x^{2n}}{2n!} \quad .$$

```
float Tcos(float x, int n){
  int i;
  float t,s; //доданок ряду
  t=s=1.0;   //початкові значення
  for(i=1;i<=n;i++) //цикл рахування наступного
                    //доданку ряду
  {
    t*=-x*x/((2*i)*(2*i-1)); //перерахунок доданку
    s+=t;   //додаємо до кучі новий доданок
  }
  return s; //повертаємо значення суми
}
```

Функція побудована так, щоб не виникало потреби рахувати значення факторіалів, бо складові многочлену мають властивість $p_{i+1}=-p_i\, x^2/(2i(2i+1))$. Тому змінна t містить поточне значення доданку послідовності, і на кожному кроці циклу шукається наступне значення зі значно меншою кількістю операцій. Справедливість співвідношення доведіть самостійно.

Графік $f(x)=x\cdot\cos(x)$ та її наближення

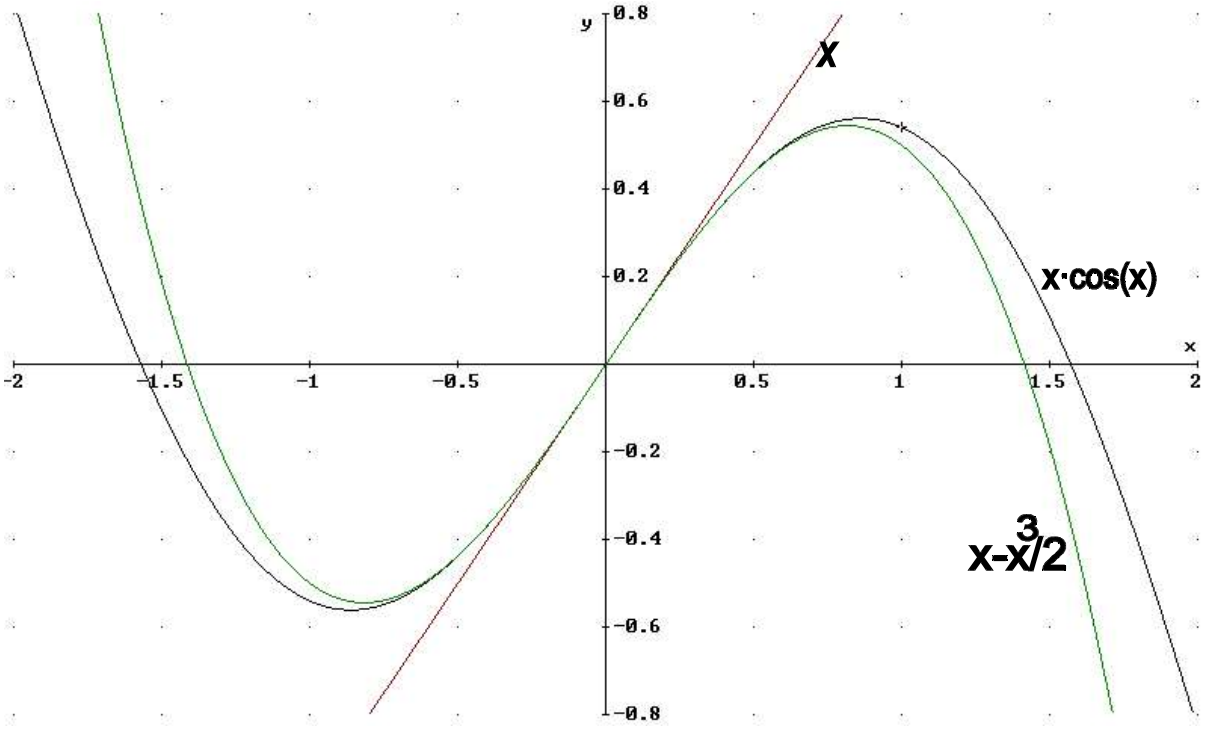

Рис. 3.1

3.3. Метод найменших квадратів

3.3.1. Загальний метод найменших квадратів використовується при необхідності наблизити до еталонної функції $f(x)$ лінійну комбінацію відомих функцій $f_0(x), f_1(x), \ldots, f_n(x)$, тобто знайти такі $a_0, a_1,\ldots, a_n$, щоб

$$S=\frac{\sum_{i=1}^{N}\left(f(x_i)-\left(a_0f_0(x_i)+a_1f_1(x_i)+\ldots+a_nf_n(x_i)\right)\right)^2}{N}=\min \quad . \tag{3.10}$$

Тут (x_i; $y_i=f(x_i)$) є наперед заданим набором еталонних точок; N – кількість таких точок. Звісно, можна відшукати мінімум по сумі модулів відхилень від еталонних значень, але при використанні квадратів знак похибки теж не впливає на загальну суму, і похибки більші за модулем впливають на результат значно сильніше, що зумовлює більш рівномірне наближення до еталонної функції.

Сума (3.10) фактично є завжди не від'ємною функцією від n+1 змінних, а її мінімум є екстремумом в якому всі часткові похідні по a_0, a_1, …, a_n рівні нулю. Для спрощення записів середнє арифметичне **позначатимемо рискою** над відповідними змінними. Тепер перетворимо вираз (3.10) і знайдемо всі часткові похідні:

$$S=\frac{1}{N}\left(\sum_{i=1}^{N}f^2(x_i)-2\sum_{i=1}^{N}f(x_i)\sum_{j=0}^{n}a_jf_j(x_i)+\sum_{i=1}^{N}\left(\sum_{j=0}^{n}a_jf_j(x_i)\right)^2\right) \quad ,$$

$$S=\frac{1}{N}\sum_{i=1}^{N}f^2(x_i)-2\frac{1}{N}\sum_{i=1}^{N}f(x_i)\sum_{j=0}^{n}a_jf_j(x_i)+\frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=0}^{n}a_jf_j(x_i)\right)^2 \quad ,$$

$$S=\overline{f^2(x)}-2\sum_{j=0}^{n}a_j\overline{f(x)f_j(x)}+\sum_{j=0}^{n}\sum_{k=0}^{n}a_ja_k\overline{f_j(x)f_k(x)} \quad .$$

Якщо прийняти $y_i=f(x_i)$, то отримаємо вираз придатний для пошуку часткових похідних:

$$S=y-2\sum_{j=0}^{n}a_j\overline{yf_j(x)}+\sum_{j=0}^{n}a_j^2\overline{f_j^2(x)}+2\sum_{j=0}^{n-1}\sum_{k=j+1}^{n}a_ja_k\overline{f_j(x)f_k(x)} \quad .$$

Похідна по змінній a_m буде рівна:

$$S'_{a_m} = 0 - 2\overline{yf_m(x)} + 2\sum_{j=0}^{n} a_j \overline{f_m(x) f_j(x)} \quad . \tag{3.11}$$

Для знаходження екстремуму прирівняємо похідні до нуля, після чого отримаємо систему лінійних рівнянь:

$$\begin{aligned}
&-\overline{yf_0(x)} + \sum_{j=0}^{n} a_j \overline{f_0(x) f_j(x)} = 0 \\
&-\overline{yf_1(x)} + \sum_{j=0}^{n} a_j \overline{f_1(x) f_j(x)} = 0 \\
&\ldots \\
&-\overline{yf_n(x)} + \sum_{j=0}^{n} a_j \overline{f_n(x) f_j(x)} = 0
\end{aligned}$$

або при розкритті знаків сум

$$\begin{aligned}
&a_0\overline{f_0^2(x)} + a_1\overline{f_0(x)f_1(x)} + \ldots + a_n\overline{f_0(x)f_n(x)} = \overline{y\,f_0(x)} \\
&a_0\overline{f_1(x)f_0(x)} + a_1\overline{f_1^2(x)} + \ldots + a_n\overline{f_1(x)f_n(x)} = \overline{y\,f_1(x)} \\
&\qquad\qquad \ldots \\
&a_0\overline{f_n(x)f_0(x)} + a_1\overline{f_n(x)f_1(x)} + \ldots + a_n\overline{f_n^2(x)} = \overline{y\,f_n(x)}
\end{aligned} \tag{3.12}$$

Відповідно, якщо система (3.12) має єдиний розв'язок то він є єдиним екстремумом-мінімумом (бо максимальна похибка звісно не обмежена).

3.3.2. Пошук найближчої прямої на основі методу найменших квадратів краще провести використовуючи систему (3.12). Для опису прямої необхідно мати дві функції $f_0(x)=1$, $f_1(x)=x$ лінійні комбінації яких дають рівняння прямих:

$$y = a_0 \cdot f_0(x) + a_1 \cdot f_1(x) = a_0 + a_1 \cdot x.$$

Отримаємо систему:

$$\begin{aligned}
&1 + \overline{x} = \overline{y}, \\
&\overline{x} + \overline{x^2} = \overline{xy}.
\end{aligned}$$

Розв'язавши систему в загальному випадку остаточно матимемо:

$$a_1=\frac{\overline{xy}-\bar{x}\cdot\bar{y}}{\overline{x^2}-\bar{x}.^2}\ ,$$

$$a_0=\bar{y}-a_1\cdot\bar{x}\ .$$

Тут $\bar{y}=\frac{1}{N}\sum_{i=1}^{N}y_i$, $\bar{x}=\frac{1}{N}\sum_{i=1}^{N}x_i$, $\overline{yx}=\frac{1}{N}\sum_{i=1}^{N}y_i x_i$, $\overline{x^2}=\frac{1}{N}\sum_{i=1}^{N}x_i^2$.

3.3.3. Наближення параболою методом найменших квадратів проводиться аналогічно, але тут необхідно взяти три функції: $f_0(x)=1$, $f_1(x)=x$, $f_2(x)=x^2$. За (3.12) маємо систему

$$\begin{aligned} a_0+a_1\bar{x}+a_2\overline{x^2}&=\bar{y},\\ a_0\bar{x}+a_1\overline{x^2}+a_2\overline{x^3}&=\overline{yx},\\ a_0\overline{x^2}+a_1\overline{x^3}+a_2\overline{x^4}&=\overline{yx^2}. \end{aligned} \qquad (3.13)$$

і маємо наступний вираз:

$$f(x)\approx a_0+a_1x+a_2x^2.$$

Тут $y=\frac{1}{N}\sum_{i=1}^{N}y_i$, $yx^c=\frac{1}{N}\sum_{i=1}^{N}y_i x_i^c$, $x^c=\frac{1}{N}\sum_{i=1}^{N}x_i^c$, де c – відповідний степінь ікс.

Приклади:

1) Даний масив точок. Побудувати рівняння прямої, яка має найменше середньоквадратичне відхилення. Побудувати графік та відмітити на ньому точки заданого масиву.

x	1,2	0,5	3,4	2,1	3,3	2,6	4,4	4,3	1
y	2,91	3,51	-1,51	0,99	-1,27	0,48	-2,93	-2,55	2,55

Розв’язання:

Знайдемо величини для розрахунку за виразом 3.12:

$\bar{x}$ = (1,2+0,5+3,4+2,1+3,3+2,6+4,4+4,3+1)/9=22,8/9≈2,53;

$\overline{x^2}$ =(1,2^2+0,5^2+3,4^2+2,1^2+3,3^2+2,6^2+4,4^2+4,3^2+1^2)/9≈8,24;

$\bar{y}$ =(2,91+3,51-1,51+0,99-1,27+0,48-2,93-2,55+2,55)/9≈0,24;

$\overline{yx}$ =(2,91·1,2+3,51·0,5-1,51·3,4+0,99·2,1-1,27·3,3+0,48·2,6-2,93·4,4-2,55·4,3 +2,55·1)/9≈-2,45.

a_1=(-2,45-2,53·0,24)/(8,24-2,53^2)≈-1,68;

a_0=4,78+7,91·2,53≈4,50.

Графік знайденої прямої та задані точки показані на рис. 3.2.

Експериментальні точки та їх наближення прямою

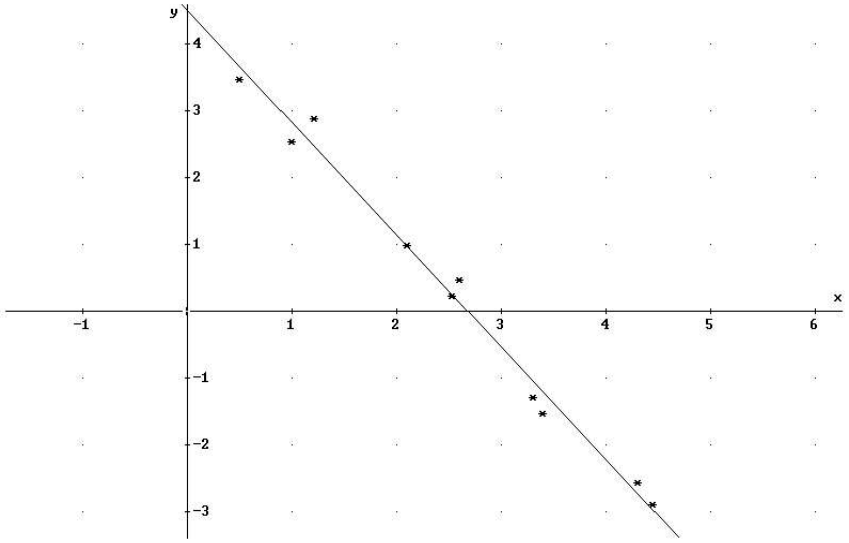

Рис.3.2

2) Вивести графіки функції $f(x)=\sin(e^x)$ та її наближення.

Розв'язання:

Для побудови графіків функції та її наближення необхідно мати функції для їх обчислення. Наведемо разрахунок значення функції:

```
float f(float x)
{
  return sin(exp(x));
}
```

Нехай нам потрібно побудувати наближення на проміжку [0;1] поліномом Тейлора третього степеня.

Для рахування значення поліному-наближення потрібно знати значення похідних:

$$f(x)=\sin(e^x)\ ;\ f'(x)=\cos(e^x)\cdot(e^x)'=e^x\cos(e^x)\ ;$$
$$f''(x)=(e^x)'\cos(e^x)+e^x(\cos(e^x))'=e^x(\cos(e^x)+(-\sin(e^x))(e^x)')=e^x(\cos(e^x)-e^x\sin(e^x))\ ;$$
$$f'''(x)=e^x(\cos(e^x)-e^x(3\sin(e^x)+e^x\cos(e^x)))\ .$$

Тепер знайдемо значення похідних при x=0.5:

$$f(0.5)\approx 0.996965\ ;\ f'(0.5)\approx -0.128346\ ;\ f''(x)\approx -2.83837\ ;$$
$$f'''(0.5)\approx -7.90956\ .$$

Тепер можна записати поліном Тейлора третього степеня:

$$T(x)=0.996965-0.128346\,x-2.83837\,x^2/2-7.90956\,x^3/6\ .$$

Мовою C++ розрахунок полінома заданого степеня буде таким:

```
float T(float x, int n)
{
  float s=0.0;
  if(n>=0)
  {
    s+=0.996965;
  }

  if(n>0)
  {
    s-=x*0.128346;
  }
```

```
    if(n>1)
    {
      s-=x*x*2.83837/2.0;
    }
    if(n>2)
    {
      s-=x*x*x*7.90956/6.0;
    }
    return s
}
```

Процедура містить декілька умовних операторів, це дозволяє рахувати поліном від першого до третього степеня, що корисно для демонстрації поступового покращення наближення.

Також для написання програми побудови графіків функції та її наближення потрібно вивести координатні вісі. Тут потрібно враховувати розмір графічного екрану в пікселях 600x440 і проводити маштабування. Нехай ми маємо вивести частину графіка коли x змінюється на проміжку $[a_0;a_1]$, а y лежить на проміжку $[b_0;b_1]$. Тоді на проміжок довжиною a_1-a_0 припадає 600 точок на екрані монітора, а на b_1-b_0 припадає 440 точок. Знаючи це, можна перерахувати координити точки на площині в екранні координати за властивостями пропорції:

$$x_e=(x-a_0)\frac{600}{a_1-a_0} \quad ; \quad y_e=(y-b_0)\frac{440}{b_1-b_0} \quad .$$

Для отримання з екранних координат реальні, перетворимо попередні рівняння:

$$x=x_e\frac{a_1-a_0}{600}+a_0 \quad ; \quad y=y_e\frac{b_1-b_0}{440}+b_0 \quad .$$

Можна писати процедуру виводу графіка на екран:

```
void GraphicPlot(int n) //приймає кількість точок
{
  float a0=-0.1;
  float a1= 1.1;
  float b0=-1.5;
  float b1= 1.5; //границі виводу графіка
  float y,x=a0; //початкове значення x
  float h=(a1-a0)/n; //крок по x
  int xe,ye,i; //екранні координати, лічильник
  y=f(x); //початкове значення y
  xe=20; //початкова координата екрану по x
  ye=240-(int)((y-b0)*440/(b1-b0));
     //початкова координати екрану по y
  setcolor(10); // графік буде зеленим
  moveto(xe,ye); //поставимо олівець в
                 //початкову точку
  for(i=1;i<=n;i++)
  { //для заданої кількості точок n...
    x=a0+h*i; //рахуємо наступний x
    xe=(int)((x-a0)*600/(a1-a0));
        //рахуємо настепний екранний x
    y=f(x); //рахуємо y
    ye=240-(int)((y-b0)*440/(b1-b0));
        //рахуємо екранний y
    lineto(xe,ye); //продовжуємо графік
  }
}
```

Особливістю розрахунку вертикальної екранної координати є різниця 240-y_e. Це пояснюється тим, що екранні координати направлені вниз і для переходу до зручного режиму знак розрахованої координати змінюється на протилежний; також з причини розташування початку координат в горі екрану, відбувається зміщення на 240 точок вниз:

```
ye=240-(int)((y-b0)*440/(b1-b0)).
```

Завдання до третього розділу

1. За останьою цифрою номеру в списку журналу визначте свою функцію, до якої будете будувати наближення:

Номер варіанту	$f(x)$	Номер варіанту	$f(x)$
1	$f(x)=\cos(4x)-x+1$	2	$f(x)=e^{-0.98x}+x^2$
3	$f(x)=\sqrt{\cos(3x-1)+x}$	4	$f(x)=\sin^{1/3}(4x)$
5	$f(x)=\ln(x+1)\cdot 3-\sin(x)$	6	$f(x)=x^2\cdot\sin(x)$
7	$f(x)=\sin(\cos(5x))$	8	$f(x)=2\sin(\sqrt{5x})$
9	$f(x)=e^{\sin(5x)}$	0	$f(x)=e^{\cos(4x)}$

Для своєї функції напишіть програму для розрахунку наближення

а) поліномом Бернштейна 3 та 6 степенів;

б) знайдіть перші три похідних своєї функції та розрахуйте значення за многочленом Тейлора;

в) побудуйте на одному малюнку графік функції та її наближень.

2. Для своєї функції з завдання 1 побудуйте таблицю:

i	0	1	2	3	4	5	6	7	8	9
x_i	0.1	0.15	0.2	0.3	0.33	0.36	0.4	0.45	0.47	0.5
y_i	$f(x_0)$	$f(x_1)$	$f(x_2)$	$f(x_3)$	$f(x_4)$	$f(x_5)$	$f(x_6)$	$f(x_7)$	$f(x_8)$	$f(x_9)$

Для цих даних побудуйте наближення прямою та параболою методом найменших квадратів. Побудуйте на графіку точки та графіки наближень.

4. РОЗВ'ЯЗАННЯ РІВНЯНЬ З ОДНІЄЮ НЕВІДОМОЮ

В школі та вищих навчальних закладах на заняттях математики навчаються аналітично знаходити розв'язки рівнянь у вигляді однозначних виразів. На практиці ж рівняння, яке допускає аналітичне розв'язання зустрічається не часто, але розв'язувати такі рівняння все одно потрібно. Для розв'язання рівнянь розроблено багато чисельних методів.

Розв’язання рівнянь $f(x)=0$ починають з локалізації існуючих коренів, для цього людина повинна вказати комп’ютеру початковий інтервал, де будуть шукатися корені. Далі початковий проміжок розбивається на рівні частини (рис 4.1) на кінцях яких рахують значення $f(x)$. Для подальшої роботи відбирають проміжки, на кінцях яких значення $f(x)$ мають протилежні знаки. Звісно, якщо на етапі початкової локалізації коренів випадково потрапимо на один з коренів точно, то це лише позбавить нас від подальшого уточнення цього кореня.

Початкове розбиття інтервалу для пошуку коренів $\ln(x)-x+2=0$

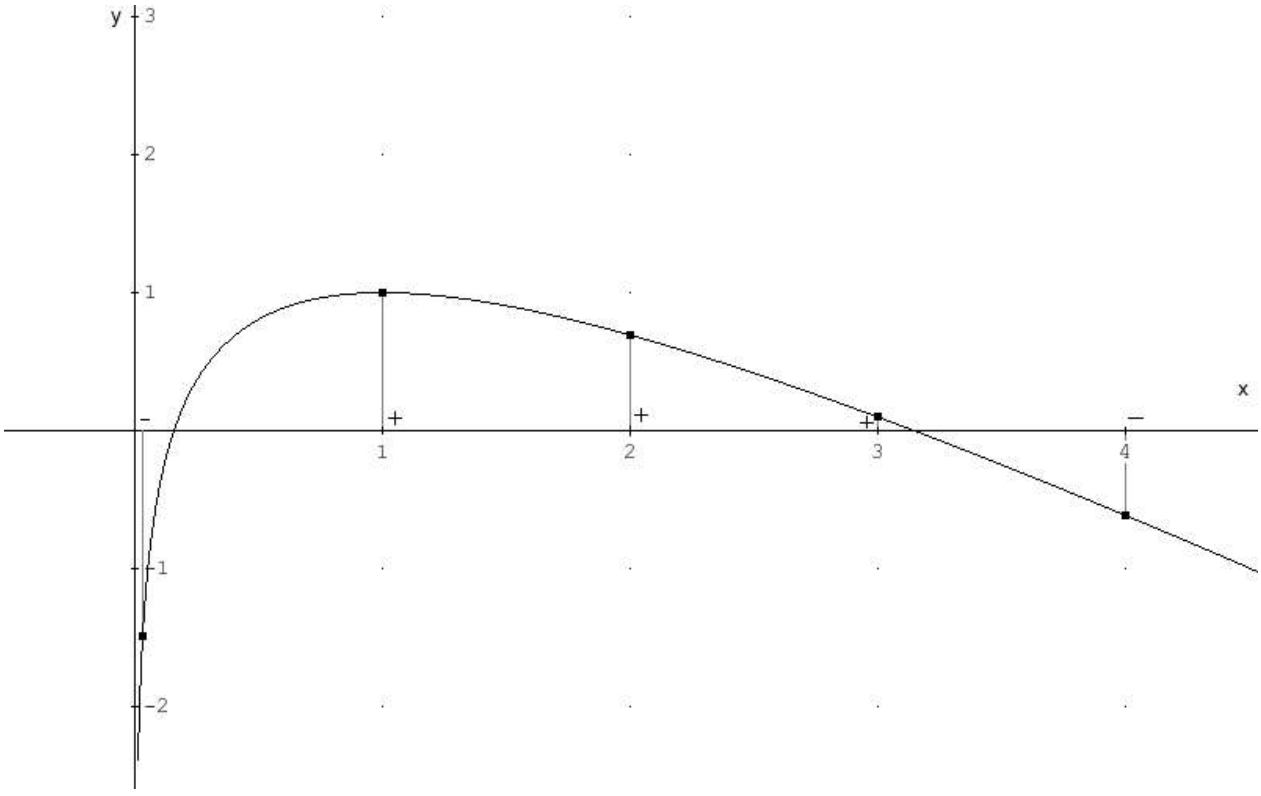

Рис. 4.1

На рисунку 4.1 показано початкове розбиття для виділення коренів функції $y(x)=\ln(x)-x+2$, і для подальшого уточнення коренів вибирають два інтервали (0,1;1) та (3;4). Іноді початкове розбиття з кроком в одиницю не дає змогу виділити інтервали з коренем, тоді проводять вибірку з меншим кроком, при потребі крок зменшують ще декілька разів, перед тим як повідомити, що коренів не знайдено (на жаль такий процес початкового виділення коренів не доводить їх відсутність на пробному проміжку).

Такий процес виділення коренів необхідно виконувати для наступних алгоритмів послідовного звуження інтервалу, та бажано проводити і для інших методів уточнення коренів.

4.1. Методи послідовного звуження інтервалу

4.1.1. Метод простої дихотомії (половинного поділу) є найпростішим методом, який при наявності локалізованого кореня дає гарантовано збіжну послідовність наближення до кореню рівняння.

Процес відшукання коренів методом дихотомії покажімо далі. Нехай нам відомий інтервал де локалізовано один з коренів $[a_0;b_0]$, при цьому $f(a_0)<0$ та $f(b_0)>0$. Тепер поділимо інтервал навпіл $c_0=(a_0+b_0)/2$ та визначемо знак $f(c_0)$. Тепе маємо три контрольних значення: $f(a_0)<0$; $f(c_0)$(> або <)0; $f(b_0)>0$ та два інтервали $[a_0;c_0]$, $[c_0;b_0]$. Один з цих інтервалів містить корінь рівняння. Звісно, той інтервал містить корінь, який на кінцях має різні знаки функції. Нехай $f(c_0)<0$, тоді корінь буде на інтервалі $[c_0;b_0]$, або при $f(c_0)>0$ — на інтервалі $[a_0;c_0]$. Новий інервал позначимо $[a_1;b_1]$ і він є вдвічі меншим, тому можна сказати, що корінь знаємо тепер з вдвічі меншою абсолютною похибкою: $x=c_0\pm|c_0-b_0|$.

Процес можна повторювати, на кожному кроці зменшучі абсолютну похибку вдвічі. Завдяки повній передбачувальності такого процесу можна

визначити похибку визначення кореню при відомій довжині початковаго інтервалу та відомій кількості операцій уточнення. Нехай початковий інтервал має довжину *d*, тоді за *n* операцій уточнення похибка зменшиться в 2^n раз: $\Delta x<d/2^{n+1}$.

Спробуємо скласти алгоритм для пошуку коренів функції. Для успішної роботи функція повинна приймати вказівник на функцію, корінь якої ми повинні відшукати, початковий інтервал та бажану точність розрахунків:

```
float KorinDihotomiya(float (*f)(float x),float a,
float b, float delta)
{
  int i,n; //лічильник та кількість циків уточненя
  float c; //середина пробного інтервалу
  n=log(fabs(b-a)/delta)/log(2);
  //тут визначається кількість ітерацій для
  //досягнення потрібної точносі
  // Δx=d/2^(n+1) → n+1=log2(d/Δx)
      for(i=0;i<n;i++)//Повторюємо задану кількість
уточнень
  {
    c=(a+b)*0.5;//Шукаємо середину інтервалу
     if( f(c)>0.0 )//перевірка знаку функції в середині
інтервалу
    {//Звуження зправа
      b=c;
    }else
    { //Звуження інтервалу зліва
      a=c;
    }
  }
```

```
    return (a+b)*0.5; //повернемо як корінь середину останього інтервалу
}
```

Алгоритм простої дихотомії представлений мовою С++ можна переписати як і блок-схемами, так і іншими мовами програмування на вибір програміста.

Проаналізуємо роботу алгоритму на прикладі розв'язання нелінійного рівняння $\cos(x)-x=0$. Для початкового виділення кореня спробуємо значення $x=0$ та $x=1$: $\cos(0)-0>0$ та $\cos(1)-1<0$; тому $a=1$, $b=0$. Тепер можна перейти до звуження інтервалу. Серединою є $c=(a+b)/2=0.5$, значення функції є $\cos(0.5)-0.5\approx 0.3776>0$, що означає зміну $b=0.5$. Тепер маємо наступний інтервал [0.5;1], середина якоко є $c=(a+b)/2=(0.5+1)/2=0.75$. Значення функції в середені цього інтервалу є $\cos(0.75)-0.75\approx -0.0183<0$. Тепер маємо нове значення $a=0.75$ з інтервалом, де розташовано корінь [0.5;0.75]. Процес повторюється до звуженя інтервалу, який забезпечить потрібну точність. У цьому випадку коренем є x=0.625±0.125.

4.1.2. Метод поділу на третини та золотої пропорції алгоритмично не відрізняється від половинного ділення, лише замість половинного ділення проміжку використовується ділення з іншими пропорціями при розрахунку c. Наприклад для методу третин $c=a+(b-a)/3$, а для золотої пропорції $c=a+(b-a)/\alpha$, де $\alpha=(1+\sqrt{5})/2\approx 1.618$ є коефіцієнтом золотого січення. Перевагами цих методик є ймовірність більш швидкого звуження інтервалу з коренем, але недолік пов'язаний з цією перевагою — також є ймовірність менш швидкого звуження інтервалу нід методу половинного ділення. Також загалом не можна передбачити швидкіть звуження інтервалу з коренем, тому потрібно слідкувати за шириною проміжка на кожному кроці звуження. Функція пошуку коренів цими методами матиме вигляд:

```
float KorFind(float (*f)(float x),float a, float b,
float delta)
{
  int n=0; //кількість циків уточненя
  float c; //третина/частина пробного інтервалу
  do//Повторюємо звуження інтервалу
  {
    c=a+(b-a)/3;//Шукаємо третину інтервалу
    //c=a+(b-a)*2.0/(1+sqrt(5)); //Для зол.проп.
    if( f(c)>0.0 )//перевірка знаку функції в середині
інтервалу
    {//Звуження зправа
      b=c;
    }else
    { //Звуження інтервалу зліва
      a=c;
    }
    n++; //рахуємо кількість уточнень
  }wihle(fabs(b-a)>delta);//кожного циклу ширину
                          //інтервалу з коренем
                          //порівнюємо з допустимою
                          //похибкою
  return (a+b)*0.5; //повернемо як корінь середину
останього інтервалу
}
```

На цьому лістингу показано, що перевірку довжини інтервалу, в якому лежить корінь, потрібно повторювати після кожного звуження.

4.2. Методи послідовного наближення

Пошук коренів є досить ресурсоємною задачею, тому дослідники намагаються винайти засоби більш швидкого наближення до коренів і тут зменшення похибки вдвічі на кожному уточнені є поганим результатом. Особливо важливим є таке вдосконалення для ручних розрахунків та написання зверх продуктивних програмних продуктів.

4.2.1 Медод хорд уточнення коренів на інтервалі використовував ще Ньютон. Метод полягає на заміні функції на інтервалі пошуку кореня прямою, що дозволяє досить точно отримати наступне звуження інтервалу. Пояснення алгоритму уточнення кореню показано на рис. 4.2.

Пояснення до методу хорд

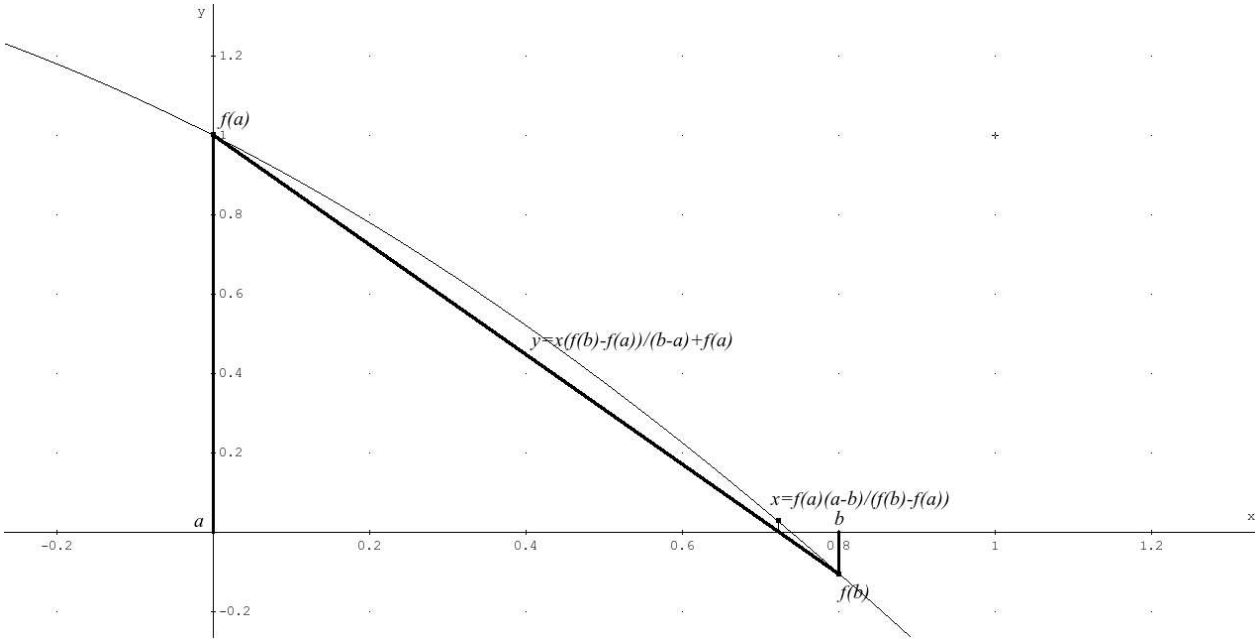

Рис. 4.2

На рисунку показано фрагмент графіку cos(*x*)-*x* з початковим виділенням коренів [0;0.8]. Замість визначення середньої точки інтервалу для подальшого уточнення, чи поділу в наперед зазначеної пропорції, точки на графіку *f*(0)=1 та *f*(0.8)=-0.103293 з'єднуються прямою. Пряма перетинає вісь OX при *x*=0.725105, *f*(0.725105)=0.023325>0. Шлях вибору границь

наступного інтервалу, де є корінь, є аналогічним методу дихотомії, і наступним інтервалом для подальшого пошуку кореня є [0.725105;0.8]. Довжина отриманого інтервалу порівняно з попереднім менша в 0.8/(0.8-0.72)≈10 раз, а метод половинного ділення зменшує проміжок лише у 2 рази. Звісно, приклад демонструє вдалу ситуацію, але й в загальному випадку метод хорд не гірший за метод половинного ділення.

Складемо алгоритм для реалізації мотоду хорд.

1. Маємо початковий інтервал [a;b], який містить в собі корінь $f(x)=0$; $f(a)<0$; $f(b)>0$.

2. Рахуємо значення c де хорда перетинає OX: $c=f(a)(a-b)/(f(b)-f(a))$.

3. Якщо $f(c)>0$, то $b:=c$, інакше $a:=c$.

4. Повторюємо пп. 2-3 до досягнення потрібної точності.

Цей алгоритм легко записати на довільній мові програмування, наприклад C++:

```
float FindHorda(float a,float b,float delta){
  float c; //уточнення
  int n; //кількість уточнень
  while(fabs(b-a)>delta)//Повторимо до досягнення
  //потрібної точності
  {
    c=f(a)*(a-b)/(f(b)-f(a));
    //Порахували точку перетину хорди
    if(f(c)>0) b=c; else a=c;
    //зменшили інтервал
    n++; //Збільшили кількість уточнень
  }
  //Повернемо як корінь останнє уточнення
  return f(a)*(a-b)/(f(b)-f(a));
}
```

Алгоритм легко вдосконалити таким чином, що на кожному уточнені значення функції, для якої шукають корінь, рахується лише один раз.

4.2.2. Метод дотичних виник вже при використанні диференціально інтегрального числення. Фактично, в околі кореня в точці *a* до графіку функції будується пряма $y=f'(a)x+f(a)$. Шукають корінь для прямої $x=-f(a)/f'(a)$, який повинен бути досить близьким до кореня самої функції. Знайдене значення ікса приймається за нове *a*, і уточнення проводиться повторно. Пояснення методу дотичних показано на рис.4.3.

До методу дотичних уточненя кореню

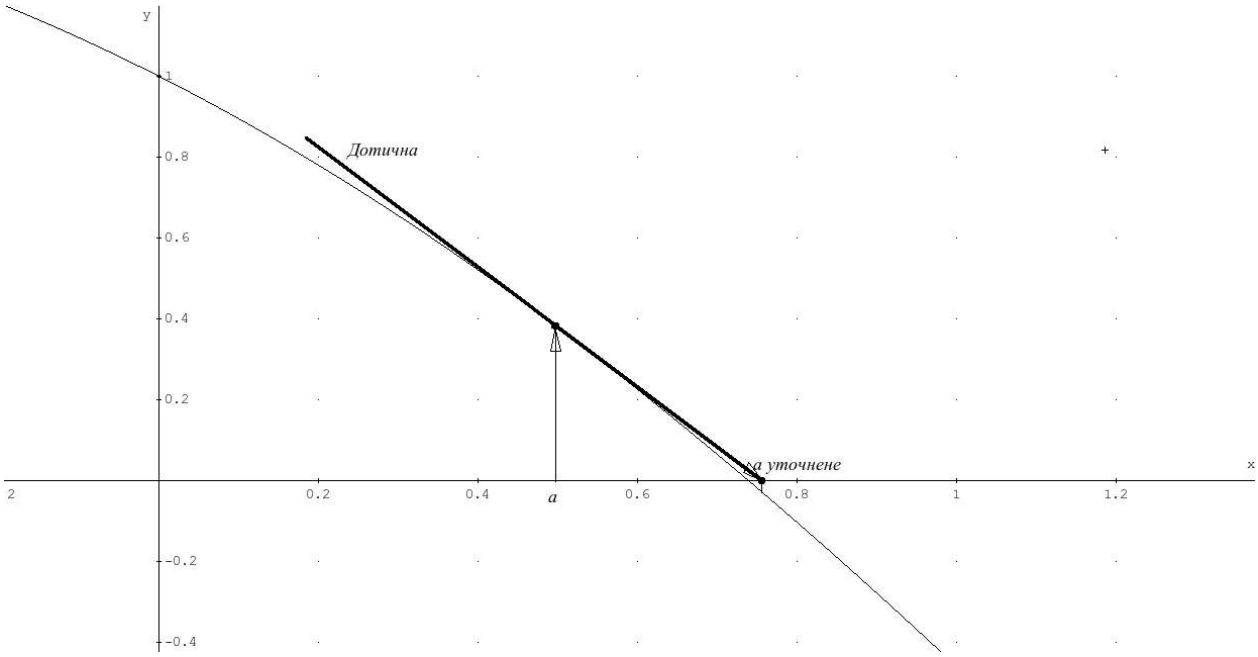

Рис. 4.3

```
float FindDotik(float x0, //початкове наближення
                float delta)//максимальна похибка
{
  float x; //уточнення
  int n; //кількість уточнень
  do{
    x0=x;
```

```
    x=-f(x0)/f(x0);
    //Порахували точку перетину дотичної
    n++; //Збільшили кількість уточнень
  }while(fabs(x0-x)>delta)//Повторимо до досягнення
  //потрібної точності
  return x;
}
```

Недоліком такого методу є вимога знання формули розрахунку не лише функції, але й її похідної, пошук якої досить важко автоматизувати. Також метод не є гарантовано збіжним. Може виникати випадок, коли лінійне наближення може нас не наближати, а навпаки віддаляти від дійсного значення кореня, і наближення проходить якісно й швидко при досить доброму початковому наближенні, яке можна отримати одним з попередніх методів. Перевагою є потреба початкового наближення не інтервалом, а одним початковим значенням.

4.3. Методи, які використовують особливості поведінки функції

Для більш швидкого відшукання коренів вже потрібно враховувати особливості самої функції, її коливність, експонентоціальність, логарифмічність та ін. Для прикладу можна розглянути аналог методу дотичної, але будувати не дотичну пряму, а дотичну функцію схожу за властивостями з досліджуваною функцією. В нашому випадку функція $f(x)=\ln(x)-x$ має логаривмичний характер, тому її краще наближати саме логарифмічною функцією $k_1\ln(x)+k_2$. Розв'язком функції-наближення є $x=e^{-k_2/k_1}$, потрібно лише знайти параметри k_1,k_2.

$$\begin{cases} k_1\ln(a)+k_2=f(a) \\ k_1/a=f'(a) \end{cases} \Rightarrow \begin{cases} k_2=f(a)-af'(a)\ln(a) \\ k_1=af'(a) \end{cases}.$$

Тоді наступне значення наближення є

$$a_{next} = e^{\frac{a f'(a)\ln(a) - f(a)}{a f'(a)}} .$$

Таким чином можна поступово наближатися до дійсного кореня $f(x)$.

Завдання до четвертого розділу

1. За останьою цифрою номеру в заліковій книжці визначте свою функцію, для якої будете розв'язувати рівняння $f(x)=0$:

Номер варіанту	$f(x)$
1	$f(x)=\cos(4x)-x+1$, $f'(x)=-4\sin(4x)-1$.
3	$f(x)=\sqrt{\cos(3x-1)+x}$, $f'(x)=\frac{3\sin(3x-1)+1}{2\sqrt{\cos(3x-1)+x}}$.
5	$f(x)=\ln(x+1)\cdot 3-\sin(x)$, $f'(x)=3/(x+1)-\cos(x)$.
7	$f(x)=\sin(\cos(5x)-3)$. $f'(x)=5\cos(\cos(5x)-3)\sin(5x)$.
9	$f(x)=e^{\sin(5x)}-x$, $f'(x)=5e^{\sin(5x)}\cos(5x)-1$.

Номер варіанту	$f(x)$
2	$f(x)=e^{-0.98x}-x^2$, $f'(x)=-0.98e^{-0.98x}-2x$.
4	$f(x)=\sin^3(4x)+x-1$, $f'(x)=14\sin^2(4x)\cos(4x)+1$.
6	$f(x)=x^2\cdot\sin(x)-1$, $f'(x)=2x\cdot\sin(x)+x^2\cos(x)$.
8	$f(x)=1-2\sin(\sqrt{5x})$, $f'(x)=-10\cos(\sqrt{5x})/(2\sqrt{5x})$.
0	$f(x)=x+e^{\cos(4x)}$, $f'(x)=1-4e^{\cos(4x)}\sin(4x)$.

Також визначте метод, за допомогою якого Ви повинні розв'язати рівняння, метод визначте за передостанньою цифрою залікової книжки:

Номер варіанту	метод
1	Метод половинного ділення
3	Метод поділу на третини
5	Метод хорд
7	Метод дотичних
9	Метод половинного ділення

Номер варіанту	метод
2	Метод поділу на третини
4	Метод хорд
6	Метод половинного ділення
8	Метод поділу на третини
0	Метод хорд

Напишіть програму для пошуку коренів Вашого варіанту завдання. Для розв'язання рівняння знайдіть початкове наближення до кореню функції за допомогою математичного пакету, або побудувати графік функції, або передбачити початковий відбір інтервалу з коренем у власній програмі.

Дайте письмово відповіді на питання:

1. Яку кількість уточнень використано у вашій програмі (в прикладах це число n)?

2. Чи вважаєте Ви, що інший метод кращий за використаний, чому?

3. Як можна вдосконалити Вашу програму (або використаний Вами метод) пошуку кореня?

4. Чи вдалося Вам знайти декілька коренів?

5. СИСТЕМИ ЛІНІЙНИХ РІВНЯНЬ

В практичних розрахунках найчастіше доводиться розв’язувати різноманітні системи лінійних рівнянь бо інші задачі часто зводяться саме до розв’язування системи лінійних рівнянь (див. попередні розділи). Тому важко переоцінити важливість точно та швидко розв’язати звичайну систему лінійних рівнянь. Розглянемо декілька загальних положень та означень.

Система лінійних рівнянь з n+1 невідомими в загальному випадку запишемо таким чином:

$$\begin{aligned} &a_{00}x_0+a_{01}x_1+\ldots+a_{0n}x_n=b_{0,}\\ &a_{10}x_0+a_{11}x_1+\ldots+a_{1n}x_n=b_{1,}\\ &\ldots\\ &a_{k0}x_0+a_{k1}x_1+\ldots+a_{kn}x_n=b_k. \end{aligned} \qquad (5.1)$$

В матричному вигляді ця система матиме вигляд

$$\begin{pmatrix} a_{00},a_{01},\ldots,a_{0n}\\ a_{10},a_{11},\ldots,a_{1n}\\ \ldots\\ a_{k0},a_{k1},\ldots,a_{kn} \end{pmatrix}\cdot\begin{pmatrix} x_0\\ x_1\\ \ldots\\ x_n \end{pmatrix}=\begin{pmatrix} b_0\\ b_1\\ \ldots\\ b_n \end{pmatrix}, \qquad (5.2)$$

$$A\cdot X=B.$$

В більшості випадків запис матрицями більш лаконічний і зрозуміліший. При перетворенні системи (віднімання рядків) та проведенні деяких оціночних викладок також використовують розширену матрицю системи яка в загальному випадку при $n=k$ записується так

$$\left(\begin{array}{c|c} a_{00},a_{01},\ldots,a_{0n} & b_0\\ a_{10},a_{11},\ldots,a_{1n} & b_1\\ \ldots & \ldots\\ a_{k0},a_{k1},\ldots,a_{kn} & b_n \end{array}\right). \qquad (5.3)$$

Відповідно матриця (5.3) матиме на стовпець більше ніж кількість стовпців аналогічної матриці з виразу (5.2).

Далі в тексті, якщо є система лінійних рівнянь то вважатимемо, що нам відомий будь-який з записів (5.1), (5.2), (5.3).

5.1. Умови єдиності та існування розв'язків

При розв'язуванні рівнянь необхідно спочатку визначити чи існують розв'язки взагалі і якщо існують, то цей розв'язок єдиний чи їх існує нескінченна кількість. Для отримання цих даних дуже зручно скористатися розширеною матрицею системи (5.3). Тут використовуються формулювання матричної алгебри. Сформулюємо ці ознаки:

1) Система **має єдиний розв'язок** тоді і лише тоді коли ранг розширеної матриці дорівнює рангу основної матриці A (5.2) та рівний кількості невідомих.
2) Система **не має розв'язків** тоді і лише тоді коли ранг розширеної матриці більший за кількість невідомих.
3) Система **має нескінченну кількість розв'язків** тоді і лише тоді коли ранг розширеної матриці менший за кількість невідомих.

5.2. Метод Крамера

Даний метод застосовний лише у випадку коли в системі 5.2 $n=k$. Даний метод досить простий описово, але при великій кількості змінних дуже швидко зростає кількість необхідних обчислень (пропорційно факторіалу від кількості змінних). Тому даний метод для ручного обчислення не варто застосовувати при розв'язуванні систем з кількістю змінних більше чотирьох.

Але, коли не вимагається висока швидкодія, з використанням ЕОМ можна розв’язувати досить великі системи.

Для пояснення процесу розрахунків потрібно розрахувати визначники:

$$\Delta=\begin{vmatrix} a_{00} & a_{01} & \dots & a_{0n} \\ a_{10} & a_{11} & \dots & a_{1n} \\ & \dots & \dots & \\ a_{n0} & a_{n1} & \dots & a_{nn} \end{vmatrix} .$$

$$\Delta_0=\begin{vmatrix} b_0 & a_{01} & \dots & a_{0n} \\ b_1 & a_{11} & \dots & a_{1n} \\ \dots & & & \\ b_n & a_{n1} & \dots & a_{nn} \end{vmatrix} ,$$

$$\Delta_1=\begin{vmatrix} a_{00} & b_0 & \dots & a_{0n} \\ a_{10} & b_1 & \dots & a_{1n} \\ & \dots & \dots & \\ a_{n0} & b_n & \dots & a_{nn} \end{vmatrix} ,$$

…

$$\Delta_n=\begin{vmatrix} a_{00} & a_{01} & \dots & b_0 \\ a_{10} & a_{11} & \dots & b_1 \\ & \dots & \dots & \\ a_{n0} & a_{n1} & \dots & b_n \end{vmatrix} .$$

Тобто в Δ_i i-тий стовпчик основної матриці замінюємо стовпчиком матриці-стовпця B. Отримавши n+1 значень визначників маємо:

$$x_i=\Delta_i/\Delta.$$

Звісно, якщо Δ=0 то значення невідомих отримати не можна, бо ділення на нуль заборонено. Тут умова Δ≠0 є еквівалентом умові існування єдиного розв’язку системи.

5.3. Метод підстановки

Метод підстановки є одним з найпростіших методів розв’язання систем рівнянь. Зміст цього методу пояснимо на прикладі розв’язання системи лінійних рівнянь з трьома невідомими:

$$\begin{aligned} 2x_0+3x_1+5x_2=2,\\ 4x_0+3x_1+2x_2=1,\\ x_0+x_1+2x_2=8. \end{aligned}$$

З першого рівняння системи знайдемо x_0:

$$x_0=2/2-3x_1/2-5x_2/2=1-1.5x_1-2.5x_2.$$

В наступних рівняннях системи замість вираженої змінної підставимо отриманий вираз:

$$\begin{aligned} 4(1-x_1\cdot 1.5-x_2\cdot 2.5)+3x_1+2x_2=1,\\ 1-x_1\cdot 1.5-x_2\cdot 2.5+x_1+2x_2=8. \end{aligned}$$

Звівши подібні:

$$\begin{aligned} -3x_1-8x_2=-3,\\ -0.5x_1-0.5x_2=7. \end{aligned}$$

Повторимо дії для x_1:

$$x_1=1-\frac{8}{3}x_2.$$

Після підстановки в останнє рівняння:

$$\begin{aligned}
&-0.5\left(1-\tfrac{8}{3}x_2\right)-0.5\,x_2=7,\\
&1-\tfrac{8}{3}x_2+x_2=-14,\\
&-\tfrac{5}{3}x_2=-15,\\
&x_2=9.
\end{aligned}$$

Тепер ми маємо можливість знайти x_1 (x_1=1-8·x_2/3=1-8·3=-23) та x_0 (x_0=1-1,5·x_1-2,5·x_2=1-1,5·9-2,5·(-23)=45).

Таким чином розв’язком системи є x_0=45, x_1=-23, x_2=9.

Аналогічно поступово позбавляючись від змінних розв’язуються системи з іншою кількістю невідомих.

5.4. Метод Гауса

Метод Гауса є фактично вдосконаленим методом підстановки. Тут також проводиться поступове виключення змінних з отриманням систем з меншою кількістю невідомих та обернений хід, коли з відомих даних добуваються значення ще невідомих змінних.

Для більш лаконічних записів використовуватимемо запис системи у вигляді розширеної матриці (4.3). Розв’яжемо попередню систему методом Гауса. Система запишеться так:

$$\left(\begin{array}{ccc|c} 2 & 3 & 5 & 2 \\ 4 & 3 & 2 & 1 \\ 1 & 1 & 2 & 8 \end{array}\right) \quad .$$

Кожне рівняння зберігає рівність при множенні його обох частин на одне й те саме число. Так як кожне рівняння відображається своєю строчкою в розширеній матриці то ці строчки можна множити на довільне відмінне від нуля число. В даному випадку нам потрібно отримати одиниці в першому стовпці. Для цього помножимо першу строку на ½, а другу строку на ¼:

$$\left(\begin{array}{cccc} 1 & 1.5 & 2.5 & 1 \\ 1 & 0.75 & 0.5 & 0.25 \\ 1 & 1 & 2 & 8 \end{array}\right) .$$

Відповідно маємо можливість віднімання рівнянь, що в розширеній матриці відповідає віднімання рядків. Віднімемо від другого та третього рядків перший:

$$\left(\begin{array}{cccc} 1 & 1.5 & 2.5 & 1 \\ 0 & -0.75 & -2 & -0.75 \\ 0 & -0.5 & -0.5 & 7 \end{array}\right) .$$

Аналогічно до попередніх дій:

$$\left(\begin{array}{cccc} 1 & 1.5 & 2.5 & 1 \\ 0 & 1 & 8/3 & 1 \\ 0 & 1 & 1 & -14 \end{array}\right) , \quad \left(\begin{array}{cccc} 1 & 1.5 & 2.5 & 1 \\ 0 & 1 & 8/3 & 1 \\ 0 & 0 & -5/3 & -15 \end{array}\right) , \quad \left(\begin{array}{cccc} 1 & 1.5 & 2.5 & 1 \\ 0 & 1 & 8/3 & 1 \\ 0 & 0 & 1 & 9 \end{array}\right) .$$

Перепишемо перетворену систему:

$$\begin{array}{c} x_0+1.5x_1+2.5x_2=1 \\ x_1+\frac{8}{3}x_2=1 \\ x_2=9 \end{array} , \quad \begin{array}{c} x_2=9 \\ x_1=1-\frac{8}{3}x_2 \\ x_0=1-1.5x_1-2.5x_2 \end{array} .$$

Тобто отримані вирази співпадають з виразами отриманими методом підстановки.

5.5. Матричний метод

Помножимо матриці (5.2):

$$\begin{pmatrix} a_{00}, a_{01}, \dots, a_{0n} \\ a_{10}, a_{11}, \dots, a_{1n} \\ \dots \\ a_{n0}, a_{n1}, \dots, a_{nn} \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ \dots \\ x_n \end{pmatrix} = \begin{pmatrix} a_{00}x_0 + a_{01}x_1 + \dots + a_{0n}x_n \\ a_{10}x_0 + a_{11}x_1 + \dots + a_{1n}x_n \\ \dots \\ a_{n0}x_0 + a_{n1}x_1 + \dots + a_{nn}x_n \end{pmatrix},$$

$$\begin{pmatrix} a_{00}x_0 + a_{01}x_1 + \dots + a_{0n}x_n \\ a_{10}x_0 + a_{11}x_1 + \dots + a_{1n}x_n \\ \dots \\ a_{n0}x_0 + a_{n1}x_1 + \dots + a_{nn}x_n \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ \dots \\ b_n \end{pmatrix}. \quad (5.4)$$

Очевидно, в даному випадку останній запис це є система лінійних рівнянь (5.1). Тому знайшовши невідому матрицю-стовпець одночасно знаходимо й розв’язок системи. Перетворимо систему (5.2) з використанням матричних позначень помноживши її обидві частини на обернену матрицю до матриці A:

$$A \cdot X = B,$$
$$A^{-1} \cdot A \cdot X = A^{-1} \cdot B,$$
$$EX = A^{-1} \cdot B,$$
$$X = A^{-1} \cdot B.$$

В результаті пошук розв’язків системи лінійних рівнянь зводиться до пошуку оберненої матриці до даної. Така матриця існує лише при умові, що визначник цієї матриці не дорівнює нулю, як і в методі Крамера це є умовою існування єдиного розв’язку системи.

Завдання до пя'того розділу

1. Для визначення свого завдання потрібно записати p, q – відповідно передостання та остання цифри номеру залікової книжки. Тоді система рівнянь для Вашого варіанту буде така:

$$\begin{aligned} p\,x_0 - 11\,x_1 + x_2 &= 5, \\ 2\,x_0 + x_1 + 3\,x_2 &= 4, \\ x_0 + 2\,x_1 + 3\,x_2 &= q. \end{aligned}$$

Написати програму для розв'язання системи лінійних рівнянь методами Крамера та Гауса. Порахувати наближено кількість операцій множення, ділення, додавання та віднімання. Показати більш економний на кількість операцій метод розв'язання. Якщо знаєте як, визначте час розв'язання системи рівнянь в циклі 10000 раз обома методами. Поясніть, як повинен змінитися час розрахунків, якщо кількість змінних у системі збільшити до чотирьох, п'яти.

6. МЕТОД ІТЕРАЦІЙ

В багатьох випадках, навіть при розв’язуванні рівнянь з однією змінною, аналітично виразити значення невідомої досить важко чи взагалі не можливо (наприклад розв’язуючи квадратне рівняння можна застосувати відомі формули для знаходження коренів, а от для рівняння $1/x=e^{-x}$ важко зразу визначити навіть кількість коренів). В цих випадках приходять на допомогу чисельні (наближені) методи пошуку коренів, які не можуть дати точні значення коренів рівняння, але дають можливість необмежено уточнювати їх. Одним з таких методів наближеного пошуку коренів є метод ітерацій.

6.1. Нелінійні рівняння з однією змінною

Для використання більшості наближених методів потрібно знати початкове наближення до розв’язку рівняння. Їх можна отримати підставивши пробні значення до рівняння і за зміною знаків результатів можна оцінити діапазон знаходження коренів.

Метод ітерацій вимагає представлення функції таким чином:

$$x=f(x).$$

Наприклад $x-\cos(x)=0$ перетворюється у $x=\cos(x)$.

Далі скористаємося позначенням x_i, що означатиме i-те наближення до кореня. Початкове значення x_0 вибирається як найближче до кореня, але це не критично. В нашому випадку початкове значення можна взяти $x_0=0$. Користуючись співвідношенням

$$x_{i+1}=f(x_i),$$

маємо:

$$x_1=\cos(x_0)=\cos(0)=1,$$
$$x_2=\cos(x_1)=\cos(1)\approx 0.5403,$$
$$x_3=\cos(x_2)=\cos(0.5403)\approx 0.8576,$$
$$\dots$$
$$x_9=\cos(x_8)=\cos(0.7221)\approx 0.7504,$$
$$\dots$$
$$x_{25}=\cos(x_{24})=\cos(0.7391)\approx 0.7391.$$

В даному випадку процес розрахунків дає певне число, що є коренем рівняння x-cos(x)=0. Але процес ітерацій не завжди збігається до певного числа, наприклад при вираженні x таким чином: x=arccos(x), x буде виходити за межі визначення оберненої тригонометричної функції. Для визначення критерію збіжності процесу потрібно визначити зміст ітерацій. Побудуємо графіки $y=x$ та y=cos(x) (рис. 6.1).

До пояснення методу ітерацій

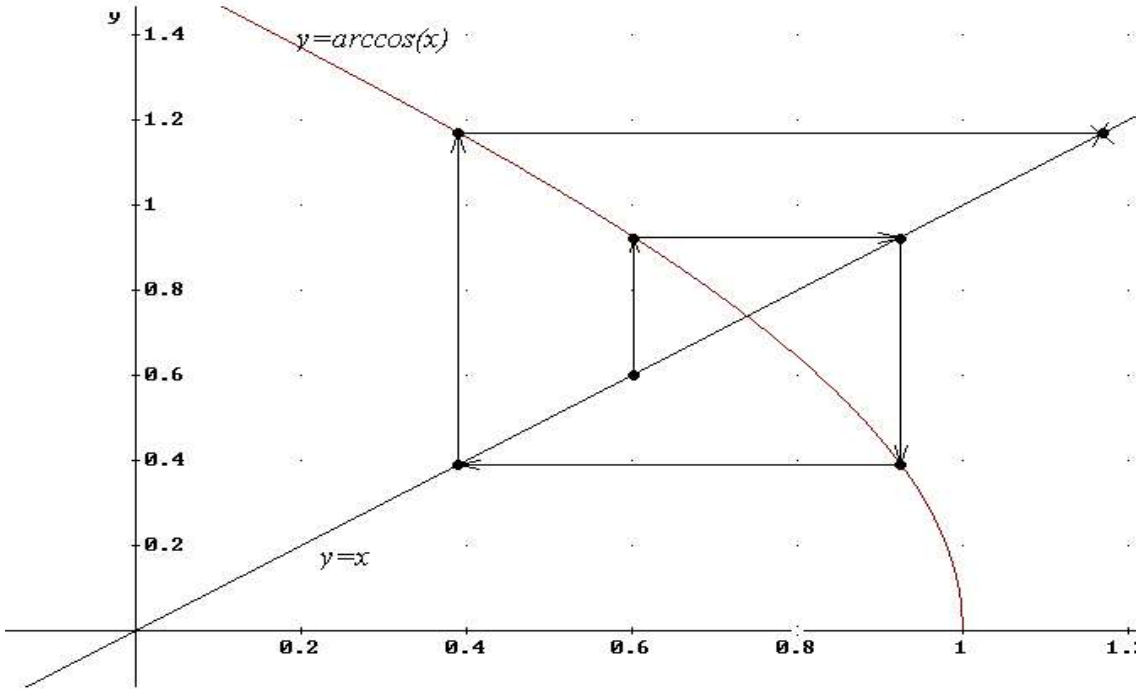

Рис. 6.1

До пояснення методу ітерацій

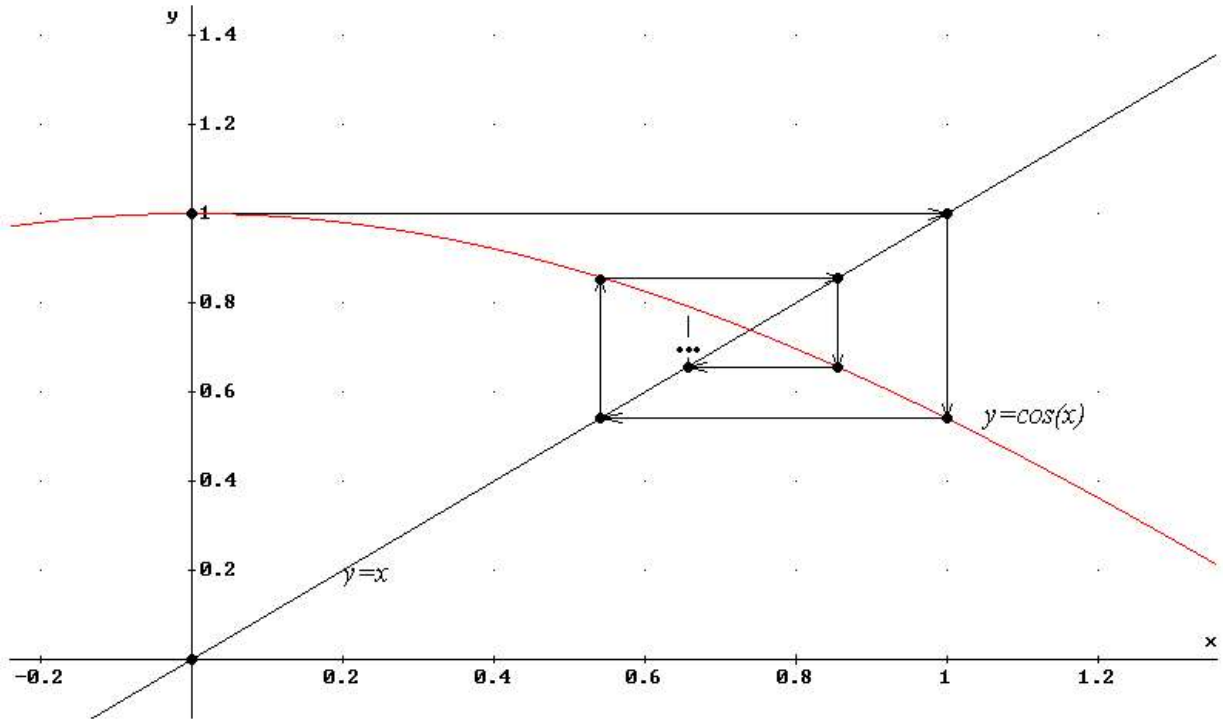

Рис. 6.2

З малюнка видно, при початковому наближенні x=0, піднімаємось до точки (0,1) графіку косинуса. Далі беручи нове x рухаємося по координатній площині вправо до перетину з y=x; продовжуючи таким чином серію підстановок рухаємось поступово наближаючись до точки перетину – кореня рівняння. На рис. 6.2 показаний випадок при якому процес ітерацій навпаки віддаляє від кореня. Зрозуміло, якщо кут нахилу (а відповідно й модуль похідної) прямої більший за кут нахилу при цьому ж аргументі кривої то ітерація наближатиме нас до кореня. За цим висновком можна визначити умову збіжності ітераційних процесів:

$$a=|f'(x)|<1. \qquad (6.1)$$

6.2. Система лінійних рівнянь

Метод простих ітерацій також можна застосовувати для системи лінійних та нелінійних рівнянь. Перепишемо систему (5.1):

$$\begin{cases} x_0 = a_{00}x_0 + a_{01}x_1 + \ldots + a_{0n}x_n - b_0, \\ x_1 = a_{10}x_0 + a_{11}x_1 + \ldots + a_{1n}x_n - b_1, \\ \ldots\ldots \\ x_n = a_{n0}x_0 + a_{n1}x_1 + \ldots + a_{nn}x_n - b_n. \end{cases}$$

Тепер, після вибору початкових наближень, пошук коренів проводиться за таким правилом:

$$\begin{cases} x_0^{next} = a_{00}x_0 + a_{01}x_1 + \ldots + a_{0n}x_n - b_0, \\ x_1^{next} = a_{10}x_0 + a_{11}x_1 + \ldots + a_{1n}x_n - b_1, \\ \ldots\ldots \\ x_n^{next} = a_{n0}x_0 + a_{n1}x_1 + \ldots + a_{nn}x_n - b_n, \end{cases} \qquad (6.2)$$

де x_i^{next} є наступним наближенням кореня. Для збіжності ітерацій при розв'язку системи повинно виконуватись умова

$$|a_{i0}| + |a_{i1}| + \ldots + |a_{in}| < 1, \quad i = 0..n. \qquad (6.3)$$

Приклад:

1) Задана система лінійних рівнянь з 3-ма невідомими. Визначити корені системи з точністю 1% ($\Delta x/x < 0{,}01$).

$$\begin{cases} x_1 = 0.2x_1 + 0.1x_2 + 0.15x_3 - 4 \\ x_2 = 0.7x_1 + 0.12x_2 + 0.05x_3 - 3 \\ x_3 = 0.2x_1 + 0.65x_2 + 0.11x_3 - 1 \end{cases}.$$

Розв'язання:

За початкові значення візьмемо $x_1 = x_2 = x_3 = 3$. Тоді провівши ітерацію

x_1=0.2·3+0.1·3+0.15·3-4=-2.65;

x_2=0.7·3+0.12·3+0.05·3-3=-0.39;

x_3=0.2·3+0.65·3+0.11·3-1=1.88.

Повторюємо ітерації:

x_1=-0.2·2.65-0,1·0.39 + 0.15·1.88-4=-4.29;

x_2=-0.7·2.65-0,12·0.39+0.25·1.88-3=-4.06;

x_3=-0.2·2.65-0,65·0.39+0.51·1.88-1=-1.58.

Продовжуємо ітерації доки значення коренів будуть змінюватись лише в четвертій значущій цифрі.

Остаточна відповідь після 20-ти ітерацій:

x_1=-8.36; x_2=-10.67; x_3=-10.79.

Вважаючи на кількість проведених ітерацій при виконанні практичних розрахунків рекомендується проводити розрахунки за допомогою математичних програмних пакетів (DERIVE, MATHCAD, MATHLAB, тощо), дуже зручно використати для застосування методу ітерацій електронні таблиці типу EXEL.

Кращі результати по швидкості отримання результатів дає **<u>метод Зейделя</u>**:

$$\begin{cases} x_0^{next} = a_{00}x_0 + a_{01}x_1 + \ldots + a_{0n}x_n - b_0, \\ x_1^{next} = a_{10}x_0^{next} + a_{11}x_1 + \ldots + a_{1n}x_n - b_1, \\ \ldots\ldots \\ x_n^{next} = a_{n0}x_0^{next} + a_{n1}x_1^{next} + \ldots + a_{nn}x_n - b_n, \end{cases}$$

Тут значення уточнення кореня використовується відразу після уточнення. В переважній більшості випадків цей метод дає змогу отримати корені значно швидше ніж ітераціями Ньютона. Для прикладу розв'яжемо попередній приклад методом Зейделя:

x_1=0.2·3+0.1·3+0.15·3-4=-2.65;
x_2=0.7·(-2.65)+0.12·3+0.05·3-3=-4.35;
x_3=0.2·(-2.65)+0.65·(-4.35)+0.11·3-1=-4.02.

Наступна ітерація дає:

x_1=0.2·(-2.65)+0.1·(-4.35) + 0.15·(-4.02)-4=-5.57;
x_2=0.7·(-5.57)+0.12·(-4.35)+0.05·(-4.02)-3=-7.62;
x_3=0.2·(-5.57)+0.65·(-7.62)+0.11·(-4.02)-1=-7.51.

Вже на 33-й ітерації отримується результат який в трьох знаках співпадає з попереднім методом ітерацій на 12-й ітерації. Той самий результат досягнуто майже вдвічі швидше.

6.3. Система нелінійних рівнянь

Цей метод є узагальненям методу ітерацій Ньютона та Зейделя, але замість системи лінійних рівнянь використовується запис:

$$\begin{cases} x_0 = f_0(x_0, x_1, \ldots, x_n) \\ x_1 = f_1(x_0, x_1, \ldots, x_n) \\ \ldots \\ x_n = f_n(x_0, x_1, \ldots, x_n) \end{cases}.$$

Процедура відшукання коренів повністю співпадає із розв'язанням системи лінійних рівнянь, але умовою збіжності тут є, що в околі уточнення початкового наближення коренів модулі похідних кожної функції менші за одиницю.

Приклад. Розв'яжемо таку систему нелінійних рівнянь:

$$\begin{cases} x_0 = \sin(x_0 + x_1 + x_2) \\ x_1 = \cos(x_0 \cdot x_1 \cdot x_2) \\ x_2 = (x_0 + x_1 + x_2)/4 \end{cases}.$$

В якості початкового наближення візьмемо $x_0 = x_1 = x_2 = 0.5$.

Уточнюємо x_0: x_0=sin(0.5+0.5+0.5)=sin(1.5)≈0.9975.

Уточнюємо x_1: x_1=cos(0.9975·0.5·0.5)≈cos(0.25)≈0.9975 .

Уточнюємо x_2: x_2=(0.9975+0.9975+0.5)/4≈0.6166.

Проводимо другу ітерацію:

Уточнюємо x_0: x_0=sin(0.9975+0.9975+0.6166)≈0.5298.

Уточнюємо x_1: x_1=cos(0.5298·0.9975·0.6166)≈0.9503 .

Уточнюємо x_2: x_2=(0.5298+0.9503+0.6166)/4≈0.5242.

Остаточний варіант маємо після 20-ти ітерацій:

x_0≈0.7730, x_1≈0.9204, x_2≈0.5645.

Недоліками такого методу є те, що іноді звести до ітераційного вигляду систему так, щоб виконувалася умова збіжності, не завжди легко.

Завдання до шостого розділу

1. За останьою цифрою номеру залікової книжки *p,* визначте свою функцію, для якої будете шукати корені методом ітерацій, тут *q* – передостання цифра номеру залікової книжки:

Номер варіанту	$0=f(x)$
1	$0=\cos(4x)-x+q$.
3	$0=\sqrt{\cos(3x-q)+x}$.
5	$0=\ln(x+1)-q\cdot\sin(x)$.
7	$0=\sin(\cos(5x)-qx)$.
9	$0=e^{\sin(5x)}-qx$.

Номер варіанту	$0=f(x)$
2	$0=e^{-0.98x}-x^{1/q}$.
4	$0=\sin(x/4)+x-q$.
6	$0=x^{q}\cdot\sin(x)-1$.
8	$0=2\mathrm{x}-\sin(q+\sqrt{x/5})$.
0	$0=x+e^{\cos(4x)}/(2+q)$.

Попередньо перетворіть вираз до виду *x*=... . Напишіть програму для розв'язання рівняння методом ітерацій. Якщо зможете знайдіть декілька коренів. Результати обчислення занесіть до звіту, зазначте кількість операцій для отримання відповіді із точністю 4-х вірних знаків (на наступній ітерації ці цифри не зміняться).

2. Розв’яжіть систему лінійних рівнянь методом ітерацій Ньютона та Зейделя. Система рівнянь для Вашого варіанту буде така:

$$\begin{cases}(1.1+p/10)x_0+0.1x_1-0.1x_2=5,\\ 0.2x_0+1.4x_1-0.2x_2=4,\\ 0.1x_0+0.3x_1+(1.1+q/10)x_2=3.\end{cases}$$

Для розв'язання напишіть програму та виконайте її. Визначте кількість ітерацій для отримання точності коренів до 4-х вірних знаків обома методами. Зазначте більш продуктивний метод для вашого варіанту завдання.

7. ЧИСЕЛЬНЕ ІНТЕГРУВАННЯ

Вважатимемо, що читач вже знайомий з поняттям інтегралу та його інтерпретацією як площі криволінійної трапеції, тому зупинятися на означеннях не будемо, а перейдемо до методів обчислення значень визначених інтегралів.

7.1. Метод прямокутників

Найчастіше формули чисельних квадратур (чисельне інтегрування) виводять, використовуючи геометричний зміст визначеного інтегралу, як площі криволінійної трапеції. Розбивають проміжок інтегрування [a;b] на частинні інтервали (рис. 7.1) та шукають наближено суму площ всіх утворених криволінійних трапецій S_i.

Розбиття проміжку інтегрування на рівномірні інтервали

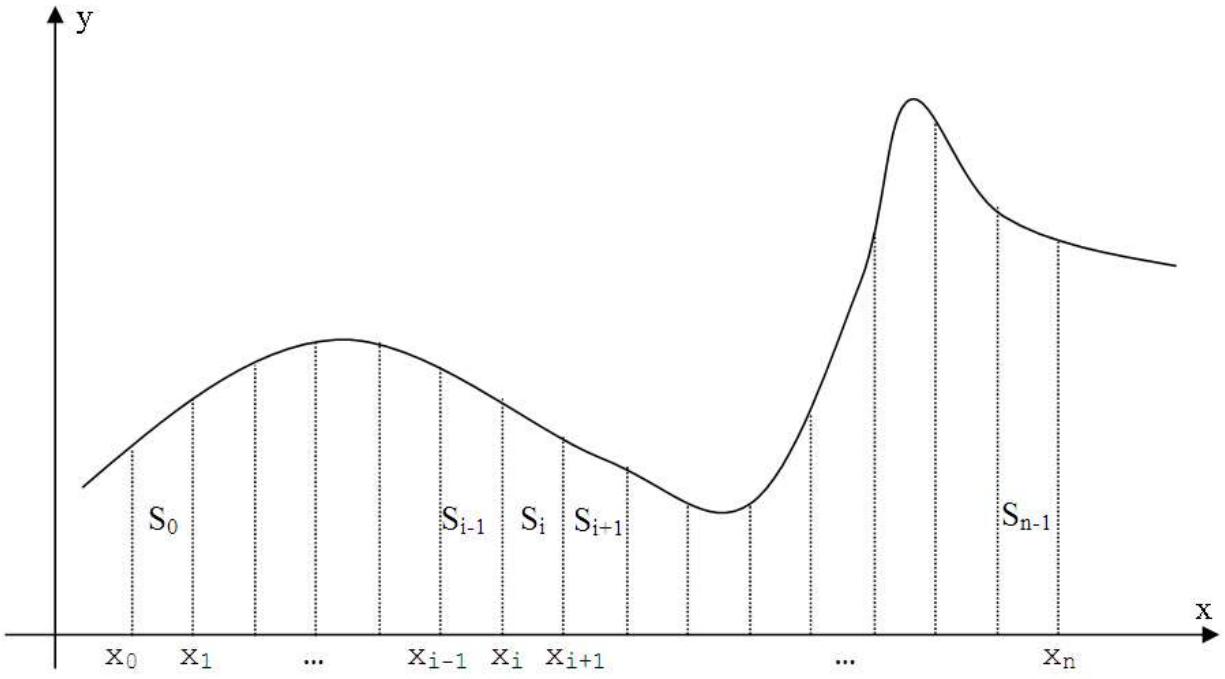

Рис. 7.1

Звісно, коли первісна існує в класі елементарних функцій, можна було б її використати для «точного» обчислення інтегралу, але для експериментальних даних відомі лише значення функції в окремих точках і ми не знаємо її аналітичне представлення (тобто формулу, за якою можна отримати значення функції $F(x_i)$ – первісної для $f(x)$). Тут потрібно використати інший спосіб – експериментальні значення наблизити відомою формулою, що легко інтегрується, а тоді інтегрувати вже цю функцію. Звісно, результат буде наближеним, але чим краще проведено наближення інтегрувальної функції, тим більш точним буде результат. Таким чином приходимо до задачі наближення експериментальних даних відомою формулою. Спробуємо наблизити дані експерименту кусково-сталими значеннями (див. рис. 7.2).

Наближення кусково-сталими функціями

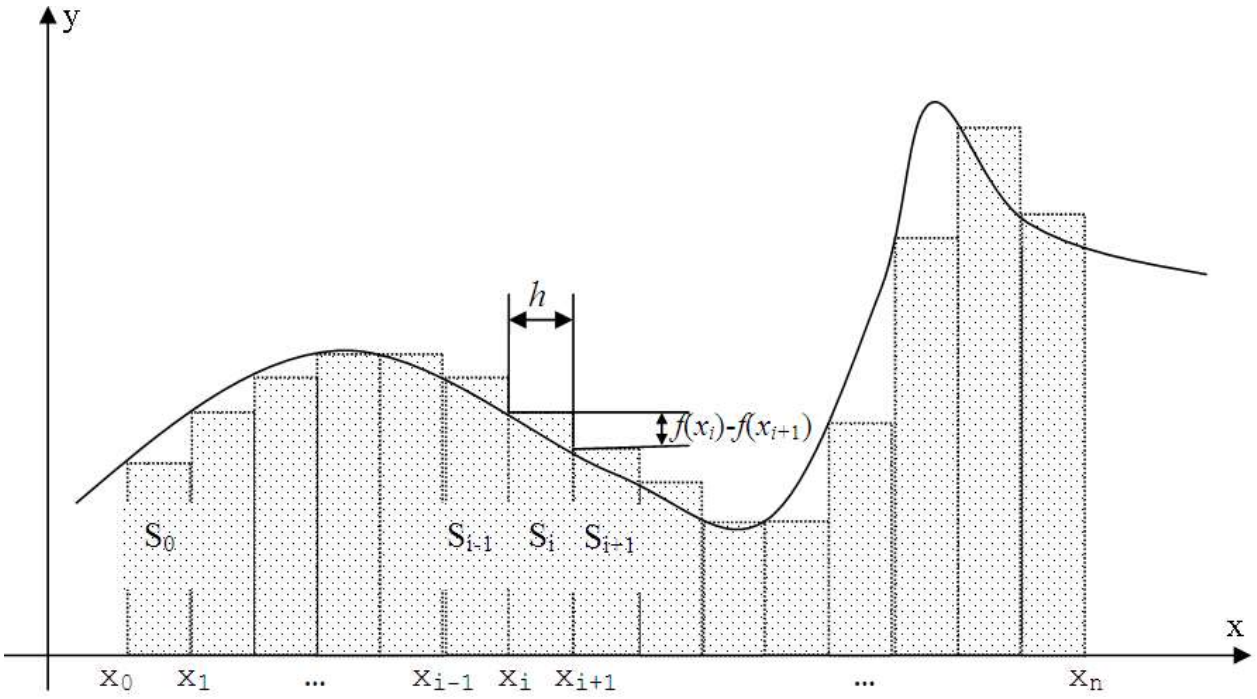

Рис. 7.2

Як видно з малюнку 7.2, утворені прямокутники мають площу, досить близьку до площ криволінійних трапецій, але в залежності від зростання чи спадання значень функції, прямокутники мають відповідно меншу чи більшу

площу від реальної. Чим швидше змінюються значення експериментальних даних, тим менш точна відповідність площ прямокутника та криволінійної трапеції. За швидкість зміни значень функції відповідає її похідна, тобто чим більшу за модулем має похідну функція, що інтегрується, тим більша похибка визначення площ. Тепер міркування запишемо у формулах.

Через те, що інтеграл не залежить від способу розбиття, для зручності будемо розбивати проміжок інтегрування на n *рівних* частин, тоді x_i-x_{i-1} буде сталим числом, яке позначатимемо літерою h. Тепер запишемо формули, що будуть використовуватися і надалі:

$$h=(b-a)/n,\ x_i=a+ih\ (x_0=a;\ x_n=b),\ y_i=f(x_i).$$

Всі ці значення вважатимемо відомими.

Площа i-ої криволінійної трапеції приблизно рівна площі прямокутника з тією ж основою h та висотою $f(\xi_i)$, де ξ_i – якась точка проміжку $[x_i;\ x_{i+1}]$. Сумуючи всі площі таких прямокутників, отримаємо відповідно формули, які дають наближене значення інтеграла. Зазвичай беруть в ролі ξ_i лівий чи правий кінець проміжка, що дає **методи лівих та правих прямокутників**:

$$S_n^{л}=\sum_{i=0}^{n-1} f(x_i)\cdot h=\frac{1}{n}\sum_{i=0}^{n-1} f(a+ih)\ ,\quad S_n^{n}=\frac{1}{n}\sum_{i=0}^{n-1} f(a+ih)\ . \qquad (7.1)$$

Не важко помітити, що ці формули дають точне значення інтеграла для кусково-сталої функції. Для інших функцій результат не точний, але тим точніший, чим менший h (більше n).

Щоб результат був дійсно корисним, необхідно знати, на яку величину він відрізняється від дійсного значення інтегралу. Дійсна площа криволінійної трапеції відрізняється від площі прямокутної трапеції

приблизно на площу деякого прямокутного трикутника (рис. 1.8), де один катет рівний h, а другий $|f(x_{i+1})-f(x_i)|$, тобто на $R_n=\sum_{i=0}^{n-1}|f(x_i)-f(x_{i+1})|\cdot h/2$.

Тут R_n позначено можливу абсолютну похибку визначення інтегралу. В свою чергу, різницю можна виразити через добуток похідної функції на крок аргументу: $|f(x_{i+1})-f(x_i)|\approx|f'(x_i)|h$. Якщо позначити числом ***M*** значення максимального модуля похідної функції $f(x)$, то оцінка похибки буде такою:

$$R_n\le n\,M\,h^2/2\text{, де } M=\sup_{[a;b]}|f'(x)| \quad .$$

Із формули видно, що при збільшенні кількості точок розбиття n вдвічі, вдвічі зменшиться h. Підставимо ці нові значення в попередню формулу:

$$R_{2n}\le(2n)\,M\,(h/2)^2/2\Rightarrow R_{2n}\le\frac{1}{2}n\,M\,h^2/2\Rightarrow R_{2n}\le\frac{R_n}{2} \quad .$$

Завдяки такій підстановці можна зробити висновок, що при збільшенні точок розбиття інтервалу вдвічі, абсолютна похибка результату стане вдвічі меншою.

Знайдемо методом прямокутників два значення інтегралу S_n при початковій кількості n інтервалів та при подвоєній кількості відрізків S_{2n}. Похибка другого результату повинна бути приблизно вдвічі менша, тобто:

$$S_{2n}+R_{2n}\approx S_n+R_n,\ S_{2n}+R_{2n}\approx S_n+2R_{2n},\ S_{2n}-S_n\approx R_{2n}.$$

Таке припущення дає змогу отримати наближене значення визначеного інтегралу та мати уявлення про надійність отриманого результату.

7.2. Метод трапецій

У виводі формули прямокутників частина кривої на проміжку наближалася сталою, та значно краще наближення буде при заміні частини кривої не сталою функцією, а лінійною (рис 7.3). Тепер площа криволінійної трапеції наближається площею не прямокутника, а площею трапеції.

Відомо, що площа трапеції S_i дорівнює добутку h на півсуму основ:

$S_i = \frac{f(x_i)+f(x_{i+1})}{2}\cdot h$, де $i=\{0;1;2;...;n\text{-}1\}$.

Сума площ всіх трапецій дає **формулу трапецій**:

$$S_n^m = \frac{h}{2}\left((f(x_0)+f(x_1))+(f(x_1)+f(x_2))+...+(f(x_{n-1})+f(x_n))\right),$$
$$S_n^m = \frac{h}{2}\left(f(a)+f(b)+2\sum_{i=1}^{n-1} f(x_i)\right). \qquad (7.2)$$

Кусково-лінійне наближення функції, що інтегрується

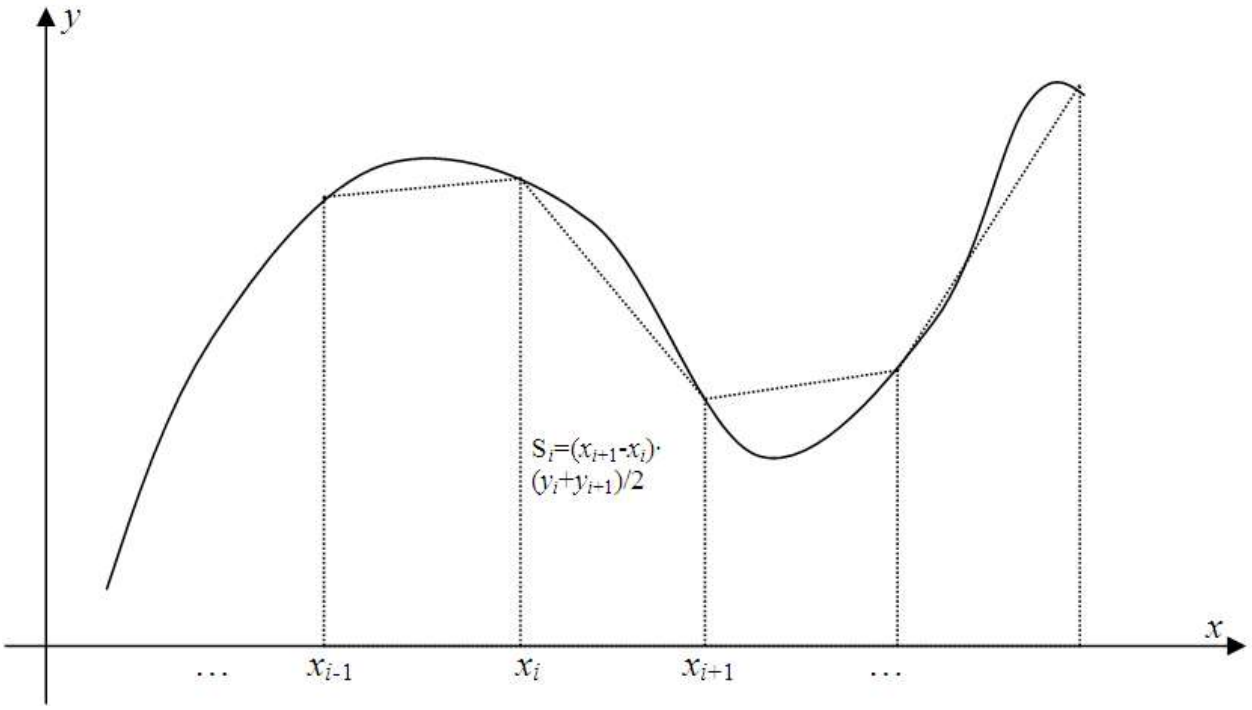

Рис. 7.3

На рисунку 7.3 показано наближення трапеціями криволінійних ділянок графіка функції. Це наближення краще за наближення прямокутниками і

основна похибка є результатом опуклості чи вгнутості графіка функції. Ступінь опуклості та вгнутості визначається другою похідною. Вважаємо на проміжку $[x_i;x_{i+1}]$ другу похідну рівною $f''(\xi_i)$, тоді похибка обчислення площі криволінійної трапеції є $R_i=\int_{x_i}^{x_{i+1}} f''(\xi_i)t^2\,dt$. Повна формула для оцінки похибки формули (7.2) [1, 161 с.], що отримана сумуванням похибок для кожної трапеції: $R_n=\sum_{i=0}^{n-1}\left(f''(\xi_i)h^3/3\right)$. На основі цієї формули побудоване твердження: модуль похибки не перевищує $R_n=nMh^3/3$, де $M=max_{[a;b]}|f''(x)|$, тобто максимум модуля другої похідної підінтегральної функції.

На практиці зручніше користуватись іншою формулою. Запишемо S_n при $h=(a-b)/n$ та при $h=(a-b)/(2n)$: (S_n і S_{2n}), відповідно їхні залишки будуть рівні

$$R_n=-\frac{(b-a)h^2}{12}M \quad , \quad R_{2n}=-\frac{(b-a)h^2}{12\cdot 4}M \quad .$$

У виразі для R_{2n} можна виділити R_n, випливає: $R_{2n}=R_n/4$. Якщо врахувати, що $S=S_n+R_n$ і $S=S_{2n}+R_{2n}$, запишемо:

$$S_n+R_n=S_{2n}+R_{2n},\ S_n+4R_{2n}=S_{2n}+R_{2n},$$
$$3R_{2n}=S_{2n}-S_n,$$
$$R_{2n}=(S_{2n}-S_n)/3. \qquad (7.3)$$

Ця формула дає змогу оцінити похибку інтегрування без дослідження другої похідної інтегрувальної функції.

Наближення функції параболою

Рис. 7.3

7.3. Метод Симпсона

Звичайно, функцію на проміжку $[x_i;x_{i+1}]$ можна наближати не лише лінійною, але й більш гнучкою функцією, наприклад, квадратичною. Так як через дві точки можна провести безліч парабол, то її проводять однозначно через три точки, вважаючи її за наближення функції на проміжку $[x_{i-1};x_{i+1}]$. Оскільки кожна парабола потребує три точки (криволінійна трапеція з двох частин), то проміжок інтегрування розбивається на парну кількість інтервалів довжиною h (рис. 7.3).

Знайдемо площу i-тої криволінійної трапеції:

Парабола задається на проміжку $x\in[x_{i-1};x_{i+1}]$ аналітично квадратним тричленом $a_2x^2+a_1x+a_0$, але в для спрощення записів та перетворень доцільно зробити заміну $t=x-x_i$. Таким чином отримаємо: $f(x_i+t)\approx a_2t^2+a_1t+a_0$. Тоді

$$\int_{x_{i-1}}^{x_{i+1}} f(x)\,dx\approx\int_{x_{i-1}}^{x_{i+1}} (a_2t^2+a_1t+a_0)\,dt=S_i \quad , \quad S_i=(a_2t^3/3+a_1t^2/2+a_0t)\Big|_{x_{i-1}}^{x_{i+1}} \quad ,$$

і враховуючи, що $x_{i+1}-x_{i-1}=2h$ остаточно маємо:

$$\int_{x_{i-1}}^{x_{i+1}} f(x)\,dx\approx S_i=2a_2h^3/3+2a_0h \quad . \tag{7.4}$$

Тут необхідно знайти лише $\boldsymbol{a_2}$ та $\boldsymbol{a_0}$. Для цього потрібно прирівняти значення функції у вузлах до значень наближення:

$$\begin{cases} f(x_i-h)=a_2h^2-a_1h+a_0, \\ f(x_i)=a_0, \\ f(x_i+h)=a_2h^2+a_1h+a_0. \end{cases}$$

Розв’язання системи дає значення потрібних коефіцієнтів a_2 та a_0:

$$a_0=f(x_i);\, a_2=\frac{f(x_{i+1})-2f(x_i)+f(x_{i-1})}{2h^2} \quad .$$

Підставимо в (7.4) значення коефіцієнтів a_2 та a_0:

$$\int_{x_i-h}^{x_i+h} f(x)\,dx\approx h(f(x_{i-1})+4f(x_i)+f(x_{i+1}))/3 \quad . \tag{7.5}$$

Щоб знайти значення інтегралу на проміжку $[a;b]$, потрібно знайти суму всіх S_i , де $i=1,3,5,...,n-1$. Тобто

$$S_n^{Симп}=\sum_{i=1}^{n/2} S_{2i-1} \quad , \quad \int_a^b f(x)\,dx \approx \frac{h}{3}\left(f(x_0)+4\sum_{i=1}^{n/2} f(x_{2i-1})+2\sum_{i=1}^{n/2-1} f(x_{2i})+f(b)\right) \quad . \quad (7.6)$$

Тут n – парне. Формула (7.6) має назву "формула Симпсона".

В [1, 162 с.] зазначені формули для оцінки похибок формули Симпсона при обчисленні інтегралу:

$$R \approx -f''''(\xi)(b-a)h^4/120 \quad \text{, та} \quad R \approx (S_{2n}-S_n)/15 \quad . \quad (7.7)$$

З оцінок похибки видно, що формула Симпсона не лише більш точна ніж попередні формули, але й її точність зростає значно швидше при зменшенні h.

Завдання до сьомого розділу

1. За останьою цифрою номеру в заліковій книжці p визначте свою функцію, для якої будете шукати визначений інтеграл $\int_1^{2+q} f(x)dx$, де q – передостання цифра номеру залікової книжки:

Номер варіанту	$f(x)$
1	$f(x)=\cos(4x)-x+1$.
3	$f(x)=\sqrt{\cos(3x-1)+x}$.
5	$f(x)=\ln(x+1)\cdot 3-\sin(x)$.
7	$f(x)=\sin(\cos(5x)-3)$.
9	$f(x)=e^{\sin(5x)}-x$.

Номер варіанту	$f(x)$
2	$f(x)=e^{-0.98x}-x^2$.
4	$f(x)=\sin^3(4x)+x-1$.
6	$f(x)=x^2\cdot\sin(x)-1$.
8	$f(x)=1-2\sin(\sqrt{5x})$.
0	$f(x)=x+e^{\cos(4x)}$.

Напишіть програму для розрахування визначеного інтегралу. Початково проміжок інтегрування розбийте на n=32 частини. Подвоюйте кількість підпроміжків n, для підвищення точності розрахунків. Зробіть оцінку похибки результату R_n. Розрахунки проведіть методами прямокутників, трапецій та Симпсона. Заповніть таблицю:

Метод	Значення	Кількість проміжків n	Похибка R_n
Прямокутників		32	$R_{64}=(R_{64}-R_{32})=$
		64	
Трапецій		32	$R_{64}=(R_{64}-R_{32})/3=$
		64	
Симпсона		32	$R_{64}=(R_{64}-R_{32})/15=$
		64	

8. ДИФЕРЕНЦІАЛЬНІ РІВНЯННЯ, ЗАДАЧА КОШІ

Нехай швидкість переміщення автомобіля виражається з його положення формулою $v(t)=2+\sin(x(t))/x(t)$. Швидкість руху є похідна положення тіла по часу: $v(t)=x'(t)$. Фактично це рівняння руху дає нелінійне диференціальне рівняння першого порядку $x'(t)=2+\sin(x(t))/x(t)$. Аналітичний (записаний елементарними функціями) загальний розв’язок такого рівняння не існує, але розв’язувати його потрібно. Яке положення автомобіля буде через 10 секунд, якщо він знаходився в початковий момент часу $x(0)=5$м?

Розглянемо досить просту фізичну задачу падіння тіла у гравітаційному полі, розташуємо якесь тіло (можна й Ваше) на відстані 400000 км від центру Землі з нульовою початковою швидкістю, та спробуємо розрахувати його швидкість на межі атмосфери при вільному падінні. Для цього ми повинні знати прискорення вільного падіння g, але для таких відстаней ця величина вже не постійна і виражається за формулою $g(h)=G\frac{M_з}{(h+R_з)^2}$, де h – відстань від поверхні Землі, G — гравітаційна стала, $R_з$ — радіус земної кулі, $M_з$ — маса Землі. Тепер можна зв'язати положення зі швидкістю $x''(t)=-G\frac{M_з}{(x(t)+R_з)^2}$ — нелінійне диференціальне рівняння другого порядку.

Ці задачі містять деякі початкові умови стану системи та рівняння до якого входить функція-параметр системи та похідні цієї функції — диференціальне рівняння. Ця комбінація має назву задачі Коші. У великій частині практичних випадках така задача не має загального розв’язку, але шукати частковий розв’язок можна за допомогою чисельних методів.

8.1. Метод Ейлера для задач Коші першого порядку

Розв'язання методом Ейлера задач Коші першого порядку покажемо на прикладі $x'(t)=2+\sin(x(t))/x(t)$ з початковою умовою $x(0)=500$м. Для розв'язання приймемо, що швидкість руху змінюється не неперервно а має стале значення на проміжку часу Δt. Тоді за цей час положення автомобіля зміниться на $\Delta x \approx x'(t)\Delta t$, і нове положення буде $x(t+\Delta t)\approx x(t)+x'(t)\Delta t = x(t)+\Delta t \sin(x(t))/x(t)$. За цим співвідношенням можна визначити положення автомобіля в наступний момент часу, а крок за кроком можна переходити як завгодно далеко в часі і визначити положення автомобіля в довільний момент. Алгоритм розрахунків матиме вигляд:

1) Визначаємо початкове положення $x(0)$ та $x'(t)$.

2) Задаємо крок в часі Δt та теперішній час $t=0$;

3) Рахуємо нове положення $x(t+\Delta t)=x(t)+x'(t)\Delta t$ та новий теперішній час $t:=t+\Delta t$.

4) Якщо не досягнуто потрібний момент часу, то переходимо до п.3.

Запишемо цей алгоритм для задачі з автомобілем на мові С++:

```
float NovePol(float x0,//початкове положення
              float dt,//крок в часі
              float kt)//кінцевий час
{
  float t=0.0;//початковий час
  float x=x0; //початкове положення є теперішнім
  while(t<kt) //повторити, доки не дійдемо до
              //кінцевого часу
  {
    x+=(2.0+sin(x)/x)*dt; //додамо зміну положення
    t+=dt; //змінимо теперішній час на наступний
  }
```

```
  return x; //повернемо нове положення
}
```

Для використання результатів розрахунків необхідно мати уявлення про їх надійність, тобто похибку обчислення. Тут разом з похибкою обчислення має істотний вплив й метод обчислення, бо неперервну зміну швидкості ми змінили на кусково рівномірний рух. Для рівноприскореного руху потрібно було врахувати й поправку $x''(t)\Delta t^2/2$. Тому можна очікувати, що похибка на кожному кроці не перевищує $max|x''(t)|\Delta t^2/2$. За час t проводиться $n=t/\Delta t$ кроків, тому загальна похибка розрахунків не перевищує $R_n \le n\, max|x''(t)|\Delta t^2/2 = t^2 \frac{max|x''(t)|}{2n}$. Фактично, як й метод інтегрування прямокутників, похибка зменшується вдвічі якщо двічі зменшити крок в часі.

Розглянемо як покращити точність розрахунків при розв'язанні задачі Коші першого порядку. Для цього потрібно трохи змінити алгоритм розв'язання ввівши крок уточнення результату.

1) Визначаємо початкове положення $x(0)$ та $x'(t)$.

2) Задаємо крок в часі Δt та теперішній час t=0;

3) Рахуємо нове положення $x_0(t+\Delta t)=x(t)+x'(x(t),t)\Delta t$.

4) Уточнюємо положення в новий час за формулою $x(t+\Delta t)=x(t)+(x'(x(t),t)+x'(x_0(t+\Delta t),t+\Delta t))\Delta t/2$, та новий теперішній час $t:=t+\Delta t$.

5)Якщо не досягнуто потрібний момент часу, то переходимо до п.3.

В цьому алгоритмі можна п.4 виконати декілька раз, уточнюючи значення наступного кроку. Фактично фізичним змістом четвертого пункту алгоритму розрахунків є те, що на проміжку часу хоча й швидкість є постійною, але вона приймається середньою між значеннями на початку та на кінці інтервалу. Фактично, таким простим прийомом, ми начебто замінили метод прямокутників методом трапецій, де враховуються значення швидкостей на початку та при кінці руху.

8.2. Метод Ейлера для задач Коші другого порядку

Метод Ейлера для задач Коші другого порядку потрібно використати для розв'язання наступної задачі про падіння тіла. В цьому випадку прискорення тіла залежить в загальному випадку, як від положення $x(t)$, так і від швидкості $x'(t)$. Тому для наступного моменту часу $t:=t+\Delta t$ потрібно перераховувати не лише положення тіла, але й його швидкість:

$$x''(t)=f(x(t),x'(t),t)\ ,$$
$$x(t+\Delta t)=x(t)+x'(t)\Delta t\ ,$$
$$x'(t+\Delta t)=x'(t)+x''(t)\Delta t\ .$$

Процедура реалізації таких розрахунків матиме вигляд:

```
float Padal(float x0,//початкове положення
            float x1,//початкова швидкість
            float dt,//крок в часі
            float kt)//кінцевий час
{
  float t=0.0;//початковий час
  float x=x0; //початкове положення є теперішнім
  float v=x1; //початкова швидкість є теперішньою
  while(t<kt) //повторити, доки не дійдемо до
              //кінцевого часу
  {
    v+=G*Mz/((x+Rz)*(x+Rz))*dt;//розрахуємо
//прискорення та приріст швидкості
    x+=v*dt; //додамо зміну положення
    t+=dt; //змінимо теперішній час на наступний
  }
```

```
  return x; //повернемо нове положення
}
```

Якщо Ви досить добре розумієте математику, то після цих прикладів зможете будувати схеми для розв’язання задачі Коші і для рівнянь більш високих порядків, та навести схему розрахунків з уточненнями.

Завдання до восьмого розділу

1. За останньою цифрою номеру залікової книжки *p*, визначте задачу Коші, якщо *q* – передостання цифра номеру залікової книжки:

Для всіх варіантів: початковий час $t_0=0$, кінцевий час $t_1=1+q$.

Номер варіанту	$f(x)$
1	$x'(t)=\sin(x(t)), x_0=3$.
3	$x'(t)=\cos(x(t)), x_0=3$.
5	$x'(t)=-x(t)\cdot\sqrt{(t)}, x_0=3$.
7	$x'(t)=x(t)\cdot\sin(t), x_0=3$.
9	$x'(t)=-\ln(\lvert x(t)\rvert+1), x_0=3$.

Номер варіанту	$f(x)$
2	$x'(t)=-\sin(x(t)), x_0=3$.
4	$x'(t)=-\cos(x(t)), x_0=3$.
6	$x'(t)=(x(t))^2-x(t), x_0=3$.
8	$x'(t)=-x^3(t)/3, x_0=3$.
0	$x'(t)=-x(t)\cos(t), x_0=3$.

Не обов'язковим додатковим завданням є побудова графіку функції $x(t)$, таблиця якої отримується в процесі розв'язання основної задачі Коші.

Розрахунки провести при різних значеннях приросту в часі $\Delta t=0.2;0.1;0.05$. Порівняйте результати та поясніть, де відповідь точніша.

Література

1. Амосов А. А. Вычислительные методы для инженеров : учебн. пособ. / А. А. Амосов, Ю. А. Дубинський, Н. В. Копченова. – М. : Высшая школа, 1994. – 544 с.
2. Барахнин В. Б. Введение в численный анализ / В. Б. Барахнин, В. П. Шапеев. – Новосибирск, 1997. – 112 с.
3. Бахвалов Н. С. Численные методы / Н. С. Бахвалов, Н. П. Жидков, Г. М. Кобельков. – М. : Бином, 2007. – 636 с.
4. Боглаев Ю. П. Вычислительная математика и программирование / Ю. П. Боглаев. – М. : Высшая школа, 1990. – 544 с.
5. Волков Е. А. Численные методы / Е. А. Волков. – М. : Высшая школа, 1987. – 312 с.
6. Демидович Б. П. Основы вычислительной математики / Б. П. Демидович. – М. : Наука, 1994. – 664 с.
7. Дэннис Дж. Численные методы безусловной оптимизации и решения нелинейных уравнений / Дж. Дэннис, Р. Шнабель – М. : Мир, 1988. – 40 с.
8. Заварыкин В. М. Численные методы / В. М. Заварыкин, В. Г. Житомирский, М. П. Лапчик. – М. : Просвещение, 1990. – 176 с.
9. Калиткин Н. Н. Численные методы / Н. Н. Калиткин. – М. : Наука, 1978. – 512 с.
10. Марчук Г. И. Методы вычислительной математики / Г. И. Марчук. – М. : Наука,1989. – 608 с.
11. Ортега Дж. Введение в численные методы решения дифференциальных уравнений / Дж. Ортега, У. Пул. – М. : Наука, 1986. – 56 с.
12. Ракитин В. И. Практическое руководство по методам вычислений с приложением программ для персональных компьютеров / В. И. Ракитин, В. Е. Первушин. – М. : Высшая школа, 1998. – 384 с.
13. Самарский А. А. Введение в численные методы / А. А. Самарский. – М. : Наука, 1997. – 240 с.