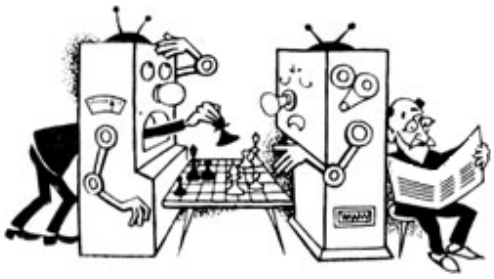
ЗМІСТ



ВСТУП

У навчальних планах вищих технічних навчальних закладів України дисципліна «Алгоритми та методи обчислень» є нормативною професійно-орієнтованою дисципліною підготовки бакалаврів з освітнього напряму «Комп'ютерна інженерія» (дещо рідше його називають «Інженерія комп'ютерних систем»). Очевидно, що ключову позицію у визначенні змісту освіти цього напряму має термін «Інженерія».

В українському науково-технічному лексиконі (як і в російському) цей термін став повноправним відносно недавно – приблизно двадцять років тому. Раніше, якщо цей термін і вживався, то вважався іншомовним запозиченням або жаргоном. Походить цей термін від латинського ingenium, що означає винахідництво, видумку, знання, мистецтво. В англокореневих мовах йому відповідає термін engineering, походження якого проте часто пов'язують зі словом engine – двигун, машина взагалі. А ось яке сучасне наповнення дає цьому терміну американська громадська організація American Engineer's Council for Professional Development (Американська рада інженерів з професійного розвитку): інженерія - це творче застосування наукових принципів для проектування або розроблення структур, машин, апаратів, виробничих процесів або використання їх окремо та в комбінаціях, конструювання або управління ними з повним знанням їх дизайну, передбачення їх поведінки за певних експлуатаційних режимів. Інакше інженерію можна визначити як сукупність робіт прикладного спрямування, що включає передпроектні техніко-економічні дослідження і обґрунтування планованих капіталовкладень, необхідну лабораторну і експериментальну доробку технологій і прототипів, їх промислову апробацію, а також наступні послуги і консультації.

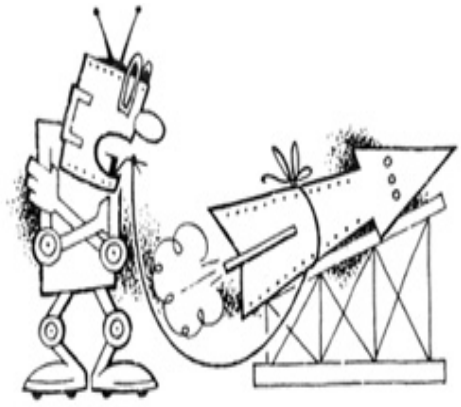
Таким чином, «Комп'ютерна інженерія» - це інженерія, предметом якої є комп'ютерні системи у всьому їх різноманітті, з усією їх науковою та технічною проблематикою. Загальною дефініцією комп'ютерних систем, достатньою для початкового ознайомлення з ними, може бути наступна: комп'ютерні системи – це складні двоєдині апаратно-програмні комплекси, що служать основним інструментальним засобом інформаційних технологій, тобто процесів утворення, передавання, зберігання, оброблення, модифікації та утилізації інформації.

Із появою дисципліни «Алгоритми та методи обчислень» у переліку нормативних дисциплін напряму «Комп'ютерна інженерія» деяка частина випускових кафедр стала її трактувати як набір теоретичних положень і фактів з традиційних розділів математики - «Теорії алгоритмів» та «Чисельних методів». Це знайшло своє відображення в ряді робочих програм дисципліни та підготовлених рукописах навчальних посібників. Однак розробники освітніх стандартів, включаючи вищеназвану дисципліну до навчальних планів, мали за мету побудувати її таким чином, щоб це була комп'ютерно-орієнтована дисципліна, максимально насичена прикладними обчислювальними аспектами з інженерії комп'ютерних систем. Викладення навчального матеріалу мало би широко використовувати макроархітектурні, мікроархітектурні, структурні, операційні особливості та характеристики сучасних комп'ютерних систем та мереж, спиратися на національні та міжнародні стандарти в галузі інформаційних технологій, сучасні мови та засоби програмування тощо. Посібники з цієї дисципліни мають охоплювати не стільки алгоритми та методи обчислень взагалі, скільки комп'ютерно-орієнтовані «методи обчислень» і «алгоритми обчислень». Найвищий ступінь професійної орієнтації такої дисципліни досягається прищепленням студенту всіх необхідних знань, умінь і навичок для того, щоб довести теоретичний математичний метод до практичного алгоритму, який покладений в основу діючої програми (мікропрограми) або якогось апаратного (технічного) засобу. Тому послідовність ***«математичний теоретичний метод, комп'ютерно-орієнтований алгоритм, програма»*** визначає, на думку авторів, парадигму побудови сучасного навчального посібника зазначеного освітнього напрямув. При цьому, враховуючи позицію дисципліни в навчальному плані і обмеженість часових витрат на її вивчення, важливо витримати певний оптимальний рівень викладу матеріалу з тим, щоб не вдатися до «захмарного» теоретизування та не опуститися до «голого» емпіризму. Саме такої «генеральної лінії» дотримувалися автори цього посібника і саме з таких міркувань в назві посібника зафіксовано дещо змінений порядок слів порівняно з назвою дисципліни в навчальному плані.

Обсяг і зміст посібника відповідають нормативним обсягу та змісту дисципліни, що вказані в освітньому стандарті напряму «Комп'ютерна інженерія», і тому є мінімально достатніми для здолання їх студентами. За необхідності більш глибокого вивчення деяких розділів чи тем це варто робити шляхом введення відповідних змін в варіативній частині навчальних планів. Для засвоєння матеріалу посібника необхідно мати математичну підготовку з дисциплін «Аналітична геометрія» та «Математичний аналіз» в обсязі, прийнятому для технічних університетів, а також програмістську підготовку з дисциплін «Програмування» та «Структури даних і алгоритми» в обсязі, передбаченому освітніми стандартами для студентів напряму «Комп'ютерна інженерія».

Матеріали посібника підготовлені викладачами кафедр, що ведуть навчальний напрям «Комп'ютерна інженерія», в Національному технічному університеті «Київський політехнічний інститут» та Київському Національному авіаційному університеті. Крім власне програмних положень посібник містить практично всі методичні знахідки, породжені більш ніж

двадцятирічним досвідом викладання цієї дисципліни (з назвами, які часом варіювали) на вказаних кафедрах.



Алгоритм вилучення кореня

РОЗДІЛ 1. ОСНОВИ ТЕОРІЇ АЛГОРИТМІВ

1.1. Основні поняття та визначення

Алгоритм – строго визначене правило дій, що включає вичерпні вказівки щодо того, як і в якій послідовності це правило треба застосовувати до вихідних даних деякої задачі, щоб отримати її розв'язок. Основними властивостями алгоритму є: детермінованість (визначеність) – однозначність результату процесу його застосування для заданих вихідних даних; дискретність процесу, що визначається алгоритмом - розчленованість його на окремі елементані акти (дії), можливість виконання яких людиною або машиною не викликає сумніву; масовість – вихідні дані для алгоритму можна вибирати із деякої множини даних, тобто алгоритм повинен забезпечувати отримання роз'вязку будь-якої задачі із класу однотипних.

Слово ***алгоритм*** походить від імені вченого, астронома та математика Аль-Хорезмі. Приблизно в 825 р. до н. е. він написав трактат, в якому описав запропоновану в Індії позиційну десяткову систему числення.

В першій половині XII століття книжка потрапила до Європи в перекладі латинською мовою під назвою *Algoritmi de numero Indorum*. Вважається, що перше слово в назві відповідає невдалій латинізації імені Аль-Хорезмі, а в цілому назва означає «Алгоритми про індійську лічбу».

Перший алгоритм, призначений для виконання на автоматичному обчислювальному пристрої (комп'ютері), описала Ада Лавлейс в 1843 році. Алгоритм мав обчислювати числа Бернуллі на аналітичній машині Чарльза Беббіджа. Цей алгоритм вважають першою комп'ютерною програмою, а його розробницю, Аду Лавлейс — першим програмістом.

З 1930-тих років починається розвиток галузі дослідження алгоритмів та становлення інформатики в її сучасному вигляді. Протягом 1935—1960 років було розроблено численні ідеї та технології, які покладені в основу сучасної інформатики.

Різноманітні теоретичні проблеми математики та прискорення розвитку фізики та техніки поставили на порядок денний проблему більш точного визначення поняття алгоритму.

Перші спроби уточнення поняття алгоритму та його дослідження здійснювали в першій половині XX століття Алан Тюрінг, Еміль Пост, Жак Ербран, Курт Гьодель, Андрій Марков, Алонзо Черч. Було запропоновано декілька визначень алгоритму, але згодом з'ясувалося, що всі вони визначають одне й те саме поняття.

Теорія алгоритмів як окремий розділ математики, що вивчає загальні властивості алгоритмів, виник у 30-х роках 20 століття. Проте алгоритми простежуються в математиці протягом всього часу її існування. Необхідність математичного уточнення інтуїтивного поняття алгоритму стала неминучою після усвідомлення неможливості існування алгоритмів розв'язку багатьох масових проблем, в першу чергу пов'язаних з арифметикою та математичною логікою (проблеми істинності формул першопорядкового числення предикатів та арифметичних формул). Для доведення факту відсутності існування деякого алгоритму треба мати його точне математичне визначення, тому після сформування поняття алгоритму як нової та окремої сутності першочерговою стала проблема знаходження адекватних формальних моделей алгоритму (відображення результатів мислення в точних поняттях та твердженнях) та дослідження їх властивостей. При цьому формальні моделі були запропоновані як для первісного поняття алгоритму, так і для похідного поняття алгоритмічно обчислюваної функції.

Алгоритмічну теорію можна розділити на ***дескриптивну*** (якісну) і ***метричну*** (кількісну). Перша досліджує алгоритми з точки зору встановлення ними відповідності між вихідними даними і результатами; до неї належать, зокрема, проблеми побудови алгоритмів, наявності в них тих чи інших властивостей і т. п. — це так звані алгоритмічні проблеми. Друга досліджує алгоритми з точки зору складності як самих алгоритмів, так і обчислень, що ними задаються, тобто процесів послідовного перетворення конструктивних об'єктів. Варто відзначити, що як складність алгоритмів, так і складність обчислень можуть визначатися різними способами. Розробка методів оцінки складності алгоритмів і обчислень має важливе теоретичне і практичне значення.

Завданняі, які виконує теорія алгоритмів:

- формалізація поняття «алгоритм» та дослідження формальних алгоритмічних систем (моделей);
- доведення алгоритмічної нерозв'язності задач;
- формальне доведення правильності та еквівалентності алгоритмів;
- класифікація задач, визначення та дослідження складності класів алгоритмів;
- доведення теоретичних нижніх оцінок складності задач;
- створення методів розробки ефективних алгоритмів;
- асимптотичний аналіз складності ітераційних алгоритмів;
- дослідження та аналіз рекурсивних алгоритмів;
- отримання явних функцій трудомісткості алгоритмів;
- розробка класифікації алгоритмів;
- дослідження ємності, складності задач і алгоритмів;
- розробка критеріїв порівняльної оцінки ресурсної ефективності алгоритмів та методів їх порівняльного аналізу.

Теорія алгоритмів є теоретичним фундаментом програмування, вона застосовується всюди, де виникають алгоритмічні проблеми (рис.1.1): основи математики, теорія керування, конструктивний аналіз, обчислювальна математика, теорія ймовірностей, лінгвістика, економіка та ін.

Математична логіка. Теорія алгоритмів виникла як розділ математичної логіки, поняття алгоритму тісно пов'язане з поняттям числення. Перші та найчисленіші застосування теорія алгоритмів має саме в математичній логіці – вона вивчає лише умовиводи зі строго визначеними об’єктами, для яких можна однозначно визначити: істинні вони чи хибні. Прикладом може бути побудова ***силогізмів*** – тверджень, у яких із заданих двох тверджень (консеквентів) виводиться третій (антицедент). Наприклад, «Всі метали – прості речовини. Магній – метал. Магній –

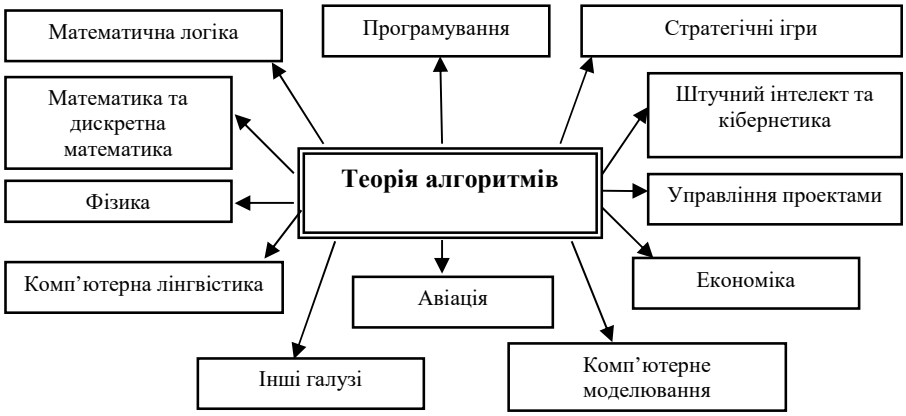


Рис. 1.1. Основні області застосування теорії алгоритмів

проста речовина». На практиці множина елементарних логічних операцій є обов’язковою частиною набору інструкцій всіх сучасних комп’ютерів і входить до мов програмування.

Стратегічні ігри – одна із найпопулярніших форм сучасних ігор, основа розробки стратегій управління підприємствами, галузями господарства. інноваційною діяльністю тощо. Розглянемо приклад простої стратегічної гри. Нехай на площині задано прямокутник, розбитий на однакові клітинки зі стороною одиничної довжини. Деякі з клітинок заштриховані – це стіни, решта клітинок – коридори. Таким чином задано лабіринт, вхід якого знаходиться у правій нижній клітинці (помічена цифрою 0), а вихід – у лівій верхній, поміченій зірочкою (рис. 1.2).

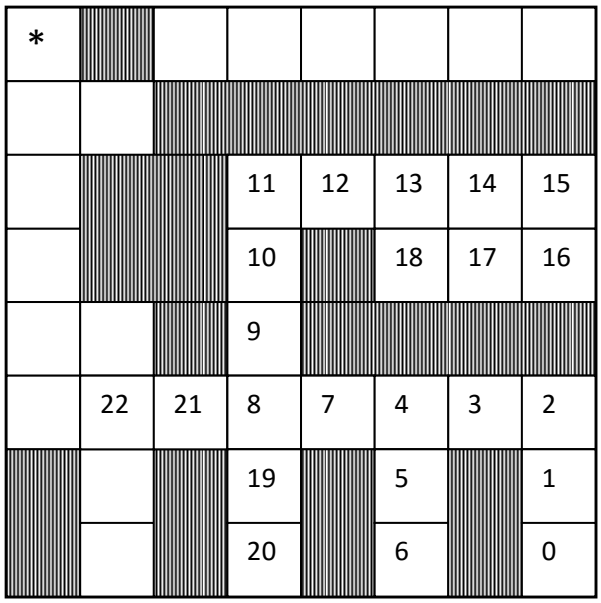


Рис. 1.2. Приклад стратегічної гри

У початковий момент часу на вхідній клітинці знаходиться робот (наприклад, на ім'я Бендер), який вміє розрізняти стіни від коридорів, робити хід на одну сусідню незаштриховану клітинку зверху, справа, знизу або зліва; вміє приймати найпростіші логічні рішення та здатен розпізнавати символ зірочки. Наперед відомо, що у такому лабіринті існує як мінімум один шлях, що веде по коридорах від входу до виходу. Необхідно запрограмувати Бендера так, щоб він, керуючись даною програмою нескладних операцій, зміг знайти вихід із лабіринту.

Зарезервуємо досить велику кількість міток, пронумерованих натуральними числами. Цими мітками Бендер буде відмічати свої ходи. Шукана програма складається із таких кроків.

1. Відмічаємо праву нижню клітинку міткою 0. Переходимо до пункту 2.

2. Перевіряємо, чи істинне те, що клітинка, у якій знаходиться Бендер, є виходом. Якщо це вірно, то переходимо до пункту 6, якщо ні, переходимо до пункту 3.

3. Перевіряємо, чи існує хоча б одна непомічена сусідня клітинка (зверху, знизу, справа або зліва). Якщо існує, переходимо до пункту 4, якщо ні - до пункту 5.

4. Обираємо довільним способом одну сусідню непомічену клітинку. Наприклад, першу непомічену клітинку при обході всіх чотирьох сусідніх клітинок за годинниковою стрілкою, починаючи з верхньої. Відмічаємо обрану клітинку найменшою невикористаною міткою, потім пересуваємо Бендера на цю клітинку і переходимо до пункту 2.

5. Повертаємося на один крок назад, тобто пересуваємо Бендера назад у ту клітинку, з якої він останній раз перейшов на поточну клітинку. Переходимо до пункту 2.

6. Робота алгоритму припиняється, вихід із лабіринту знайдено. Задачу розв'язано.

Цей алгоритм завжди приводить до знаходження виходу з лабіринту.

Іншими прикладами стратегічних ігор можуть бути «хрестики-нулики», шахи і т.п.

Математика. Уже на початку XX століття для математиків слово «алгоритм» означало довільний арифметичний або алгебраїчний процес, що його виконують за строго визначеними правилами. Всі математичні обчислення відбуваються за певною схемою, алгоритмом. Навіть порядок використання математичних знаків – це алгоритм, неправильне застосування або незнання якого призведе до отримання неправильних результатів. Приклад застосування алгоритмів в математиці – це порівняння чисел, обчислення площ та периметрів фігур, розв'язування диференціальних рівнянь та ін.

Фізика. Результатами вивчення кожного з розділів фізики є вирішення певних задач, що моделюють різноманітні фізичні процеси та явища. Для кожного виду фізичних задач існує чітко визначена схема їх розв'язку, наприклад, для розв'язання задач знаходження відстані через відомі швидкість та час, обчислення прискорення, об'єму, висоти, кінетичної і потенціальної енергії та ін.

Так, загальним алгоритмом для розв'язання задач фізики є наступна послідовність кроків:

- зрозуміти отриману задачу (побачити фізичну модель);
- проаналізувати та побудувати математичну модель явища: обрати систему відліку, зрозуміти суть фізичного процесу, визначити початкові та додаткові умови,
- розв'язати отриману математичну задачу відносно шуканої величини;
- розв'язок перевірити та оцінити.

Штучний інтелект та кібернетика вивчають різноманітні технології створення інтелектуальних машин та інтелектуальних комп'ютерних програм. Зокрема, методи штучного інтелекту використовують алгоритми управління в робототехніці, розпізнаванні образів, нейронних мережах, експертних системах та ін. Одним із відомих є так званий алгоритм Британського музею, запропонований Ньюелом, Саймоном та Шоу, що полягає у написанні ряду правил для породження здогадок та перевірки їх правильності. Навіть поняття інтелекту вчені розглядають як виконання певного алгоритму: здатність пристрою, що приймає рішення, досягати певного успіху при пошуку широкого різноманіття цілей у певному діапазоні.

Набором правил композиції, отриманим з аналізу музичних творів, користуються при машинному створенні музики. Алгоритм машинного створення музичної композиції використовує спеціальний пристрій, який пропонує закодовані цифрами ноти. Кожна нота проходить через фільтр правил композиції, побудований на основі бази даних. Якщо нота задовольняє правилам, вона заноситься в нотний рядок, якщо ні – нота відкидається і замість неї аналізується інша. Цей процес відбувається доти, доки не буде створено готову композицію.

Комп'ютерна лінгвістика – напрям штучного інтелекту, що використовує математичні моделі для опису природних мов. Полем діяльності комп'ютерних лінгвістів є розроблення алгоритмів та прикладних програм для оброблення природно-мовної інформації. Прикладом застосування теорії алгоритмів є морфологічний, синтаксичний та семантичний розбір, стратегії пошуку.

Сьогодні мережа Інтернет надає можливість швидко отримувати доступ до інформації, вилучаючи її у великих обсягах. Управління та маніпуляція даними здійснюють за допомогою алгоритмів визначення оптимальних маршрутів, за якими переміщаються дані, швидкий пошук сторінок, на яких знаходиться інформація та ін.

Управління проектами – це синтетична дисципліна, що стосується проектної діяльності, в ході якої визначаються та досягаються чіткі цілі проекту при балансуванні між об'єктом робіт, ресурсами, часом, якістю та ризиками. Ключовим фактором успіху проектного управління є наявність завчасно визначеного плану, мінімізація ризиків та відхилень від плану, ефективне управління змінами. План у даному випадку – це і є алгоритм. Для високої рентабельності

проекту необхідно оптимізувати витрати та ресурси. Наприклад, при прокладанні доріг за допомогою розроблених алгоритмів визначають найкоротший шлях від однієї точки сполучення до іншої; в авіакомпаніях визначають мінімальну ціну на авіабілети, щоб зменшити кількість вільних місць у літаку; при створенні підприємства складають бізнес-план, що включає в себе певну кількість наперед визначених кроків.

Економіка широко використовує стратегії планування, оцінку ефективності роботи підприємства, системи бухгалтерського обліку і т.д. Електронна комерція дозволяє укладати угоди та надавати товари та послуги за допомогою електронних технічних засобів. Для швидкого поширення цієї системи важливо вміти захищати таку інформацію, як номери кредитних карт, паролі та банківські рахунки. Такі базові технології у цій області, як криптографія з відкритим ключем та цифровий підпис, базуються на чисельних алгоритмах.

Комп'ютерне моделювання використовує різноманітні алгоритми (набори дій) для реалізації конкретного завдання у вигляді роботи певної моделі або системи. Комп'ютерне моделювання є одним з ефективних методів вивчення складних систем. Логічність та формалізованість (побудова за певними правилами, алгоритмами) комп'ютерних моделей дозволяє виявити основні фактори, що визначають властивості об'єкту дослідження.

Авіація користується алгоритмами прийняття рішень: збору інформації, аналізу варіантів та виконання дій, алгоритмами навігаційного пошуку, управління в нестандартних та екстрених ситуаціях.

Програмування – це процес створення комп'ютерних програм, написання інструкцій мовою програмування за певним алгоритмом – планом, методом виконання поставленого завдання.

Алгоритмізація виробничих процесів – це здійснення математичного опису (створення математичної моделі) виробничого процесу. Джерелом вихідної, початкової інформації для алгоритмізації виробничих процесів служать теоретичні та експериментальні дані, а також евристичні, неформальні відомості щодо процесу вивчення. Ця інформація може бути отримана завчасно (апріорні дані) та безпосередньо в процесі досліджень (апостеріорні дані). Через складність та великий об'єм даних алгоритмізація виробничого процесу часто протікає за індивідуальною схемою, найбільш раціональною для даного складного об'єкту та конкретних умов дослідження.

Найпоширеніша схема алгоритмізації виробничого процесу включає наступні етапи:

- попередній аналіз задачі алгоритмізації та об'єкту дослідження – ставляться цілі та основні етапи дослідження, оцінюється очікувана економічна ефективність та доцільність прийнятої схеми дослідження об'єкту та результатів його алгоритмічного аналізу, при цьому враховується неповнота інформації;
- структурний опис виробничого процесу, що досліджується, пов'язаний із застосуванням методів мережевих представлень (блок-схем, графів) для відображення зв'язків, що існують між параметрами та елементами виробничого процесу;
- теоретичний аналіз рівнянь зв'язку між параметрами процесу;
- експериментальне визначення статичних та динамічних характеристик процесу;
- моделювання процесу та перевірка адекватності (відповідності) математичного опису реальному виробництву;
- аналіз отриманої математичної моделі та підготовка рекомендацій з поліпшення виробничого процесу;
- формування оптимальних алгоритмів на основі рекомендацій попередніх етапів;

- перевірка та коректування алгоритмів забезпечення системи управління виробничими процесами в умовах експлуатації системи.

1.2. Властивості та класифікація алгоритмів

Властивості алгоритмів

Дискретність – процес, що визначається алгоритмом, можна розчленувати (розділити) на окремі елементарні етапи - ***кроки***, кожен з яких називають кроком алгоритмічного процесу чи алгоритму.

Скінченність – довільний алгоритм задають як скінчений набір інструкцій. Алгоритм має завжди завершуватись після виконання скінченної кількості кроків. Процедуру, яка має решту характеристик алгоритму, без, можливо, скінченності, називають ***методом обчислень***.

Детермінованість – обчислення відбуваються детерміновано, тобто обчислювальний пристрій однозначно вирішує, які інструкції необхідно виконувати у поточний момент часу.

Визначеність – кожен крок алгоритму має бути точно визначений. Дії, які необхідно здійснити, повинні бути чітко та недвозначно визначені для кожного можливого випадку. Це означає, що виконання команд алгоритму відбувається у єдиний спосіб та призводить до однакового результату для однакових вхідних даних.

Направленість – у кожного алгоритму є ***вхідні*** та ***вихідні*** дані. Вхідні дані: алгоритм має деяку кількість (можливо, нульову) даних, заданих до початку його роботи або значення яких визначають під час роботи алгоритму. Вхідні дані алгоритму можуть бути обмежені набором допустимих даних. Застосування алгоритму до недопустимих вхідних даних може призводити до того, що алгоритм ніколи не зупиниться або потрапить в тупиковий стан (зависання), з якого не зможе продовжити його виконання. Вихідні дані: алгоритм формує певну кількість даних, що мають досить визначений зв'язок із вхідними даними. В алгоритмі мають бути чіткі вказівки щодо його зупинки і щодо видачі вихідних даних після зупинки.

Елементарність дій – робота кожного алгоритму реалізується шляхом виконання послідовності деяких елементарних дій. Ці дії називають ***кроками,*** а процес їх виконання називають ***алгоритмічним процесом***. Кожна дія є настільки простою, що не допускає можливості неоднозначного трактування.

Масовість – можливість застосування алгоритму до різних вхідних даних. Тобто, кожен алгоритм призначений для розв'язку класу однотипних задач.

Алгоритм вважається ***правильним***, якщо для довільних допустимих даних він закінчує роботу і видає результат, який задовольняє вимогам задачі.

Алгоритм ***однозначний***, якщо при застосуванні до одних і тих самих вхідних даних він видає один і той же результат.

Будь-який алгоритм застосовується до вхідних (початкових) даних та видає результуючі дані. В ході роботи алгоритму з’являються проміжні дані. Для опису даних фіксується набір елементарних символів (алфавіт даних) та даються провила побудови складних даних із простих. Приклади простих даних: цілі та дійсні числа, логічні та символьні змінні, приклади складних – масиви, рядки, структури.

Всі дані для свого розміщення потребують пам’яті. В прикладних алгоритмічних моделях об’єм даних можна вимірювати кількістю комірок пам’яті, у яких розміщені дані.

Елементарні кроки алгоритму складаються із базових дій, число яких скінченне. Як базові дії часто розуміють машинні команди, що входять у набір команд комп’ютера. У випадку подання алгоритмів мовами високого рівня як базові дії можуть виступати оператори мови.

Для сучасності характерне широке використання алгоритмічних мов, які можна розглядати як уточнення поняття алгоритм, а саме: алгоритм трактується як текст, записаний алгоритмічною мовою. Семантика такої мови визначає для кожного алгоритму (програми) деяку сукупність процесів обчислень, які реалізуються в залежності від стану інформації, що переробляється алгоритмом.

Алгоритмічна мова – формальна мова, призначена для запису алгоритмів. Використання алгоритмічних мов базується на можливості формального задання правил конструювання алгоритмів. Алгоритмічну мову задають її алфавітом (або словником вихідних символів) і точними описами її синтаксису (граматики, правил побудови мовних конструкцій) та семантики (змістовним наповненням формальних побудов). У кожній алгоритмічній мові повинні бути засоби для задання операторів, що відповідають діям по переробці інформації, та операторів переходу, що визначають порядок виконання цих дій.

Універсальні алгоритмічні мови, за допомогою яких будуються класичні алгоритмічні схеми (нормальні алгоритми Маркова, рекурсивні функції, машина Тюрінга, машина Поста), виявилися практично непридатними для опису алгоритмів розв'язання задач при їх реалізації на комп'ютерах з тієї причини, що всі ці мови зорієнтовані на розгляд фундаментальних питань алгоритмічної теорії, і у них не знаходять свого відображення важливі особливості структури і функціонування сучасних комп'ютерів, наприклад, наявність цілої ієрархії запам'ятовуючих пристроїв та зовнішнього середовища, з яким пов'язане введення та виведення даних. Тому необхідність розв'язку практичних задач за допомогою комп'ютерів призвела до створення мов програмування, для яких алгоритмічні мови служать основою.

Класифікація алгоритмів

Класифікацію алгоритмів пояснює рис. 1.3.

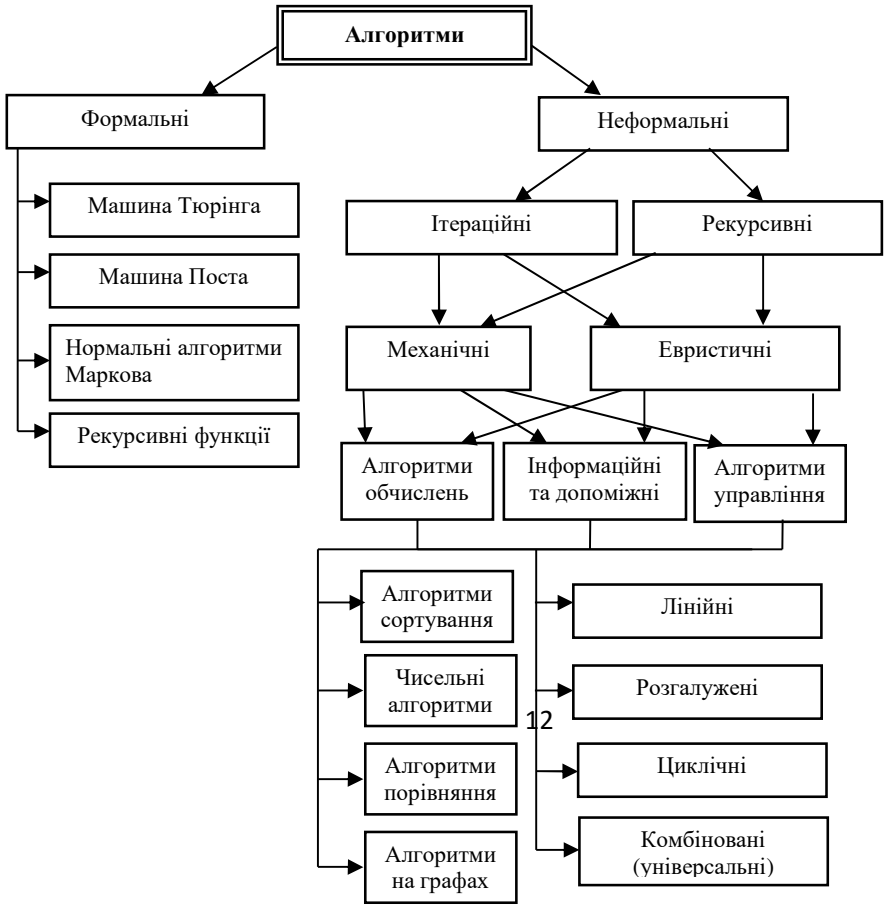


Рис. 1.3. Класифікація алгоритмів

Ітераційні (покрокові) алгоритми на кожному кроці використовують одну і ту ж формулу, яка виражає певну змінну величину через її значення, отримані на попередніх кроках цього алгоритму.

Рекурсивні алгоритми в процесі їх виконання звертаються самі до себе, але з іншими вхідними даними (можна вважати, що це застосування математичної індукції до послідовності дій алгоритму).

Механічні алгоритми – детерміновані (жорсткі), наприклад, алгоритм роботи машини, двигуна, літака. Такі алгоритми задають певні дії, вказуючи їх у єдиній та чіткій послідовності, і тим самим забезпечує однозначний шуканий результат, якщо виконуються ті умови процесу, задачі, для яких розроблено алгоритм.

Евристичний алгоритм – це алгоритм, який не має строгого обґрунтування, але в більшості випадків дозволяє отримати правильний результат.

Алгоритми обчислень працюють з простими видами даних (числа, матриці), але сам процес обчислень може бути тривалим і складним.

Інформаційні алгоритми являють собою набір порівняно невеликих процедур (наприклад, пошуку числа, слова), проте працюють з великими обсягами інформації (базами даних). Для їх ефективної роботи важливо мати правильно організовані дані.

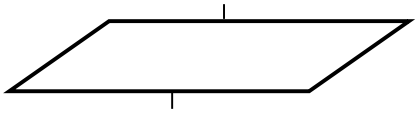
Алгоритми управління характеризуються тим, що дані до них потрапляють від зовнішніх процесів, якими вони управляють. Результатом роботи таких алгоритмів є різні керувальні впливи.

Блок-схемний спосіб завдання алгоритму – це графічне зображення алгоритму у вигляді окремих частин – блоків. Кожен блок відповідає окремому етапу розв'язання задачі. Порядок виконання етапів (блоків) задається з'єднувальними лініями або номерами блоків. Правила графічного виконання блок-схем алгоритмів в Україні встановлюються «Єдиною системою програмної документації» (ЄСПД). Відповідно до ЄСПД функціональні зв'язки (або переміщення) між блоками на блок-схемах, направлені згори вниз та зліва направо, стрілками не позначають. Якщо ж направленість переміщення по блок-схемі алгоритма має бути знизу вгору та справа наліво, то його позначають стрілками відповідного спрямування.

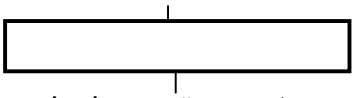
Типові дії алгоритму відображаються наступними геометричними фігурами.

1. Блоки початку та кінця алгоритму:

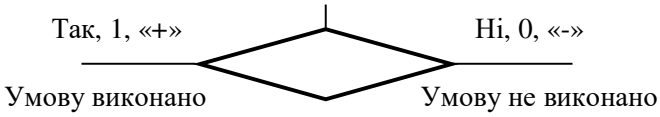
2. Блок вводу-виводу даних:



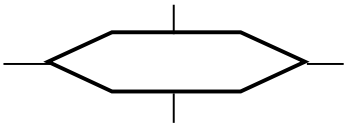
3. Блок виконання певних операцій або розв’вязання деякої арифметичної задачі:



4. Умовний блок (відповідає перевірці деякої умови):

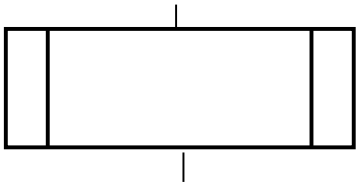


5. Блок циклу:



6. Перенесення алгоритму на іншу сторінку (з іншої сторінки):

7. Блок підпрограм:



8. Коментарі (пояснення):

Лінійний алгоритм – це процес, в якому команди виконуються послідовно, у порядку їх запису, одна за одною (рис. 1.4).

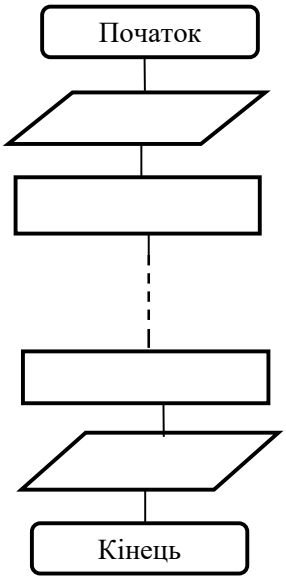


Рис. 1.4. Графічне зображення лінійного алгоритму

Розгалужені алгоритми описують процеси, в яких послідовності дій залежать від виконання (чи невиконання) деякої умови (рис. 1.5).

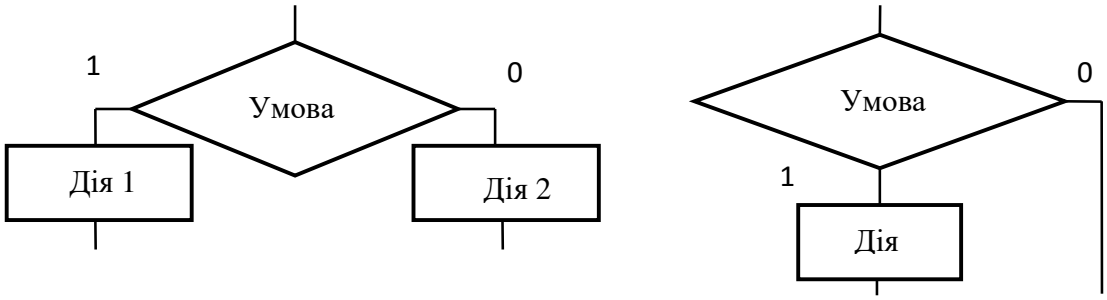


Рис. 1.5. Графічне зображення розгалуженого алгоритму

Циклічні алгоритми застосовуються, коли в залежності від певних умов необхідно повторювати одні і ті ж дії деяку кількість разів. Розрізняють цикли з передумовою, з параметром та з післяумовою (рис. 1.6).

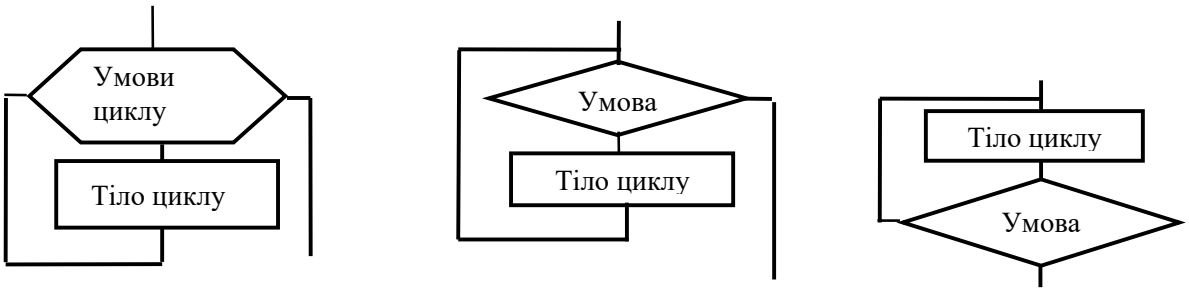


Рис. 1.6. Графічне зображення циклічного алгоритму

Комбіновані алгоритми використовують елементи лінійних, розгалужених та циклічних алгоритмів. Прикладом є алгоритм (рис. 1.7) до раніш приведної задачі про робота Бендера.

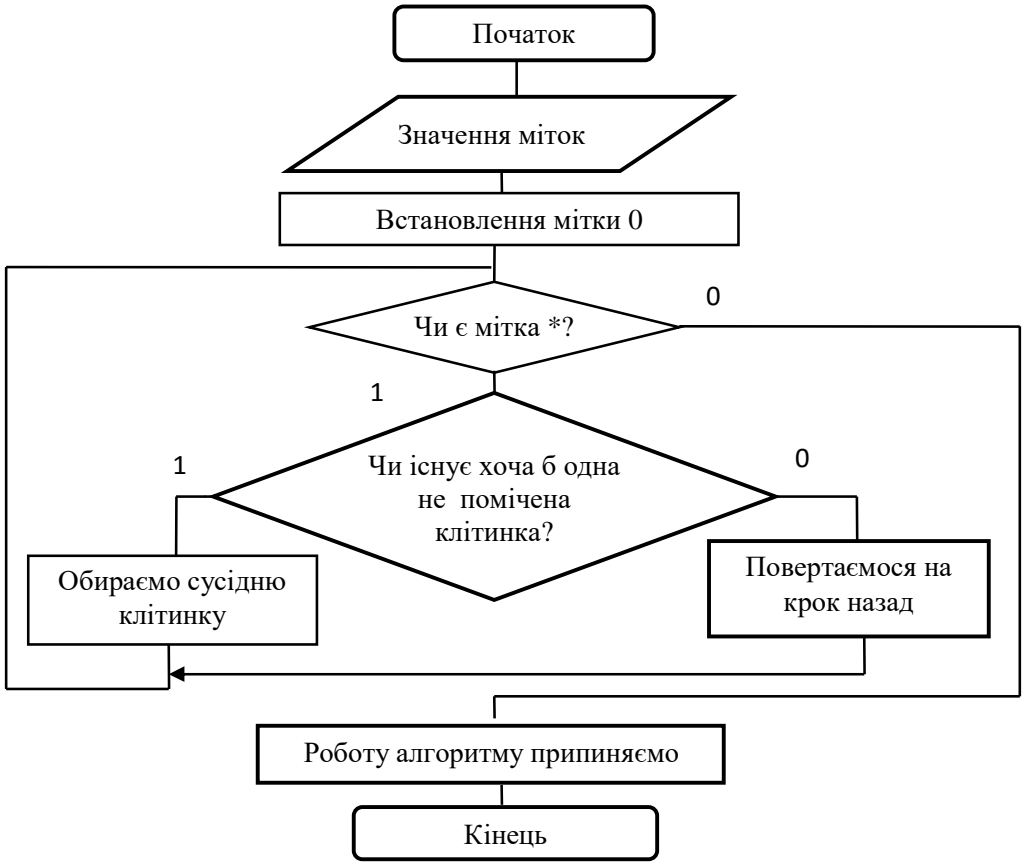


Рис. 1.7. Графічне зображення комбінованого алгоритму

Зауважимо, що графічне зображення алгоритмів у вигляді блок-схем є наочним і зручним для відносно нескладних алгоритмів (з числом операційних та умовних блоків, яке не перевищує, наприклад, декількох десятків). Однак графічне зображення обчислювальних алгоритмів, які розглядаються в наступних розділах, приводить до дуже складних, громіздких і

важкоосяжних блок-схем. Тому надалі для стислого запису алгоритмів використовуються псевдокоди, що значно спрощує перехід до кодування алгоритмів мовами програмування високого рівня.

На практиці процесс розроблення деякого алгоритму включає наступні етапи.

1. ***Постановка завдання*** – це точне формулювання проблеми для правильного її розуміння та розв'язання.

2. ***Розроблення моделі*** (формалізованої сукупності взаємопов'язаних математичних та формально-логічних виразів, що відображають реальні процеси та явища) передбачає виявлення функціональних залежностей та представлення їх у вигляді математичної формули, програмного комплексу, формального опису.

3. ***Побудова алгоритму.*** Після постановки завдання та побудови для нього моделі необхідно розробити алгоритму його розв'язку. Вибір методу розроблення визначається побудованою моделлю і може істотно вплинути на ефективність розробленого алгоритму. Для побудови коректного алгоритму необхідно ретельно проаналізувати побудовану модель та її обмеження.

4. ***Перевірка правильності алгоритму*** є одним із найбільш важких етапів побудови алгоритму.

5. ***Аналіз алгоритму*** – перед тим, як розпочати реалізацію алгоритму на певній апаратній платформі, необхідно провести оцінки часу його виконання та об'єму необхідної комп'ютерної пам'яті – основних параметрів, які визначають можливість конкретної реалізації алгоритму. Цілком ймовірно, що для розв'язання навіть відносно алгоритмічно простих задач можуть знадобитися десятки або сотні годин машинного часу та величезні сховища даних.

6. ***Програмна реалізація алгоритму*** – безпосереднє створення програми, що включає етапи: 1) ***кодування*** програми; 2) ***відлагодження,*** що передбачає усунення синтаксичних та логічних помилок; 3) ***тестування*** – передбачає доведення правильності програми, що можна провести аналогічно доведенню правильності алгоритму.

Будь-який розроблений алгоритм повинен задовольняти наступним ***критеріям***:

- досягнення оптимального часу розв'язання задачі;
- повне використання наявних ресурсів (операційних і пам'яті);
- забезпечення необхідної точності обчислень;;
- щонайширше використання стандартних підпрограм (якщо є розроблені програми, що можуть бути використані у створюваній, ними потрібно користуватися, а не створювати заново).

1.3. Приклади реалізації алгоритмів абстрактними машинами Тюрінга і Поста

Перші фундаментальні наукові роботи з теорії алгоритмів були опубліковані в середині 30-тих років 20-го століття А. Тюрінгом, А. Черчем та Е. Постом. Запропоновані ними машина Тюрінга, машина Поста та клас рекурсивно-обчислюваних функцій Черча були першими формальними описами алгоритму, що використовували строго визначені моделі обчислень. Важливим розвитком цих робіт стало формулювання та доведення існування алгоритмічно нерозв'язних задач.

Машина Тюрінга. У 1936 році студент Кембриджського університету Алан Тюрінг написав статтю «Про обчислювальні числа», у якій для уточнення поняття «алгоритм» розглянув гіпотетичний (абстрактний, уявний) пристрій, призначений для виконання логічних операцій. Згодом цей пристрій отримав назву машини Тюрінга.

Отже, основою машини Тюрінга (рис. 1.8) є стрічка, розділена на окремі однакові комірки. У комірках можуть міститися символи із зовнішнього алфавіту ($A = a_0, a_1, ..., a_n$), причому вміст комірок може змінюватися. Стрічка потенційно нескінченна як в один бік, та і в другий, тобто не можна дійти до її кінця, проте у довільний момент часу лише скінченна кількість комірок може бути непорожньою. На початку стрічка містить вхідні числа, в кінці – вихідні числа. Решта комірок стрічки використовується як простір пам'яті для обчислень. Крім стрічки машина має пристрій зчитування, який в процесі роботи машини може переглядати лише одну комірку та розпізнавати вміст цієї комірки. Залежно від виконуваної команди пристрій зчитування здатен змінювати вміст поточної комірки та зсуватися за один крок (на одну комірку) вліво або вправо.

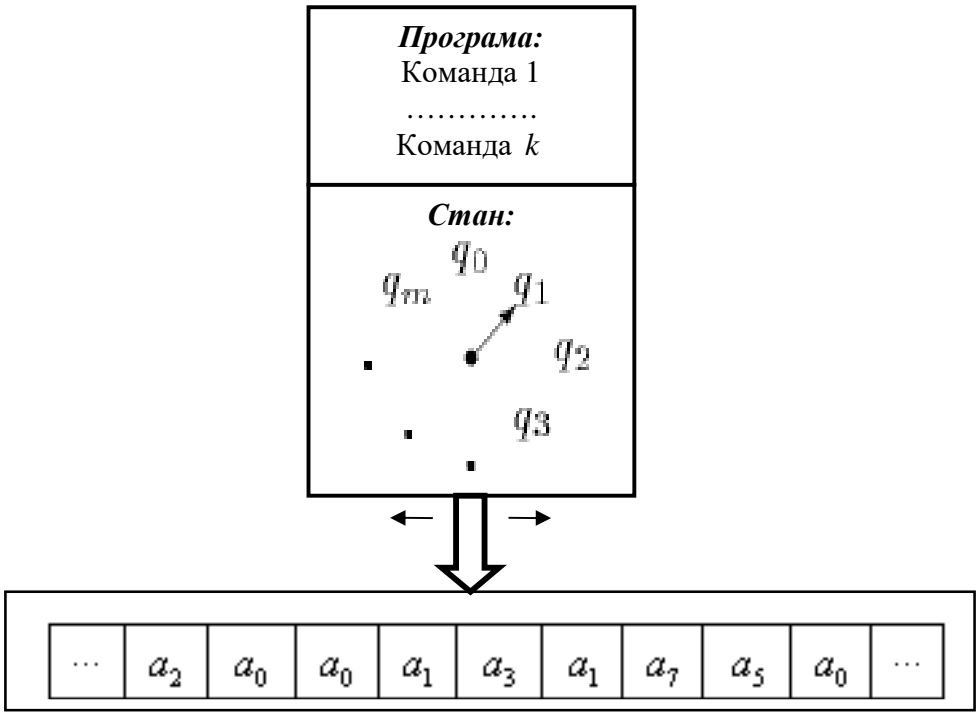


Рис. 1.8. Складові машини Тюрінга

Кожна машина Тюрінга має скінченне число станів $q_0, q_1, ..., q_m$, у яких вона може перебувати. Кожен стан може вказувати на два можливі напрямки дій: першого треба дотримуватися, якщо в комірці стрічки міститься 0, другого – якщо там міститься 1. У процесі роботи машина може декілька разів повертатися в один і той самий стан. Робота завжди починається з виділеного початкового стану. Крім того, є виділений кінцевий стан. Якщо машина потрапляє у кінцевий стан, то вона зупиняється. У будь-якому разі робота машини складається з трьох типів елементарних кроків:

- символ записується в поточну комірку, при цьому попередній символ стирається;
- стрічка зсувається на одну комірку вправо чи вліво;
- вказується інша інструкція.

Очевидно, що список станів може визначати деяку функцію переходів, яка за даним станом вказує на те, яку із трьох дій необхідно виконувати. Надалі припускається, що не існує двох правил переходу з однаковою лівою частиною, тобто машина детермінована. Машину Тюрінга можна описати мовою програмування з процедурною частиною та статичною і динамічною пам'яттю. Скінченна множина правил переходу еквівалентна процедурній частині. Внутрішні стани машини еквівалентні можливим станам статичної пам'яті. Стрічка машини виконує потрійну функцію: входу, виходу та динамічної пам'яті. Оскільки машина може робити записи на стрічці та переміщати її в обох напрямках, стрічку можна використовувати для зберігання довільного об'єму проміжних результатів. Величини переміщень стрічки не обмежені, тому довільний символ на стрічці може бути прочитаний машиною Тюрінга без зміни вмісту стрічки. Функції зчитування та запису машини Тюрінга можуть вилучати інформацію з довільної комірки та вводити інформацію у довільну комірку динамічної пам'яті. Таким чином, вищевикладене можна вважати дескриптивним визначенням машини Тюрінга.

Формальне трактування машина Тюрінга зводиться до впорядкованої п'ятірки

$$M = \left\langle A, Q, P, q_1, q_0 \right\rangle, \tag{1.1}$$

де $A = \{a_0, a_1, ..., a_n\}$ – скінченний зовнішній алфавіт (при цьому, що $n \geq 1, a_0 = 0, a_1 = 1$); $Q = \{q_0, q_1, ..., q_m\}$ – скінченний алфавіт внутрішніх станів; q_0 – початковий стан; q_m – кінцевий стан; $P = T\{i, j\}, 0 \leq i \leq m, 0 \leq j \leq n$ – програма, що складається з команд $T(i, j)$, кожна з яких є слово вигляду

$$q_i a_j \rightarrow a_l d q_k \ (0 \leq k \leq m, 0 \leq l \leq n), \tag{1.2}$$

де q_i – внутрішній стан машини; a_j – символ, що його зчитують; q_k – новий внутрішній стан; a_l – новий символ, що його записують; d – напрямок руху пристрою зчитування: *Л* – наліво; *П* – направо; *Е* – на місці. Крім того, передбачається, що кожній парі $q_i a_j$ відповідає точно одна команда. Множина таких команд називається програмою машини, а отже, в програмі є $m \times (n+1)$ команд.

Гіпотеза Тюрінга. Для знаходження значень функції, заданої в деякому алфавіті, тоді і лише тоді існує деякий алгоритм, коли функція обчислювана за Тюрінгом, тобто, коли її можна обчислити на відповідній машині Тюрінга.

Приклад 1.1. Нехай для обчислення деякої функції розроблено набір команд

$$T(0,1)=(1,П,0), \tag{1.3}$$
$$T(0,0)=(1,П,1), \tag{1.4}$$
$$T(1,1)=(1,П,1), \tag{1.5}$$
$$T(1,0)=(0,Л,2), \tag{1.6}$$
$$T(2,1)=(0,Л,3), \tag{1.7}$$
$$T(3,1)=(0,Л,4), \tag{1.8}$$
$$T(4,1)=(1,Л,4), \tag{1.9}$$
$$T(4,0)=(0,П,5). \tag{1.10}$$

Необхідно перевірити, чи функція обчислювана, тобто чи правильний алгоритм, якщо на стрічці записано слово 0**1**10110, а пристрій зчитування розміщений на першій позиції, де вказана одиниця.

Розв'язок. Заданий набір команд повністю визначає дії машини Тюрінга у всіх можливих ситуаціях і, тим самим, повністю задає поведінку абстрактного пристрою. Перевіримо, чи обчислюється задана функція шляхом виконання заданого набору команд. Наприклад,

розглянемо першу команду (1.3). З врахуванням загального правила запису команд (1.2) можна інтерпретувати команду (1.3) наступним чином

$q_i = 0$ – поточний внутрішній стан;

$a_j = 1$ – поточний символ алфавіту, що знаходиться в даний момент у комірці;

$a_l = 1$ – символ, який необхідно буде записати в поточну комірку, якщо поточний внутрішній стан машини Тюрінга $q_i = 0$, а попередньо в комірці знаходилась одиниця;

$d = П$ – після виконання команди пристрій зчитування переміщується на одну комірку вправо;

$q_k = 0$ – після виконання команди машина Тюрінга переходить у внутрішній стан, що дорівнює нулю.

Отже, якщо машина Тюрінга перебуває у внутрішньому стані, що дорівнює нулю, а поточний символ на стрічці одиниця, то необхідно символ в комірці залишити рівним одиниці, перемістится на одну комірку вправо і перейти у внутрішній стан, рівний одиниці.

Аналогічно інтерпретують і команди (1.4) – (1.10). Команди продовжують виконувати, поки не буде досягнуто кроку зупинки (кроку, який помічений як кінцевий). У момент зупинки мають виконуватися такі умови:

- всередині вхідного слова не має бути порожніх комірок;
- автомат повинен зупинися під одним із символів вихідного слова, а якщо слово пусте, то під довільною коміркою стрічки;

Може бути випадок, коли машина Тюрінга зациклюється, тобто ніколи не потрапляє на крок зупинки (наприклад, на кожному кроці автомат зсувається вправо і тому не може зупинитися, оскільки стрічка нескінченна). Тоді вважають, що машина Тюрінга не застосовна до заданого вхідного слова. Таким чином, обчислення або не обчислення функції залежить не лише від самого алгоритму, але й від вхідного слова. Тобто, один і той самий алгоритм можна застосовувати до одного вхідного слова і неможливо застосовувати для іншого. Отже послідовність дій для цього прикладу має бути такою.

1. Записуємо на стрічну вхідне слово 0**1**10110.
2. Встановлюємо автомат у початковий стан $q_0 = 0$.
3. Розміщуємо початковий стан $q_0 = 0$ над поточним заданим символом $a_j = 1$.
4. Оскільки зчитувальний пристрій розміщений над одиницею, а початковий стан q_0, то виконується команда (1.3). Після виконання команди послідовність на стрічці матиме вигляд: 01**1**0110. Наступною повинна спрацьовувати команда (1.3): Т(0,1)=(1,П,0).
5. Виконання команд продовжується, доки не буде досягнуто кроку зупинки.

Процес виконання команд та поточні результати для цього прикладу ілюструє табл. 1.1.

Таблиця 1.1- **Результати роботи машини Тюрінга**

Номер поточного внутрішнього стану	Поточний рядок символів	Команда	Отриманий рядок символів
0	0***1***10110	*T(0,1)=(1,П,0)*	01***1***0110

0	01***1***0110	*T(0,1)=(1,П,0)*	011***0***110
0	011***0***110	*T(0,0)=(1,П,1)*	0111***1***10
1	0111***1***10	*T(1,1)=(1,П,1)*	01111***1***0
1	01111***1***0	*T(1,1)=(1,П,1)*	011111***0***
1	011111***0***	*T(1,0)=(0,Л,2)*	01111***1***0
2	01111***1***0	*T(2,1)=(0,Л,3)*	0111***1***00
3	0111***1***00	*T(3,1)=(0,Л,4)*	011***1***000
4	011***1***000	*T(4,1)=(1,Л,4)*	01***1***1000
4	01***1***1000	*T(4,1)=(1,Л,4)*	0***1***11000
4	0***1***11000	*T(4,1)=(1,Л,4)*	***0***111000
4	***0***111000	*T(4,0)=(0,П,5)*	*0***1***11000

Машина Тюрінга дійшла до виконання останньої команди. Таким чином доведено, що функція обчислювана для даної послідовності символів 0***1***10110, а кінцева послідовність символів має вигляд: *0***1***11000.

Приклад 1.2. Нехай задано алфавіт $A=\{0,1,2,3,4,5,6,7,8,9\}$ і P – число, записане у десятковій системі. Необхідно написати алгоритм (набір команд машини Тюрінга) для отримання на стрічці числа, на одиницю більшого за задане число.

Розв'язок.

1. Для збільшення заданого числа на одиницю необхідно збільшити на одиницю останню цифру числа, заданого у десятковій системі. Для цього спочатку потрібно написати команди, які будуть переміщувати пристрій зчитування під останню цифру.
2. Введемо два стани: $q=1$ – це стан, в якому алфавіт розміщується під останньою цифрою числа, рухаючись постійно вправо, не змінюючи видимі цифри та залишаючись у тому ж стані; $q=2$ – стан, в якому машина додає одиницю до тієї цифри, яку «бачить» у поточний момент часу.
3. Стан зупинки позначимо символом «!».
4. Команди для переміщення пристрою зчитування під останню цифру заданого числа мають вигляд:

$$\begin{gathered} T(1,0)=(0,П,1),\\ T(1,1)=(1,П,1),\\ T(1,2)=(2,П,1),\\ T(1,3)=(3,П,1),\\ T(1,4)=(4,П,1),\\ T(1,5)=(5,П,1),\\ T(1,6)=(6,П,1),\\ T(1,7)=(7,П,1),\\ T(1,8)=(8,П,1),\\ T(1,9)=(9,П,1),\\ T(1,С)=(С,Л,2). \end{gathered} \qquad (1.11)$$

5. Остання інструкція (1.11) необхідна тому, що коли автомат розміщений під останньою цифрою, то він цього «не бачить», оскільки «не знає», що знаходиться в сусідніх комірках. Визначити це він може тільки тоді, коли натрапить на порожню комірку. Тому, доходячи до

першої пустої комірки, необхідно повернутися назад під останню цифру та перейти у інший стан. Такому поверненню і відповідає команда (1.11).

6. Якщо остання цифра заданого числа знайдена і приймає значення від 0 до 8, цю цифру заміняють на цифру, яка на одиницю більша і далі машина зупиняється.

7. Якщо остання цифра заданого числа дорівнює 9, машина Тюрінга заміняє її на 0 і зсуває вліво, залишаючись у стані 2. Таким чином, тепер одиниця буде додаватися до попередньої цифри, після чого процедура обчислень зупиняється.

8. Команди для додавання одиниці до числа, заданого у десятковій системі:

$$T(2,0)=(1,E,!), \quad (1.12)$$
$$T(2,1)=(2,E,!), \quad (1.13)$$
$$T(2,2)=(3,E,!), \quad (1.14)$$
$$T(2,3)=(4,E,!), \quad (1.15)$$
$$T(2,4)=(5,E,!), \quad (1.16)$$
$$T(2,5)=(6,E,!), \quad (1.17)$$
$$T(2,6)=(7,E,!), \quad (1.18)$$
$$T(2,7)=(8,E,!), \quad (1.19)$$
$$T(2,8)=(9,E,!), \quad (1.20)$$
$$T(2,9)=(0,Л,2), \quad (1.21)$$
$$T(2,C)=(1,E,!). \quad (1.22)$$

Набір команд (1.12) – (1.22) по суті і є алгоритмом розв'язання поставленої задачі. Для доведення його коректності необхідно виконати перевірку, як і в попередньому прикладі, чи обчислює задану послідовність символів алфавіту побудована машина Тюрінга. Уточнення уявлення про обчислення на основі поняття машини Тюрінга дало змогу довести алгоритмічну нерозв'язність різноманітних масових проблем, тобто проблем про знаходження єдиного методу розв'язання деякого класу задач, умови яких можуть варіювати у певних межах.

Машина Поста – це абстрактна обчислювальна машина, запропонована для уточнення та формалізації поняття «алгоритм». Еміль Пост у 1936 р. описав систему, що має алгоритмічну повноту та здатність визначати, чи є задача алгоритмічно розв'язною.

Машина Поста складається з нескінченної стрічки, розділеної на однакові комірки, що можуть бути або порожніми – 0 або містити мітку 1, та пристрою зчитування (каретки), здатної пересуватися по стрічці, перевіряти наявність мітки, стирати та записувати мітку.

Стан машини Поста описується станом стрічки та положенням каретки. Стан стрічки – це інформація про те, які комірки порожні, а які помічені. Крок – рух каретки на одну комірку вправо чи вліво. Кареткою керує програма, що складається із рядків команд, кожна з яких має наступний синтаксис

$$iKj,$$

де i – номер команди; K – дія каретки; j – номер наступної команди. Для машини Поста існує шість типів команд:

1) V_j – поставити мітку і перейти до j-го рядка програми;

2) X_j – стерти мітку і перейти до j-го рядка програми;

3) $<-j$ – переміститися вліво і перейти до j-го рядка програми;

4) $->j$ – переміститися вправо і перейти до j-го рядка програми;

5) $?j_1;j_2$ – якщо в комірці немає мітки, то перейти до j_1-го рядка програми, інакше перейти до j_2 рядка програми;

6) ! – завершення програми.

Вважається, що програма досягає свого завершення, якщо написаний алгоритм правильно виконано, або коли пристрій зчитування повинен записати чи стерти мітку там, де вона є чи її немає відповідно (тобто, задано виконання недопустимої команди), або в випадку зациклення – коли машина ніколи не досягає команди «стоп».

Елементарні дії машини Поста простіші за команди машини Тюрінга, тому для машини Поста існує більша кількість команд, ніж для машини Тюрінга.

На машині Поста можна виконувати різноманітні обчислення. Для машини Поста найбільш характерні такі особливості складання програм:

- досягнення заданої мети можуть забезпечити декілька програм;
- якщо розширюється клас початкових даних, то побудова такої програми стає більш важкою;
- для побудови програм з розв'язання складних задач більш прості програми використовують як складові;
- під час написання програм слід враховувати якими числами необхідно буде оперувати, а також розміщеня цих чисел в пам'яті;
- тенденція до мінімізації часу виконання програми.

Приклад 1.3. Нехай на стрічці задано число n та набір команд машини Поста, що збільшує задане число на одиницю. Потрібно довести коректність заданого алгоритму, якщо задане число – 3. Команди машини Поста:

1.1 -> 2 (1.23)
2.2 ? 1;3 (1.24)
3.3 <- 4 (1.25)
4.4 V 5 (1.26)
5.5 ! (1.27)

Розв'язок. Ціле додатне число на стрічці Поста подають мітками, що йдуть підряд і кількість яких на одиницю більша, ніж число, що кодується. Це зумовлено тим, що одна мітка позначає 0, а дві – одиницю і т.д.

Припустимо, що каретка розміщена зліва від міток і бачить порожню комірку. Застосуємо задані команди для перевірки правильності алгоритму (табл. 1.2). Обчислення починаються з першої команди, тобто з (1.23), якщо в умові немає ніяких додаткових пояснень.

Таким чином, остання команда – це закінчення виконання алгоритму, на стрічці отримано число 4 (послідовність 11111).

Таблиця 1.2-**Результати виконання команд машини Поста**

Поточний рядок символів	Команда	Отриманий рядок символів
0001111	**1.**1 -> 2	0***0***01111
0***0***01111	**2.**2 ? 1;3	0***0***01111
0***0***01111	**1.**1 -> 2	00***0***1111
00***0***1111	**2.**2 ? 1;3	00***0***1111
00***0***1111	**1.**1 -> 2	000***1***111

000***1***111	**2.**2 ? 1;3	000***1***111
000***1***111	**3.**3 <- 4	00***0***1111
00***0***1111	**4.**4 V 5	00***1***1111

Нормальні алгоритми Маркова – це система послідовних застосувань підстановок, які реалізують певні процедури отримання нових слів з базових, побудованих із символів деякого алфавіту. Як і машини Тюрінга, нормальні алгоритми не виконують самих обчислень, вони лише перетворюють слова заміною літер за заданими правилами.

Нормально обчислюваною називають функцію, яку можна реалізувати нормальним алгоритмом. Тобто, алгоритмом, який кожне слово з множини допустимих даних функції перетворює на її вихідні значення.

Творець теорії нормальних алгоритмів А. А. Марков в 40-вих роках минулого століття висунув гіпотезу, яка отримала назву принципу нормалізації Маркова: для знаходження значень функції, заданої в деякому алфавіті, тоді і лише тоді існує деякий алгоритм, коли функція нормально обчислювана. Подібно до гіпотез Тюрінга та Черча, принцип нормалізації Маркова не може бути доведений математичними засобами.

Нормальні алгоритми є вербальними, тобто призначені для застосування до слів у різноманітних алфавітах. Визначення довільного нормального алгоритму складається з двох частин: визначення алфавіту алгоритму та визначення його схеми. Схемою нормального алгоритму називають скінченний упорядкований набір формул підстановки, кожна з яких може бути простою або завершальною. Простими формулами підстановки називають слова вигляду $L \to D$, де L і D – два довільні слова в алфавіті алгоритму (відповідно ліва і права частини формули). $L \to \cdot D$ – кінцева формула.

Процес застосування нормального алгоритму до довільного слова є дискретною послідовністю елементарних кроків, які полягають у наступному. Нехай V – слово, отримане на попередньому кроці роботи алгоритму або початкове слово. Якщо серед формул підстановки немає такої, ліва частина якої входила б в V, то обирають верхню формулу, після чого слово вважається результатом поточного кроку і підлягає подальшому обробленню на наступному кроці.

Нехай є початковий рядок «Я купив кг А у Т М.» і задані правила підстановки: «А» → «апельсин»; «кг» → «кілограм»; «М»→ «магазині»; «Т»→ «той»; «тому магазині»→ .«тій точці».

В процесі виконання алгоритму з рядком відбуваються такі зміни:

1. «Я купив кг апельсинів у Т М.»
2. «Я купив кілограм апельсинів у Т М.»
3. «Я купив кілограм апельсинів у Т магазині.»
4. «Я купив кілограм апельсинів у тому магазині.»
5. «Я купив кілограм апельсинів у тій точці.»

Нехай дано алгоритм перетворення двійкових чисел в одиничні, тобто на виході повинні отримати рядок із N одиниць, якщо на вході було N у двійковій системі. Наприклад, 101.

Правила перетворення:

1. «|0» → "011"
2. «1» → "01"
3. «0» → ""

Виконання**:**

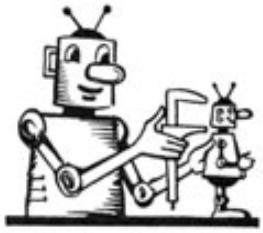
1. «0101»
2. «00111»
3. "001101"
4. "0010111"
5. "00011111"
6. "0011111"
7. "0111111"
8. "11111"

Довільний нормальний алгоритм еквівалентний деякій машині Тюрінга і навпаки – довільна машина Тюрінга еквівалентна деякому нормальному алгоритму. Нормальні алгоритми виявилися зручним засобом для побудови багатьох розділів конструктивної математики. Крім того, закладені у визначенні нормального алгоритму ідеї використовуються у ряді мов програмування, орієнтованих на оброблення символьної інформації.

Контрольні запитання та завдання до розділу 1

1. Що таке алгоритм? Області застосування алгоритмів.
2. Що включає в себе теорія алгоритмів? Які її основні задачі?
3. Які властивості алгоритмів розрізняють?
4. Що таке алгоритмічна мова?
5. Як класифікують алгоритми?
6. Принципова різниця між формальними та не формальними алгоритмами?
7. Що таке блок-схемний спосіб задання алгоритмів?
8. Що таке лінійні алгоритми?
9. Що таке розгалужені алгоритми?
10. Що таке циклічні алгоритми?
11. Основні етапи побудови алгоритмів?
12. Що таке машина Тюрінга?
13. З яких компонентів складається машина Тюрінга?
16. Як машину Тюрінга задають в загальному вигляді?
17. Як формулюється гіпотеза Тюрінга?
18. Що представляють собою команди в машині Тюрінга?
19. Нехай для обчислення деякої функції розробили набір команд для машини Тюрінга: *T(0,1)=(1,П,0); T(0,0)=(1,П,1); T(1,1)=(1,П,1); T(1,0)=(0,Л,2); T(2,1)=(0,Л,3); T(3,1)=(0,Л,4); T(4,1)=(1,Л,4); T(4,0)=(0,П,5).* Чи обчислювана функція, тобто чи правильний алгоритм, якщо код, записаний на стрічці 01**1**0110?
20. Що таке машина Поста?
21. Які типи команд характерні для машини Поста?
22. Чим відрізняється машина Тюрінга від машини Поста?
23. Чи можна задану програму застосовувати до заданих станів машини Поста, якщо початковий стан стрічки: 1110011?

1. ? 3; 2	6. → 7
2. → 1	7. ? 8; 9
3. → 4	8. !
4. ? 6; 5	9. → 4
5. ← 1	



РОЗДІЛ 2. НАБЛИЖЕНІ ЧИСЛА. ОБЧИСЛЕННЯ ФУНКЦІЙ ЗА ДОПОМОГОЮ СТЕПЕНЕВИХ РЯДІВ

2.1. Наближені числа, джерела виникнення й класифікація похибок

Математичний опис процесу розв'язку довільної науково-технічної задачі зводиться до опису послідовності арифметичних операцій, пов'язаних між собою логічними умовами. Сучасні комп'ютери можуть виконувати лише досить прості операції. Тому для того, щоб вказати алгоритм обчислення арифметичного виразу, в загальному випадку необхідно визначити порядок виконання операцій. Для точних методів це не має особливого значення, однак, в процесі комп'ютерної реалізації різних форм запису алгоритмів арифметичних обчислень виникають різні побічні ефекти. Це зумовлено округленням результатів до певної кількості розрядів, яка прийнята у даній системі обчислень. Точні операції додавання та множення є комутативними, асоціативними і пов'язані між собою законом дистрибутивності. У разі виконання їх на комп'ютері ці операції, строго кажучи, такими вже не є. Відповідно, різне розставлення дужок в одному і тому ж арифметичному виразі в процесі його обчислення на комп'ютері може приводити до різних результатів. Таким чином, довільне завдання при постановці його на комп'ютері визначає множину обчислювальних алгоритмів, що відрізняються один від одного порядком виконання арифметичних дій. Незважаючи на математичну еквівалентність всіх цих модифікацій, їх відмінності в обчислювальному сенсі можуть бути дуже великими.

Похибка – величина, що характеризує ступінь наближення точних та приблизних значень величин.

Наближеним числом (наближенням) a називають число, яке незначно відрізняється від точного числа A (у загальному випадку невідомого) та заміняє його впроцесі обчислень.

Похибкою наближеного числа a називається різниця (A - a) між точним числом A та його наближенням a.

Якщо $a < A$, то наближення називають ***наближенням з недостачею***, а якщо $a > A$ – ***наближенням з надлишком***.

У випадку, коли знак похибки невідомий, користуються абсолютною похибкою наближеного числа.

Абсолютна похибка Δa наближеного числа a - це абсолютна величина різниці між точним числом A та числом a, тобто $\Delta a = |A - a|$.

Якщо точне значення A невідоме, то вводять верхню оцінка абсолютної похибки, яка називається граничною абсолютною похибкою.

Гранична абсолютна похибка наближеного числа – це довільне додатне число, що не менше за абсолютну похибку цього числа $\overline{\Delta}(a) \geq \Delta a$, де $\overline{\Delta}(a)$ – гранична абсолютна похибка.

Нехай потрібно визначити граничну абсолютну похибку числа $a = 3{,}14$, що заміняє π. Оскільки відомо, що $3{,}14 < \pi < 3{,}15$, то $|\pi - a| < 0{,}01$ і відповідно $\overline{\Delta}(a) = 0{,}01$.

Абсолютна величина похибки не повністю характеризує ступінь наближеності наближеного числа.

Відносною похибкою наближеного числа a називається відношення абсолютної похибки $\Delta(a)$ до абсолютної величини точного числа $A\,(A \neq 0)$

$$\delta(a) = \frac{\Delta(a)}{|A|}.$$

Граничною відносною похибкою $\overline{\delta}(a)$ наближеного числа a називається довільне додатне число, не менше відносної похибки цього числа

$$\overline{\delta}(a) \geq \delta(a),$$

тобто $\frac{\Delta(a)}{|A|} \leq \overline{\delta}(a)$, звідки $\Delta(a) \leq |A|\,\overline{\delta}(a)$. Тоді можна припустити, що $\overline{\Delta}(a) = |A|\,\overline{\delta}(a)$.

Похибка розв'язання задачі може бути зумовлена такими причинами:

- неточним математичним описом задачі, зокрема неточним заданням початкових даних;
- неточним методом, що застосовується для розв'язання, оскільки отримання точного розв'язку потребує великої або нескінченної кількості арифметичних операцій;
- операціями округлення при введенні/виведенні даних та в процесі виконання інших обчислювальних операцій.

Отже, за джерелом виникнення похибки класифікують на три класи:

1) неусувна похибка;
2) похибка методу;
3) обчислювальна похибка.

Розглянемо обчислювальну похибку більш детально.

Множина F чисел з рухомою (або так званою «плаваючою») точкою (комою) характеризується чотирма параметрами: основою p, точністю t та інтервалом порядку $[L,U]$. Кожне число x з рухомою точкою, яке належить до F, можна подати у вигляді

$$x = \pm\left(\frac{a_1}{p} + \frac{a_2}{p^2} + \ldots + \frac{a_t}{p^t}\right)p^l,$$

де цілі числа $a_1, a_2, \ldots, a_t$ задовольняють нерівності $0 \leq a_i \leq p-1$, $i = 1,2,\ldots,t$, $L \leq l \leq U$.

Якщо для кожного $x \neq 0$ із множини F має місце $a_1 \neq 0$, то система чисел F з рухомою точкою називається нормалізованою. Число $g = \pm\left(\frac{a_1}{p} + \frac{a_2}{p^2} + ... + \frac{a_t}{p^t}\right)$ називається дробовою частиною, або мантисою, а ціле число l – показником (порядком, експонентою).

Множина F не є континуумом або навіть нескінченною множиною. Вона включає рівно $2(p-1)\cdot p^{t-1}(U-L+1)+1$ чисел. Оскільки F – скінченна множина, то за її допомогою відобразити континуум дійсних чисел можна лише більш-менш задовільно. Неможливо також відобразити дійсні числа за абсолютною величиною більші, ніж максимальний елемент множини F, та ненульові дійсниі числа, за абсолютною величиною менші за найменший елемент F. Таким чином, кожне число з множини F відповідає деякому інтервалу дійсних чисел.

Оскільки відстань між двома сусідніми числами в F, тобто крок h_{fp} подання чисел в арифметиці з рухомою точкою становить $h_{fp} = p^{-t+l}$, то абсолютна похибка подання чисел в ній дорівнює $\Delta_{fp} \le \frac{h_{fp}}{2} = \frac{1}{2}p^{-t+l}$. Якщо $p = 2$, відповідно маємо $\Delta_{fp} \le 2^{-t+l-1}$. Тоді відносна похибка δ_{fp} подання дійсного числа $x = \pm g \cdot p^l$ дорівнює $\delta_{fp} = \frac{\Delta_{fp}}{|\pm x|} \le \frac{p^{-t+l}}{2|x|} = \frac{p^{-t+l}}{2g\cdot p^l} = \frac{p^t}{2g}$ і досягає максимального значення $\delta_{\max} = \frac{p^{-t}}{2p^{-1}} = \frac{1}{2}p^{1-t}$ при $g = g_{\min} = p^{-1}$. Якщо $p = 2$, то $\delta_{\max} = 2^{-t}$.

Мінімальне значення відносної похибки досягається при $g = g_{\min} = 1 - p^{-t} \approx 1$ і дорівнює $\delta_{\min} = \frac{p^{-t}}{2(1-p^{-t})} \approx \frac{1}{2}p^{-t}$, при $p = 2$ відносна похибка дорівнює $\delta_{\min} = 2^{-t-1}$. Оскільки, як правило, $t >> 1$, то можна вважати, що відносна помилка постійна на всьому діапазоні подання чисел і дорівнює 2^{-t} при $p = 2$.

Сформулюємо основні теореми про похибки.

Теорема 1. Абсолютна похибка Δs алгебраїчної суми $s = \sum_{i=1}^{n} a_i$ декількох наближених чисел не перевищує суми абсолютних похибок доданків

$$\Delta s \le \sum_{i=1}^{n} \Delta a_i. \qquad (2.1)$$

Очевидно, якщо є один доданок, абсолютна похибка якого значно перевищує абсолютні похибки решти, то абсолютну похибку суми можна вважати абсолютною похибкою цього доданка.

При великій кількості доданків оцінка (2.1) виявляється завищеною через часткову компенсацію похибок різних знаків, тому при $n > 10$ використовують ***статистичну*** оцінку похибки. Тоді

$$\Delta s \le \sqrt{3n}\cdot \Delta a_k,$$

де Δa_k – найбільша з абсолютних похибок серед всіх доданків.

Теорема 2. Відносна похибка добутку $P = \prod_{i=1}^{n} a_i$ не перевищує суми відносних похибок множників $\delta p \le \sum_{i=1}^{n} \delta a_i$. При великій кількості множників $\delta p \le \sqrt{3n}\delta a_k$, де δa_k – найбільша з відносних похибок серед всіх множників.

Теорема 3. Відносна похибка S-го степеня наближеного числа в S разів більша за відносну похибку основи (як для цілих, так і для дробових S).

Теорема 4. Похибка обчислення значень функції $f(x_1, x_2, ..., x_n)$ декількох змінних, аргументи якої задані наближено, оцінюється за допомогою її диференціала

$$\Delta f \le \sum_{i=1}^{n} \left| \frac{\partial f}{\partial x_i} \right| \Delta x_i .$$

В теорії похибок розрізняють задачі двох типів.

1. Знаючи похибку початкових даних, визначити похибку результату (***пряма задача***).
2. За заданою похибкою результату визначити, з якою похибкою необхідно брати початкові дані, щоб забезпечити потрібну похибку результату (***обернена задача***).

Як приклад прямої задачі розглянемо обчислення похибки довільної суперпозиції алгебраїчних дій

$$A = \sqrt{\frac{a}{a+b}}.$$

Послідовно застосовуючи теореми про похибки, маємо

$$\delta A \le \frac{1}{2}(\delta a + \delta a + b) \le \frac{1}{2}\left(\frac{\Delta a}{A} + \frac{\Delta a + \Delta b}{|a+b|}\right),$$

$$\Delta A \le \frac{1}{2}\left(\frac{\Delta a}{|a|} + \frac{\Delta a + \Delta b}{|a+b|}\right)\left|\sqrt{\frac{a}{a+b}}\right|.$$

Як приклад оберненої задачі розглянемо наступну. Сторона квадрата приблизно дорівнює 1м. З якою точністю її необхідно виміряти, щоб похибка обчислення площі не перевищувала 1см^2?

Площа квадрата S обчислюється як $(a+\Delta a)^2$, де $(a+\Delta a)$ – приблизна довжина сторони квадрата. Отже $S = (a+\Delta a)^2 = a^2 + 2a\Delta a + (\Delta a)^2$. Оскільки $(\Delta a)^2$ є величиною другого порядку малості, нею можна знехтувати. Тоді похибка вимірювання площі відносно величини першого порядку малості буде $2a\Delta a = 10^{-4}, \Delta a = 5 \cdot 10^{-5}$.

2.2. Обчислення значень поліномів. Схема Горнера

Поліномом або многочленом однієї змінної називається вираз

$$P(x) = a_0 x^n + a_1 x^{n-1} + ... + a_{n-1} x + a_n, \qquad (2.2)$$

де $a_0, a_1, ..., a_n$ – дійсні коефіцієнти, x – змінна.

Поліномом (многочленом) від декількох змінних називається скінченна сума, в якій кожен з доданків є добутком скінченного числа цілих степенів змінних та константи

$$P(x) = a_0 + a_1 xy^2 + a_2 z^3 + a_3 xyz + ... \qquad (2.3)$$

Поліноми є одним із найважливіших класів елементарних функцій.

Схема Горнера – це алгоритм обчислення значення полінома, записаного у вигляді суми мономів (одночленів) за заданого значення змінної. Схема Горнера є також простим алгоритмом для ділення многочлена на біном.

Нехай задано поліном

$$P(x) = a_0 x^n + a_1 x^{n-1} + ... + a_{n-1} x + a_n,$$

де $(a_1, a_2, ..., a_n)$ – дійсні коефіцієнти. Необхідно обчислити значення многочлена, якщо $x = x_0$. Подамо поліном у такому вигляді

$$P(x) = ((...(((a_0 x + a_1)x + a_2)x + a_3)x + ... + a_{n-1})x + a_n). \qquad (2.4)$$

Тоді можна послідовно обчислити

$$\begin{cases} b_0 = a_0, \\ b_1 = a_1 + b_0 x, \\ b_2 = a_2 + b_1 x, \\ ..... \\ b_n = a_n + b_{n-1} x. \end{cases} \qquad (2.5)$$

Шукане значення $P(x_0) = b_n$. Формули (2.5) дозволяють проводити практичні обчислення за такою схемою, яка і називається ***схемою Горнера.***

(2.6)

a_0	a_1	a_2	…	a_n
+	+	+	…	+
0	b_0x_0	b_1x_0	…	$b_{n-1}x_0$
b_0	b_1	b_2	…	$b_n=P(x_0)$

Приклад 2.1. Нехай потрібно обчислити значення полінома $P(x) = 3x^3 + 2x^2 - 5x + 7$, якщо x_0=3.

Розв'язок. Складемо схему Горнера, підставляючи значення коефіцієнтів заданого полінома в (2.6).

3	2	-5	7
+	+	+	+
0	3×3	11×3	28×3
3	11	28	91=*P(x0)*

Схему Горнера можна застосовувати для ділення полінома на біном, тобто для обчислення частки та залишку від ділення поліному $P(x)=a_0x^n+a_1x^{n-1}+...+a_{n-1}x+a_n$ на лінійний двочлен $(x-s)$. Схему ділення за Горнером можна подати у вигляді табл. 2.1.

Таблиця 2.1- **Схема Горнера для ділення полінома на біном**

s_i	Коефіцієнти полінома				
	a_n	a_{n-1}	a_{n-2}	…	a_0
s	$a_n=b_n$	$b_{n-1}=a_{n-1}+b_ns$	$b_{n-2}=a_{n-2}+b_{n-1}s$	…	$b_0=a_0+b_1s$

Отримані числа $b_n,b_{n-1},b_{n-2},...,b_1$ – коефіцієнти частки від ділення полінома на двочлен, а b_0 – залишок, тобто

$$P(x)=a_0+a_1x+a_2x^2+a_3x^3+...+a_nx^n=(x-s)(b_nx^{n-1}+b_{n-1}x^{n-2}+...+b_1)+b_0. \qquad (2.7)$$

Приклад 2.2. Знайти частку та залишок від ділення полінома $2x^4-3x^3-x^2+4x+13$ на лінійний двочлен $(x-1)$.

Розв'язок. У даному прикладі $s=1$, коефіцієнти $a_4=2$, $a_3=-3$, $a_2=-1$, $a_1=4$, $a_0=13$. Використовуючи схему Горнера, заповнимо таблицю (табл. 2.2).

Таблиця 2.2.

s_i	Коефіцієнти полінома				
	$a_4=2$	$a_3=-3$	$a_2=-1$	$a_1=4$	$a_0=13$
s	$a_4=b_4=2$	$b_3=a_3+b_4\cdot s=$ $=-3+2\cdot 1=-1$	$b_2=a_2+b_3\cdot s=$ $=-1-1\cdot 1=-2$	$b_1=a_1+b_2\cdot s=$ $=4-2\cdot 1=2$	$b_0=a_0+b_1\cdot s=$ $=13+2\cdot 1=15$

Таким чином, $b_4x^3+b_3x^2+b_2x+b_1=2x^3-x^2-2x+2$ – частка, $b_0=15$ – залишок від ділення.

2.3. Обчислення елементарних функцій за допомогою степеневих рядів

Одним з методів обчислення елементарних функцій є розкладання функції у степеневий ряд. Головним при цьому є визначення умов збіжності ряду та оцінка похибки наближення функції степеневим рядом.

Слід розрізняти методичну похибку наближення функції степеневим рядом, що зумовлена зрізанням нескінченного ряду, в який розкладають функцію, та обчислювальну похибку, яка виникає під час виконання операцій округлення.

Дійсна функція $f(x)$ називається ***аналітичною у точці*** ξ, якщо в деякому околі $|x-\xi| < R$ цієї точки функція розкладається у степеневий ряд Тейлора

$$f(x) = f(\xi) + f'(\xi)(x-\xi) + \frac{f''(\xi)}{2!}(x-\xi)^2 + \ldots + \frac{f^{(n)}(\xi)}{n!}(x-\xi)^n. \qquad (2.7)$$

Якщо $\xi = 0$ отримаємо ряд Маклорена

$$f(x) = f(0) + f'(0)x + \frac{f''(0)}{2!}x^2 + \ldots + \frac{f^{(n)}(0)}{n!}x^n. \qquad (2.8)$$

Залишковий член, що визначає похибку наближення функції $f(x)$ степеневим рядом (2.7), дорівнює

$$R_n(x) = f(x) - \sum_{k=0}^{n} \frac{f^{(k)}(\xi)}{k!}(x-\xi)^k = \frac{f^{(n+1)}(\xi + \theta(x-\xi))}{(n+1)!}(x-\xi)^{n+1},$$

де $0 < \theta < 1$. Для ряду (2.9) залишковим членом є

$$R_n(x) = \frac{f^{(n+1)}(\theta \cdot x)}{(n+1)!}x^{n+1},$$

де $0 < \theta < 1$. Залишковий член ряду Тейлора характеризується тим, що він досить швидко спадає з наближенням x до ξ.

Якщо значення $f(\xi)$ відоме й необхідно знайти значення $f(\xi + h)$, де h – «мала поправка», то (2.7) можна переписати у вигляді

$$f(\xi + h) = f(\xi) + f'(\xi)h + \frac{f''(\xi)}{2!}h^2 + \ldots + \frac{f^{(n)}(\xi)}{n!}h^n + R_n(h),$$

де $R_n(x) = \frac{f^{(n+1)}(\xi + \theta \cdot x)}{(n+1)!}h^{n+1}$, $0 < \theta < 1$.

Для розв'язання задач у реальному часі (наприклад, у спеціалізованих процесорах) природним є прагнення утримувати мінімальну кількість членів ряду, які забезпечують потрібну точність значень функції. Для цього необхідно заздалегідь дослідити степеневий ряд з метою визначення залежності похибки наближення функції степеневим рядом від кількості членів ряду, діапазону зміни аргументу функції тощо.

Для експоненціальної функції e^x розклад у ряд Маклорена має вигляд

$$e^x = 1 + x + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!}. \qquad (2.9)$$

Інтервалом збіжності цього ряду є $-\infty < x < \infty$, а граничне значення залишкового члена ряду

$$R_n(x) = \frac{e^{\theta x}}{(n+1)!}x^{n+1}, \; 0 < \theta < 1.$$

Якщо аргумент x набуває великих (за абсолютною величиною) значень, то ряд (2.9) є дуже громіздким і малопридатним для практичних обчислень. Тому x подають як суму $x=[x]+\{x\}$, де $[x]$ – ціла частина числа x, а $0\le\{x\}<1$ – його дробова частина. Тоді

$$e^x = e^{[x]}\cdot e^{\{x\}}.$$

Перший множник $e^{[x]}$ отримують перемноженням самої на себе $[x]$ разів константи e – при $x>0$, або константи $1/e$ – при $x<0$.

Розглянемо тепер обчислення другого множника $e^{\{x\}}$, тобто випадок, коли $|x|<1$.

Позначаючи як $U_k(x)$ загальний член розкладу (2.10), отримаємо

$$e^x \approx S_n(x) = \sum_{k=0}^{n} U_k(x), \qquad (2.10)$$

де $U_0=1$, $S_0(x)=1$, а всі наступні U_k та S_k обчислюють за рекурентними співвідношеннями

$$U_k(x)=\frac{x}{k}U_{k-1}(x),\ S_k(x)=S_{k-1}(x)+U_k(x),\ k\ge 1. \qquad (2.11)$$

Залишковим членом ряду (2.10) є

$$|R_n(x)| = \sum_{k=n+1}^{\infty}\left|\frac{x^k}{k!}\right| \le \frac{|x|^{n+1}}{(n+1)!}+\frac{|x|^{n+2}}{(n+2)!}+\ldots < \frac{|x|^{n+1}}{(n+1)!}\left(1+\frac{|x|}{n+2}+\frac{|x|^2}{(n+2)^2}+\ldots\right) =$$

$$= \frac{|x|^{n+1}}{(n+1)!}\cdot\frac{1}{1-\frac{|x|}{n+2}} < \frac{|x|^{n+1}}{(n+1)!}\cdot\frac{n+1}{n}.$$

Звідси $|R_n(x)|<\frac{|x|^n}{n!}\cdot\frac{|x|}{n}<U_n$.

Отже, процес додавання можна припинити, як тільки черговий член ряду (2.10) стане за абсолютною величиною меншим за задану похибку ε, тобто, коли $|U_n|\le\varepsilon$.

Що стосується обсягу обчислень, то рекурентні співвідношення (2.11) набагато ефективніші, ніж безпосереднє обчислення за (2.9).

Для обчислення функцій синуса та косинуса за допомогою формул зведення аргумент x має належати відрізку $0\le x\le\frac{\pi}{2}$. Якщо $0\le x\le\frac{\pi}{4}$, тоді

$$\sin x = \sum_{n=0}^{\infty}(-1)^n\frac{x^{2n+1}}{(2n+1)!}, \qquad (2.12)$$

а якщо $\frac{\pi}{4}\le x\le\frac{\pi}{2}$, то

$$\sin x=\cos z=\sum_{n=0}^{\infty}(-1)^n\frac{z^{2n}}{(2n)!}, \qquad (2.13)$$

де $z = \frac{\pi}{2} - x$ і $0 \le z \le \frac{\pi}{4}$. Ряд (2.12) обчислюють з використанням рекурентних співвідношень

$$\sin x = U_1 + U_2 + ... + U_n + R_n,$$

де $U_1 = x$, $U_{k+1} = -\frac{x^2}{2k(2k+1)} U_k$, $k = 1, 2, ..., n-1$. Оскільки ряд (2.12) є знакозмінним і значення його членів спадають за абсолютною величиною, то для R_n доречною є оцінка

$$| R_n | \le \frac{x^{2n+1}}{(2n+1)!} = | U_{n+1} |.$$

Тобто, процес додавання можна припинити, коли $| U_{n+1} | < \varepsilon$.

Аналогічно для ряду (2.13) маємо

$$\cos z = U_1 + U_2 + ... + U_n + R_n,$$

де $U_1 = 1$, $U_{k+1} = -\frac{x^2}{2k \cdot (2k-1)} U_k$, $k = 1, 2, ..., n-1$,

$$| R_n | \le \frac{z^{2n}}{(2n)!} = | U_{n+1} |$$

Для обчислення натурального логарифма додатного числа x спочатку це число подають у вигляді

$$x = 2^m z,$$

де m – ціле число і $1/2 < z < 1$. Далі припускають, що $a = \frac{1-z}{1+z}$, де $0 < a < 1/3$. Тепер натуральний логарифм можна обчислити як

$$\ln x = \ln 2^m z = m \ln 2 + \ln z = m \ln 2 - 2(a + \frac{a^3}{3} + ... + \frac{a^{2n-1}}{2n-1}) - R_n.$$

Позначаючи $L_k = \frac{a^{2k-1}}{2k-1}$, $k = 1, 2, ..., n$, отримаємо

$$\ln x = m \ln 2 - 2(L_1 + L_2 + ... + L_n) - R_n$$

Для $\ln 2$ існує відповідна бібліотечна константа. Процес додавання припиняють, коли $L_n < 4\varepsilon$.

Перед застосуванням формул наближення елементарних функцій за допомогою степеневих рядів необхідно звести аргумент x до проміжку найшвидшої збіжності скориставшись однією з таких формул:

$$x = [x] + \{x\};$$
$$\sin(\pi \pm x) = \pm \sin x;$$
$$\sin(\pi / 2 - x) = \cos x;$$
$$x = 2^m z.$$

Інакше при підсумовуванні доданків, які відрізняються на декілька порядків, похибка округлення за величиною стає сумірною зі значенням результату.

Гіперболічні функції обчислюють таким чином. Гіперболічний синус визначають як $shx = \frac{e^x - e^{-x}}{2}$, причому $sh(-x) = -shx$. Цю функцію розкладають в ряд

$$shx = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \ldots .$$

Вважаючи, що $x > 0$, обчислення гіперболічного синуса виконують за формулою

$$shx = U_1 + U_2 + U_3 + \ldots + U_n + R_n,$$

де де $U_1 = 1, U_{k+1} = -\frac{x^2}{2k(2k-1)} U_k$, $k = 1, 2, \ldots, n-1$, а R_n – залишковий член ряду, причому

$R_n < \frac{1}{3} U_n$.

Гіперболічний косинус визначають як $chx = \frac{e^x + e^{-x}}{2}$, причому $ch(-x) = chx.$ Цю функцію розкладають в ряд

$$chx = x + \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots$$

Обчислення гіперболічного косинуса виконують за формулою

$$chx = U_1 + U_2 + U_3 + \ldots + U_n + R_n,$$

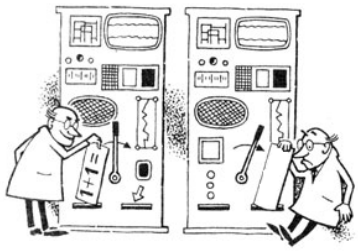
де $U_1 = 1$, $U_{k+1} = \frac{x^2}{2k(2k-1)} U_k$, $k = 1, 2, \ldots, n-1$, а R_n – залишковий член ряду, причому

$R_n < \frac{2}{3} U_n$.

Контрольні запитання та завдання до розділу 2

1. Дати визначення наближеного числа.
2. Що таке похибка?
3. Дати визначення абсолютної й відносної похибки.
4. Як поділяють похибки в залежності від джерела їх виникнення?
5. Які існують основні теореми про похибки?
6. У чому полягає суть прямої та оберненої задач теорії похибок?
7. Яку функцію називають аналітичною?
8. Що таке схема Горнера та її основне призначення?
9. За схемою Горнера обчислити значення полінома $P(x) = 2x^4 + x^2 + 3x - 1$ при $x = 2$.
10. За допомогою схеми схеми Горнера знайти частку та залишок від ділення полінома з попереднього прикладу на двочлен $x + 2$.
11. В чому полягає загальна методика обчислення функцій із застосуванням рядів?

12. Якими чинниками визначається методична похибка наближення функції степеневим рядом?
13. Чому дорівнює методична похибка наближення функції спадним знакозмінним рядом?
14. Чому дорівнює методична похибка наближення функції спадним знакопостійним рядом?
15. Записати вираз для розкладання функції в ряд Тейлора.
16. Чим відрізняється ряд Тейлора від ряду Маклорена?
17. Розкласти в ряд Маклорена функції e^x, shx, chx.



РОЗДІЛ 3. МЕТОДИ НАБЛИЖЕНОГО РОЗВ'ЯЗАННЯ АЛГЕБРАЇЧНИХ ТА ТРАНСЦЕНДЕНТНИХ РІВНЯНЬ

3.1. Відокремлення коренів рівняння

Нехай задано дійсну функцію $f(x)$, яка є визначеною й неперервною на деякому скінченному або нескінечнному інтервалі $a < x < b$. Необхідно знайти корені рівняння

$$f(x) = 0. \qquad (3.1)$$

Кожне значення x, яке перетворює $f(x)$ на нуль, називають коренем рівняння (3.1).

Наближені корені рівняння знаходять у два етапи: спочатку корені відокремлюють, тобто визначають по можливості малі відрізки, на яких міститься точно один корінь рівняння, а потім початкове наближення (значення всередині знайденого відрізка) уточнюють тим чи іншим способом, поки не буде виконуватись нерівність $| x_k - \xi | < \varepsilon$, де ξ – точний корінь рівняння, x_k – k -е наближення до нього, ε – задана похибка (визначає точність).

Розрізняють аналітичний та графічний способи відокремлення коренів рівняння.

Аналітичний спосіб ґрунтується на такій теоремі аналізу. Якщо неперервна функція $f(x)$ на кінцях відрізка $[a, b]$ набуває значень різних знаків, тобто $f(a)f(b) < 0$, то всередині цього відрізка міститься принаймні один корінь рівняння $f(x) = 0$, тобто знайдеться хоча б одне число $\xi \in [a,\ b]$ таке, що $f(\xi) = 0$. Корінь гарантовано буде тільки один, якщо похідна $f'(x)$ існує та зберігає сталий знак всередині інтервалу $[a,\ b]$, тобто якщо $f'(x) > 0$ або $f'(x) < 0$ при $a \le x \le b$. Отже, відокремлення коренів аналітичним способом зводиться до визначення відрізків $[a_i,\ b_i]$ на осі x, на кінцях яких $f(a_i)f(b_i) < 0$ і на яких похідна $f'(x)$ зберігає незмінний знак. Цей спосіб потребує порівняно великої кількості визначень знака $f(x)$ у різних точках осі x.

У цьому сенсі графічний спосіб відокремлення коренів рівняння є більш наочним. Корені рівняння (3.1) визначають, як точки перетину графіка $f(x)$ з віссю абсцис.

Для складних функцій $f(x)$ рівняння $f(x) = 0$ доцільно подати у вигляді $f(x) = f_1(x) - f_2(x) = 0$, де $f_1(x)$ і $f_2(x)$ – функції простіші, ніж $f(x)$. Корені рівняння (3.1) у цьому випадку будуть абсцисами точок перетину функцій $f_1(x)$ і $f_2(x)$.

На практиці можна поєднувати обидва способи, тобто спочатку графічним способом грубо визначити розташування коренів, а потім аналітично перевірити виконання вимоги єдиності кореня на кожному із інтервалів.

Приклад 3.1. Відокремити дійсні корені рівняння $x^2 - \cos x = 0$ на відрізках, довжина яких не перевищує 0.5.

Розв'язок. Побудуємо графік функції $f(x) = x^2 - \cos x$ (рис. 3.1). перепишемо задане рівняння у вигляді $x^2 = \cos x$ і побудуємо графіки функцій лівої і правої частин отриманого рівняння (рис. 3.2).

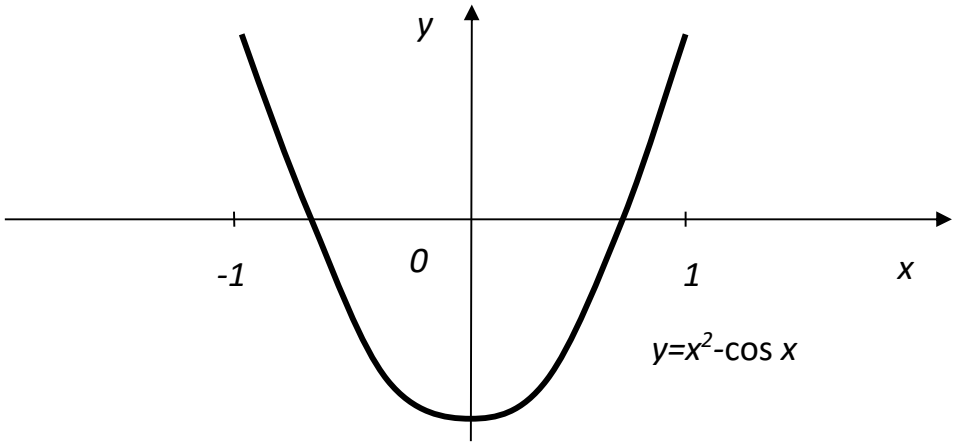


Рис. 3.1. Відокремлення коренів рівняння $x^2 - \cos x = 0$.

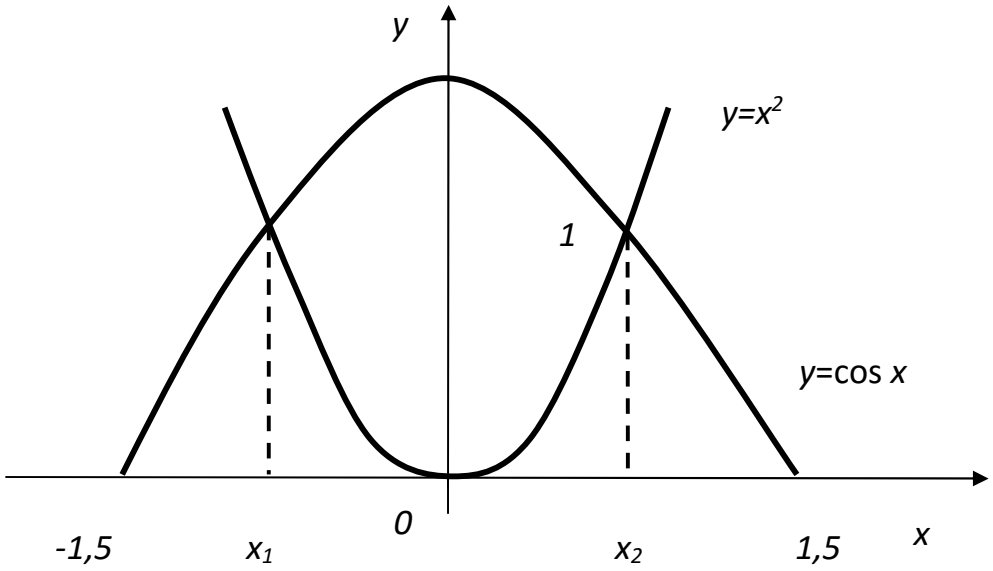


Рис. 3.2. Побудова графіків функцій $x^2 = \cos x$

Скористаємось спочатку графічним методом відокремлення коренів рівняння. З рис. 3.1 видно, що корінь рівняння x_1 знаходиться на інтервалі $[-1;-0.5]$, а корінь x_2 – на інтервалі $[0.5;1]$ (грубе визначення розміщення коренів). Те саме підтверджує рис. 3.2.

Далі за допомогою аналітичного способу відокремлення коренів рівняння, встановимо, що

1) $f(-1)>0$, $f(-0.5)<0$ (тобто $f(-1)\cdot f(-0.5)<0$, $f'(x)<0$ на відрізку $[-1;-0.5]$], і, отже, $x_1\in[-1;-0.5]$;

2) $f(0.5)<0$, $f(1)>0$ (тобто $f(0.5)\cdot f(1)<0$, $f'(x)>0$ на відрізку $[0.5;1]$, і, отже, $x_2\in[0.5;1]$. Тому перший корінь рівняння знаходиться на відрізку $[-1;-0.5]$, а другий – на відрізку $[0.5;1]$.

3.2. Уточнення коренів (метод поділу відрізка навпіл)

На цьому етапі уточнюють дійсні корені рівняння (3.1), які містяться в $[a_i, b_i]$, з похибкою, не більшою за ε.

Метод поділу відрізка навпіл (метод бісекції) полягає в тому, що відрізок $[a,b]$, який містить корінь, ділять навпіл, після чого розглядають ту його половину, на кінцях якої $f(x)$ набуває різних знаків (рис. 3.3).

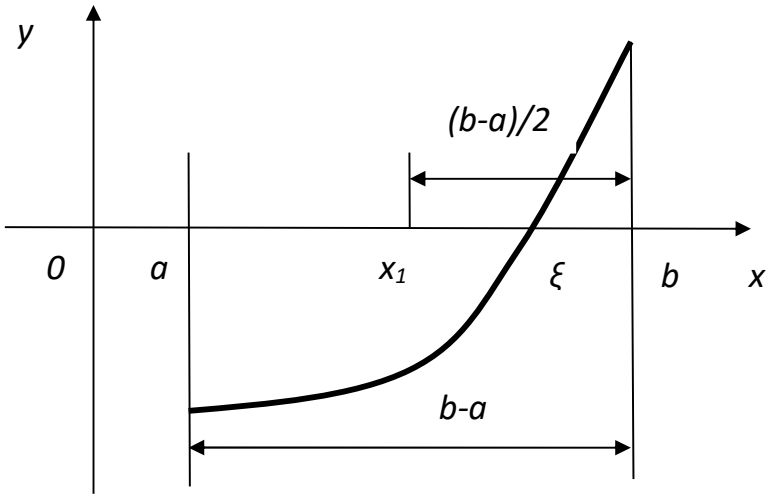


Рис. 3.3. Геометрична інтерпретація методу поділу відрізка навпіл

Новий відрізок знову ділять навпіл і т.д. Для забезпечення наближення до кореня з похибкою, не більшою за ε, процес ділення навпіл продовжують доти, поки не буде виконуватись нерівність $|b_k-a_k|\le 2\varepsilon$, де $[a_k,b_k]$ – відрізок після k-го поділу, такий що $f(a_k)\cdot f(b_k)<0$. Якщо вважати, що $x_k=(a_k+b_k)/2$, то в такому разі $|x_k-\xi|<\varepsilon$, а значення x_k і є шуканим коренем рівняння.

Псевдокод алгоритму уточнення коренів методом поділу відрізка навпіл

Вхідні дані:
[a,b] – інтервал відділення кореня;
eps – точність визначення кореня;

f(x) – ліва частина рівняння.
Вихід: x - значення кореня з заданою точністю.

```
while( (b – a)  >= 2*eps){
    c = (b + a)/2;
    if( f(c)*f(a) < 0)
      b = c;
    else
      a = c;
}
x = (b + a)/2;
```

3.3. Метод послідовних наближень (метод ітерацій)

Метод полягає в тому, що рівняння (3.1) замінюють еквівалентним рівнянням

$$x = \varphi(x), \tag{3.2}$$

для якого в будь-який спосіб обирають початкове наближення x_0 (рис.3.4)з проміжку $[a,b]$ та підставляють у праву частину (3.2). В результаті маємо

$$x_1 = \varphi(x_0).$$

Отримане наближення x_1 підставляють у праву частину (3.2) для обчислення x_2 і т.д.; k -е наближення кореня може бути отримане з рівняння

$$x_k = \varphi(x_{k-1}). \tag{3.3}$$

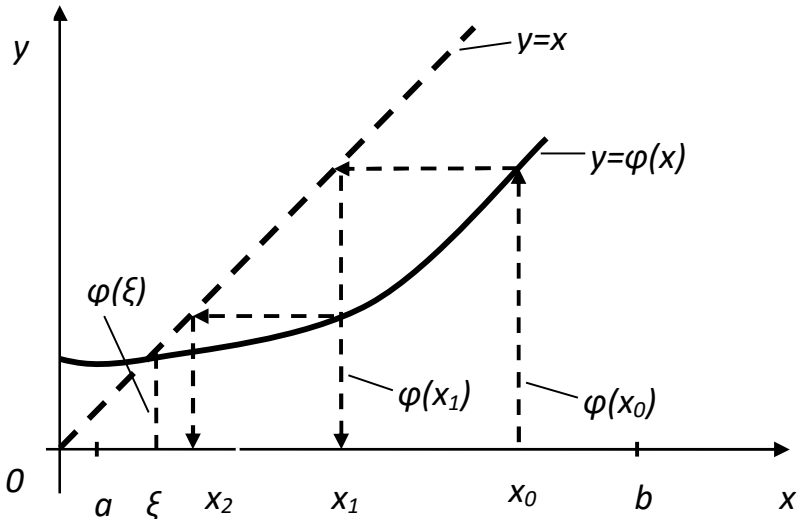


Рис. 3.4. Геометрична інтерпретація методу послідовних наближень (методу ітерацій)

Умову збіжності (3.3) до точного значення кореня визначає наступна теорема.

Теорема. Нехай функція $\varphi(x)$ визначена та диференційована на відрізку $[a,b]$, причому всі її значення належать діапазону $[a,b]$, тобто $\varphi(x) \in [a,b]$. Якщо існує правильний дріб q, такий що $|\varphi'(x)| \le q < 1$ при $a < x < b$, то:

1) процес ітерації $x_k = \varphi(x_{k-1}), k = 1,2,...,n$ збігається незалежно від вибору початкового наближення $x_0 \in [a,b]$;

2) граничне значення $\xi = \lim\limits_{n\to\infty} x_n$ є єдиним коренем рівняння $x = \varphi(x)$ на відрізку $[a,b]$.

Таким чином, зведенням рівняння (3.1) до вигляду (3.2), яке можна виконати в різний спосіб, необхідно забезпечити виконання умови $|\varphi'(x)| < 1$ на інтервалі відокремлення кореня.

Слід зазначити, що метод ітерації збігається тим швидше, чим менше значення має q .

Приклад 3.2. Нехай необхідно звести до вигляду, придатного для застосування методу ітерацій до рівняння $x^5 - x - 0.2 = 0$ з метою уточнення кореня на відрізку $[1;1.3]$.

Якщо записати це рівняння у вигляді $x = x^5 - 0.2$, тобто $\varphi(x) = x^5 - 0.2$, а $\varphi'(x) = 5x^4$, то, очевидно, що умова збіжності виконуватись не буде. Якщо ж це рівняння звести до вигляду $x = \sqrt[5]{x + 0.2}$, тобто $\varphi(x) = \sqrt[5]{x + 0.2}$; $\varphi'(x) = \dfrac{1}{5 \cdot \sqrt[5]{(x+0.2)^4}}$, то на інтервалі $1 \le x \le 1.3$ $|\varphi'(x)| \le 0.18$ і $q = 0.18$. Отже, умова $|\varphi'(x)| < 1$ на інтервалі [1;1.3] відокремлення кореня виконується.

Існує узагальнений спосіб зведення рівняння (3.1) до вигляду (3.2), який полягає в наступному. Якщо $f'(x) > 0$ і $0 < m_1 \le f'(x) \le M_1$ на інтервалі $[a,b]$, то рівняння (3.2) слід подати у вигляді $x = x - \lambda \cdot f(x)$, тобто $\varphi(x) \equiv x - \lambda \cdot f(x)$, де $\lambda > 0$ підбирають таким чином, щоб на інтервалі $[a,b]$ виконувалась нерівність $0 < \varphi'(x) \equiv 1 - \lambda \cdot f'(x) \le q < 1$. Ця нерівність буде виконуватись, якщо $\lambda = 1/M_1$. Відповідно $q = 1 - m_1/M_1$.

У випадку $f'(x) < 0$ рівняння $f(x) = 0$ слід замінити на рівняння $-f(x) = 0$.

Для забезпечення точності $|x_k - \xi| < \varepsilon$ необхідно продовжувати обчислювати наближення за формулою (3.3) , доки не буде виконуватись нерівність

$$|x_k - x_{k-1}| \le \frac{1-q}{q}\varepsilon.$$

Метод ітерацій є самовиправним, оскільки він забезпечує збіжність для будь-якого $x_0 \in [a,b]$ і тому окрема помилка в обчисленнях, яка не вийшла за межі $[a,b]$, може розглядатися як нове початкове наближення, і не вплине на кінцевий результат.

Псевдокод алгоритму уточнення коренів методом ітерацій

Вхідні дані:
[a,b] – інтервал відділення кореня;
eps – точність визначення кореня;
f(x) – ліва частина рівняння.
Вихід: xk - значення кореня з заданою точністю.
1. Привести рівняння f(x) = 0 до вигляду x = phi(x).
2. Визначити значення q як max{phi'(x)} при всіх $x \in [a;b]$.
3. Обрати початкове значення $x0 \in [a;b]$

4. Обчислити xk = phi(x0).

```
5. while( abs(xk – x0)  > (1 – q)/q * eps){
      x0 = xk;
      xk = phi(x0);
   }
```

3.4. Метод пропорційного поділу відрізка (метод хорд)

Метод полягає в тому, що відрізок кривої на інтервалі $[a,b]$ замінюють хордою, яку проводять через точки [*a, f(a)*] та [*b, f(b)*]. Припустимо для визначеності, що $f(a) < 0$ і $f(b) > 0$. Замість того, щоб ділити відрізок навпіл, більш природно поділити його у відношенні $-f(a)/f(b)$ (рис. 3.5). Це дасть наближене значення кореня

$$x_1 = a + h_1, \tag{3.4}$$

де

$$h_1 = \frac{-f(a)}{-f(a)+f(b)} \cdot (b-a) = -\frac{f(a)}{f(b)-f(a)} \cdot (b-a) \tag{3.5}$$

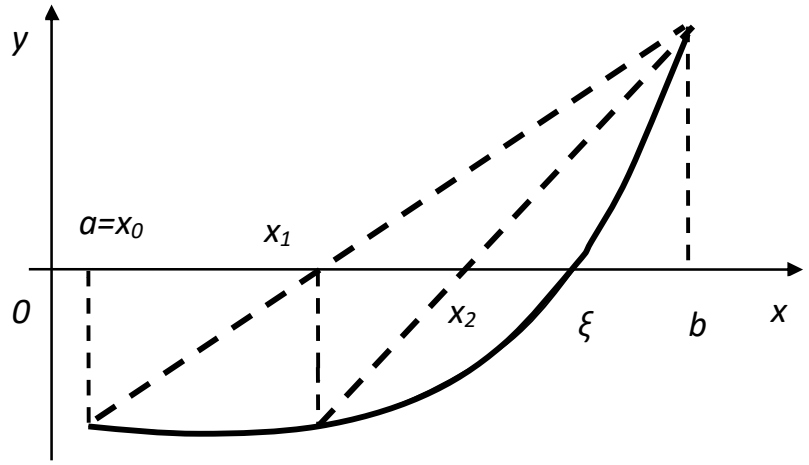


Рис. 3.5. Геометрична інтерпретація методу хорд

Далі, застосовуючи цей прийом до того з відрізків [*a*,*x1*] або [*x1*,*b*], на кінцях якого $f(x)$ має протилежні знаки, дістанемо друге наближення кореня x_2 до точки ζ, яка є точним значенням кореня. Геометрично метод еквівалентний заміні кривої $y = f(x)$ хордою, що проходить через точки $[a, f(a)]$ і $[b, f(b)]$. Справді, рівнянням хорди є

$$\frac{x-a}{b-a}=\frac{y-f(a)}{f(b)-f(a)}.$$

Звідси, вважаючи, що $x=x_1$ і $y=0$, отримаємо

$$x_1=a-\frac{f(a)}{f(b)-f(a)}(b-a).$$

Останнє співвідношення еквівалентне виразам (3.4) і (3.5). Припустимо, що $f''(x)>0$, у противному випадку замінимо рівняння $f(x)=0$ на рівняння $-f(x)=0$. Як видно з рис. 3.4 кінець хорди *b* нерухомий і послідовні наближення x_k, починаючи з $x_0=a$, які обчислюють за формулою

$$x_{n+1}=x_n-\frac{f(x_n)}{f(b)-f(x_n)}\cdot(b-x_n),\ n=0,\ 1,\ 2,\ ... \qquad (3.6)$$

утворять монотонну спадну послідовність

$$a<\xi<...<x_{n+1}<x_n...<x_1<x_0.$$

У випадку, якщо $f(a)>0$ і $f(b)<0$, то нерухомим є кінець хорди *a*, а послідовні наближення x_k, починаючи з $x_0=b$, які обчислюють за формулою

$$x_{n+1}=x_n-\frac{f(x_n)}{f(x_n)-f(a)}\cdot(x_n-a),\quad n=0,\ 1,\ 2,\ ...,. \qquad (3.7)$$

утворять монотонну зростаючу послідовність, причому

$$x_0<x_1<x_2<...<x_n<x_{n+1}<...<\xi<b.$$

Отже, можна зробити такі висновки:

1) нерухомим є той кінець хорди, для якого знак $f(x)$ збігається зі знаком $f''(x)$, тобто $f''(x)f(x)>0$;

2) послідовні наближення x_n лежать по той бік кореня ξ, де $f(x)$ має знак, протилежний знаку $f''(x)$.

В обох випадках кожне наступне наближення x_{n+1} ближче до ξ, ніж попереднє x_n. Нехай

$$\overline{\zeta}=\lim_{n\to\infty}x_n\quad(a<\overline{\xi}<b)$$

Ця границя існує, тому що послідовність $\{x_n\}$ обмежена і монотонна. Переходячи до границі рівності (3.7), отримаємо

$$\bar{\zeta} = \bar{\xi} - \frac{f(\bar{\xi})}{f(\bar{\xi}) - f(a)}(\xi - a),$$

звідси $f(\bar{\xi}) = 0$. Оскільки згідно з припущенням рівняння $f(x) = 0$ має єдиний корінь в інтервалі $[a,\ b]$, то $\bar{\xi} = \xi$, що й треба було довести. Об'єднуючи формули (3.6) і (3.7), послідовні наближення до точного значення кореня можна обчислювати за формулою

$$x_{n+1} = x_n - \frac{f(x_n)}{f(x_n) - f(c)}(x_n - c),\ n = 0,\ 1,\ 2, \ldots\ ,$$

де c – нерухома точка, у якій функція $f(x)$ та її друга похідна $f''(x)$ мають однаковий знак, тобто $f(x)f''(x) > 0$. За початкове (нульове) наближення вибирають кінець інтервалу $[a,\ b]$, протилежний нерухомій точці. Отже, умовою, за якою можна здійснити коректний вибір початкового наближення, є сталість знака другої похідної на інтервалі $[a,\ b]$. Якщо ця умова не виконується, потрібно зменшити інтервал $[a,\ b]$.

Для оцінювання точності отриманого наближення користуються оцінкою

$$|x_n - \xi| \le \frac{|f(x_n)|}{m_1},$$

де $0 < m_1 \le |f'(x)|$ для $x \in [a,\ b]$. Константу m_1 приймають такою, щоб вона дорівнювала $\min|f'(x)|$ на $[a,\ b]$. Якщо при $a \le x \le b$, $m_1 = 0$, то інтервал $[a,\ b]$ необхідно зменшити. Процес обчислення чергового наближення завершується, як тільки $\frac{|f(x_n)|}{m_1} \le \varepsilon$. Така оцінка точності отриманого наближення випливає з теореми Лагранжа про середнє.

Нехай ξ - точний, а $\bar{x}$ - наближений корені рівняння $f(x) = 0$, розташовані на одному відрізку $[a,\ b]$, причому $0 < m_1 \le |f'(x)|$ при $a \le x \le b$ (зокрема як m_1 можна взяти найменше значення $f'(x)$ на інтервалі $[a,\ b]$). У такому випадку справедлива оцінка

$$|\bar{x} - \xi| \le \frac{|f(\bar{x})|}{m_1}.$$

Для доведення застосуємо теорему Лагранжа, з якої маємо $f(\bar{x})-f(\xi)=(\bar{x}-\xi)f'(c)$, де с – точка між значеннями $\bar{x}$ і ξ, тобто $c\in(a,b)$. Звідси, оскільки $f(\xi)=0$ і $f'(c)\le m_1$, отримуємо

$$\left|f(\bar{x})-f(\xi)\right|=\left|f(\bar{x})\right|\ge m_1\left|\bar{x}-\xi\right|,$$

отже

$$\left|\bar{x}-\xi\right|\le\frac{f(\bar{x})}{m_1}.$$

Використовуючи цю формулу, остаточно маємо

$$\left|x_n-\xi\right|\le\frac{\left|f(x_n)\right|}{m_1},$$

де $|f'(x)|\le m_1$ при $a\le x\le b$.

Псевдокод алгоритму наближеного розв'язання рівнянь за методом хорд:

Вхідні дані:
[a,b] – інтервал відділення кореня;
eps – точність визначення кореня;
f(x) – ліва частина рівняння.
f'(x) – перша похідна лівої частини рівняння.
f"(x) – друга похідна лівої частини рівняння.
Вихід: xk - значення кореня з заданою точністю.

1. *Обчислити константу m1 як min{|f'(x)|} при всіх* $x\in[a;b]$.

2. *Визначити початкове значення xk як кінець інтервалу [a;b], протилежний тому для якого справджується нерівність f"(x)f(x) > 0. Позначимо нерухомий кінець інтервалу точкою c.*

```
3.while( abs(f(xk))/m1)  >  eps){
      xk = xk-f(xk)/(f(xk)-f(c))*(c - xk);
   }
```

3.5. Метод дотичних (метод Ньютона)

Метод полягає в тому, що відрізок кривої $f(x)$ на інтервалі $[a,\ b]$ замінюють дотичною (рис.3.6), проведеною у точці $[a,f(a)]$ або у точці $[b,f(b)]$– залежно від властивостей $f(x)$, а саме: за початкове наближення x_0 вибирають той кінець відрізка $[a,\ b]$, якому відповідає ордината з таким самим знаком, що й знак $f''(x)$, тобто $f''(x)f(x)>0$.

Послідовні наближення кореня рівняння обчислюють за формулою

$$x_{n+1}=x_n-\frac{f(x_n)}{f'(x_n)},\qquad n=0,\ 1,\ 2,\ldots\ .$$

Точність отриманого наближення оцінюють за формулою

$$|x_n - \xi| \le \frac{|f(x_n)|}{m_1}.$$

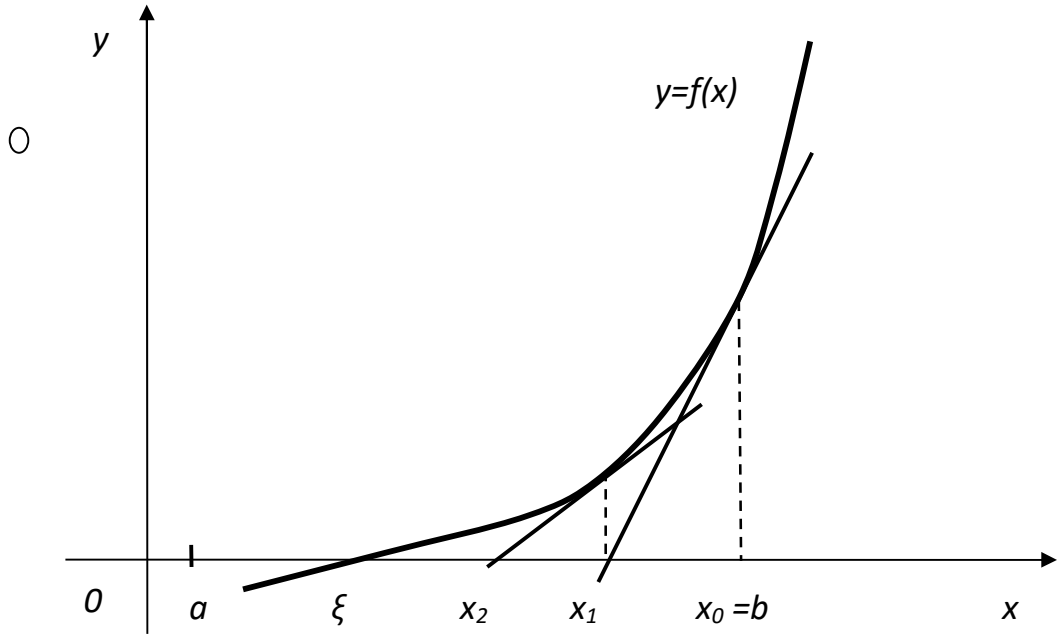


Рис. 3.6. Геометрична інтерпретація методу дотичних

Псевдокод алгоритму наближеного розв'язання рівнянь за методом дотичних

Вхідні дані:
[a,b] – інтервал відділення кореня;
eps – точність визначення кореня;
f(x) – ліва частина рівняння.
f'(x) – перша похідна лівої частини рівняння.
f"(x) – друга похідна лівої частини рівняння.
Вихід: xk - значення кореня з заданою точністю.

1. *Обчислити константу m1 як min{|f'(x)|} при всіх* $x \in [a;b]$.
2. *Визначити початкове значення xk як кінець інтервалу [a;b], для якого виконується нерівність f"(x)f(x) > 0.*

```
3.while( abs(f(xk))/m1)  >  eps){
      xk = xk – f(xk)/f'(xk);
   }
```

Контрольні запитання та завдання до розділу 3

1. З яких етапів складається розв'язання рівнянь наближеними методами?
2. Які існують способи для відокремлення коренів рівняння?
3 .Сформулюйте умови існування та єдиності кореня рівняння на інтервалі $[a,b]$.
4. У чому полягає суть методу поділу відрізку навпіл?
5. Обчислити перші 2 наближення коренів рівняння $2x^2+5x-1=0$, які належать проміжку $[-3;-2]$, методом половинного поділу.
6. У чому полягає суть методу дотичних?
7. Як визначити нульове наближення кореня за методом дотичних?
8. Надайте геометричну інтерпретацію методу дотичних.
9. Обчислити перші 2 наближення корені рівняння $x^3-3x^2+x=0$, що належать проміжку [3;3.5], методом дотичних.
10. У чому полягає суть методу хорд?
11. Як визначається нульове наближення кореня за методом хорд?
12. Обчислити перші 2 наближення кореня рівняння $x^2 - 2x - 1= 0$, що належить проміжку [2;2.7], за методом хорд.
13. Сформулюйте достатні умови застосування методу хорд і методу дотичних.
14. У чому полягає суть методу простих ітерацій?
15. Сформулюйте умови збіжності методу ітерацій.
16. Обчислити перші 2 наближення кореня рівняння $3x^2-x-3=0$, що належить проміжку [1;2], методом простої ітерації.
17. У чому полягає універсальний спосіб зведення рівняння до вигляду, що відповідає умові збіжності методу ітерацій?

РОЗДІЛ 4. МЕТОДИ РОЗВ'ЯЗАННЯ СИСТЕМ ЛІНІЙНИХ АЛГЕБРАЇЧНИХ РІВНЯНЬ

4.1. Системи лінійних рівнянь

Система лінійних алгебраїчних ***рівнянь*** з n невідомими має вигляд

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

де x_1, x_2,..., x_n - невідомі; a_{11}, a_{12},..., a_{nn} - коефіцієнти; $b_1, b_2, \ldots, b_n$ - вільні члени. Множину чисел $\{t_1, t_2, \ldots, t_n\}$, яка перетворює кожне з рівнянь системи на тотожність, називають розв'язком системи.

Систему лінійних рівнянь називають ***сумісною***, якщо вона має хоча б один розв'язок, і ***несумісною***, якщо розв'язку немає. Сумісну систему лінійних рівнянь, яка має один розв'язок, називають ***визначеною.*** Якщо ж сумісна система має декілька розв'язків, то її називають ***невизначеною***. Дві системи лінійних рівнянь з однією і тією самою кількістю невідомих називають ***еквівалентними***, якщо вони обидві несумісні або обидві сумісні і мають один і той самий розв'язок.

Методи розв'язання систем лінійних алгебраїчних рівнянь діляться на три групи. ***Точні або прямі методи*** характеризуються тим, що значення невідомих отримують за фіксовану кількість алгебраїчних операцій. Алгоритми цієї групи складаються з двох етапів: на першому систему перетворюють до того чи іншого спрощеного вигляду, на другому – розв'язують спрощену систему й отримують значення невідомих. За ***методами послідовних наближень*** розв'язують задачу виходячи з певної точності, яка досягається послідовним уточненням початкового розв'язку. До третьої групи належать стохастичні методи, які базуються на ідеї ***методу Монте-Карло***.

4.2. Метод єдиного поділу

Метод (схема) єдиного поділу належить до прямих методів, є одним із варіантів виключення Гауса та ґрунтується на ідеї еквівалентного перетворення вихідної системи до трикутного вигляду (прямий хід) і подальшого її розв'язання (зворотний хід).

Розглянемо систему лінійних алгебраїчних рівнянь 3-го порядку

$$a_{11}^{(0)}x_1 + a_{12}^{(0)}x_2 + a_{13}^{(0)}x_3 = a_{14}^{(0)},$$

$$a_{21}^{(0)}x_1 + a_{22}^{(0)}x_2 + a_{23}^{(0)}x_3 = a_{24}^{(0)}, \qquad (4.1)$$

$$a_{31}^{(0)}x_1 + a_{32}^{(0)}x_2 + a_{33}^{(0)}x_3 = a_{34}^{(0)}.$$

Верхній індекс вказує номер кроку перетворення коефіцієнтів. Припустимо, що $a_{11}^{(0)} \neq 0$ (головний елемент першого рядка). Поділимо перше рівняння системи (4.1) на $a_{11}^{(0)}$, тоді матимемо

$$x_1 + a_{12}^{(1)}x_2 + a_{13}^{(1)}x_3 = a_{14}^{(1)}, \qquad (4.2)$$

де $a_{1j}^{(1)} = a_{1j}^{(0)} / a_{11}^{(0)}$ *(j = 2, 3, 4).* Якщо тепер рівняння (4.2) послідовно множити на $a_{i1}^{(0)}$ (*i = 2,3)* і віднімати його від другого та третього рівнянь, то коефіцієнти при x_1 у двох останніх рівняннях дорівнюватимуть 0, тобто змінну x_1 буде виключено з них. Перетворені рівняння матимуть вигляд

$$a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 = a_{24}^{(1)};$$

$$a_{32}^{(1)}x_2 + a_{33}^{(1)}x_3 = a_{34}^{(1)}, \qquad (4.3)$$

де $a_{ij}^{(1)} = a_{ij}^{(0)} - a_{1j}^{(1)}a_{i1}^{(0)}$, *(i = 2, 3; j = 2, 3, 4).*

Припустимо, що головний елемент другого рядка $a_{22}^{(1)}$ також не є нулем. Тоді, виконуючи ділення на нього першого рівняння системи (4.3), отримаємо

$$x_2 + a_{23}^{(2)}x_3 = a_{24}^{(2)}, \qquad (4.4)$$

де $a_{2j}^{(2)} = a_{2j}^{(1)} / a_{22}^{(1)}$ *(j = 3, 4).* Виключимо тепер невідоме x_2 шляхом множення (4.4) на $a_{32}^{(1)}$ і наступного віднімання його від другого рівняння системи (4.3). Внаслідок цього дістанемо

$$a_{33}^{(2)}x_3 = a_{34}^{(2)}, \qquad (4.5)$$

де $a_{3j}^{(2)} = a_{3j}^{(1)} - a_{2j}^{(2)}a_{32}^{(1)}$ *(j = 3, 4).*

Нарешті, якщо $a_{33}^{(2)} \neq 0$, то діленням рівняння (4.5) на головний елемент третього рядка, отримаємо

$$x_3 = a_{34}^{(3)}, \tag{4.6}$$

де $a_{34}^{(3)} = a_{34}^{(2)} / a_{33}^{(2)}$.

Об'єднаємо тепер рівняння (4.2), (4.4) та (4.6) в одну систему з верхньою трикутною матрицею, що еквівалентна вихідній системі

$$\begin{aligned} x_1 + a_{12}^{(1)} x_2 + a_{13}^{(1)} x_3 &= a_{14}^{(1)}; \\ x_2 + a_{23}^{(2)} x_3 &= a_{24}^{(2)}; \\ x_3 &= a_{34}^{(3)}. \end{aligned} \tag{4.7}$$

Із системи (4.7) невідомі x_1, x_2, x_3 можна отримати у зворотному порядку

$$\begin{aligned} x_3 &= a_{34}^{(3)}; \\ x_2 &= a_{24}^{(2)} - a_{23}^{(2)} x_3; \\ x_1 &= a_{14}^{(1)} - a_{12}^{(1)} x_2 - a_{13}^{(1)} x_3. \end{aligned} \tag{4.8}$$

Процес зведення системи (4.1) до трикутного вигляду називають прямим ходом, а визначення невідомих за формулами (4.8) - зворотним ходом.

Зауваження. Головні елементи, які отримують виконуючи прямий хід, дають можливість обчислити головний визначник матриці коефіцієнтів вихідної системи:

$$\det A = \prod_{i=1}^{n} a_{ii}^{(i-1)}.$$

Очевидно, що виконання прямого ходу потребує такої кількості алгебраїчних множень

$$n(n+1) + (n-1)n + \ldots + 1 \cdot 2 = (1^2 + 2^2 + \ldots + n^2) + (1 + 2 + \ldots + n) = \frac{n(n+1)(n+2)}{3}$$

Зворотний хід потребує $\frac{n(n-1)}{2}$ алгебраїчних множень, а їх загальна кількість дорівнює

$$\frac{2n(n+1)(n+2)}{3} + n(n-1) < n^3.$$

Отже, обчислювальна складність методу має порядок $O(n^3)$.

Недоліком розглянутого методу є те, що в ході реалізації прямого ходу один з головних елементів може виявитися рівним нулю, а це робить неможливим отримання розв'язку системи, тоді як вона може бути сумісною і мати єдиний розв'язок. Крім того, аналіз похибок показує, що в разі виконання прямого ходу похибка тим менша, чим більші $a_{kk}^{(k-1)}$. Ці обставини враховуються при реалізації схеми з вибором головного елемента.

Приклад 4.1. Розв'язати методом єдиного поділу систему

$$\begin{cases} 4x_1 + x_2 - 2x_3 = 8, \\ x_1 - 5x_2 + x_3 = -10, \\ 3x_1 + x_2 - 5x_3 = 10. \end{cases}$$

Поділимо перше рівняння на $a_{11}^{(0)} = 4$

$$\begin{cases} x_1 - 0.25x_2 - 0.5x_3 = 2, \\ x_1 - 5x_2 + x_3 = -10, \\ 3x_1 + x_2 - 5x_3 = 10. \end{cases} \tag{4.9}$$

Далі отримуємо перший головний рядок

$$x_1 + 0.25x_2 - 0.5x_3 = 2. \tag{4.10}$$

Скористаємося ним для виключення невідомого x_1 з другого й третього рівнянь (4.9)

$$\begin{cases} x_2 - 0.2857x_3 = 2.2857, \\ -0.25x_2 + 3.5x_3 = -4. \end{cases} \tag{4.11}$$

Як результат ділення першого рівняння (4.11) на $a_{22}^{(1)} = 1$ отримаємо другий головний рядок

$$x_2 - 0.2857x_3 = 2.2857. \tag{4.12}$$

Скористаємося цим рядком для виключення невідомого x_2 з рівнянь (4.11), Тоді маємо

$$\{x_3 = -1. \tag{4.13}$$

В результаті ділення першого рівняння (4.13) на $a_{33}^{(2)} = 1$ отримаємо третій головний рядок

$$x_3 = -1. \tag{4.14}$$

На цьому прямий хід завершено. З рівнянь (4.10), (4.12) і (4.14) утворимо систему з трикутною матрицею

$$\begin{cases} x_1 + 0.25x_2 - 0.5x_3 = 2, \\ x_2 - 0.2857x_3 = 2.2857, \\ x_3 = -1. \end{cases} \tag{4.15}$$

Скориставшись рівняннями (4.15), виконаємо зворотний хід і маємо розв'язок системи: $x_1 = 1$, $x_2 = 2$, $x_3 = -1$.

Псевдокод алгоритму розв'язання систем лінійних алгебраїчних рівнянь методом єдиного поділу

Вхідні дані:
n – порядок системи
A – матриця коефіцієнтів системи;
B – стовпчик вільних членів системи;
Вихід: X – вектор розв'язків системи.
// Прямий хід
1. k = 1.
2. Якщо $a_{kk} = 0$ *- вихід*
3. Поділити k–й рядок системи на a_{kk} *.*
4. Запам'ятати k–е рівняння (головний рядок) системи з трикутною матрицею.
5. for(i = k + 1; i < n; i = i + 1){
a. Помножити головний рядок на $a_{ik}, i \neq k$ *.*
b. Відняти від нього рядок i *.*
}

6. $k = k + 1$.

7. Якщо k <= n, перейти до кроку 2.
// Зворотний хід
8. k = n.
9. Обчислити $x_k = b_k - \sum_{j=k+1}^{n} a_{kj} x_j$ *.*
10. k = k - 1.
11. Якщо k > 0, перейти до кроку 8.

Зауваження. Щоб крок 5.*b* виконувався коректно, треба мати копію головного рядка і відновлювати його кожен раз перед виконанням кроку 5.*a*.

4.3. Виключення Гауса-Жордана

Цей метод також належить до прямих. Ідея методу виключення Гауса-Жордана полягає в тому, щоб за допомогою послідовного виключення невідомих звести матрицю коефіцієнтів вихідної системи до одиничної матриці. При цьому не потрібен зворотній хід..

Розглянемо систему лінійних алгебраїчних рівнянь 3-го порядку (верхній індекс при кожному з коефіцієнтів вказує номер кроку їх перетворення).

$$\begin{aligned} a_{11}^{(0)} x_1 + a_{12}^{(0)} x_2 + a_{13}^{(0)} x_3 &= a_{14}^{(0)}, \\ a_{21}^{(0)} x_1 + a_{22}^{(0)} x_2 + a_{23}^{(0)} x_3 &= a_{24}^{(0)}, \\ a_{31}^{(0)} x_1 + a_{32}^{(0)} x_2 + a_{33}^{(0)} x_3 &= a_{34}^{(0)}. \end{aligned} \qquad (4.16)$$

Припустимо, що $a_{11}^{(0)} \neq 0$ (головний елемент першого рядка). Поділимо перший рядок (4.14) на $a_{11}^{(0)}$, в результаті матимемо

$$\begin{aligned} 1x_1 + a_{12}^{(1)} x_2 + a_{13}^{(1)} x_3 &= a_{14}^{(1)}, \\ a_{21}^{(0)} x_1 + a_{22}^{(0)} x_2 + a_{23}^{(0)} x_3 &= a_{24}^{(0)}, \\ a_{31}^{(0)} x_1 + a_{32}^{(0)} x_2 + a_{33}^{(0)} x_3 &= a_{34}^{(0)}. \end{aligned} \tag{4.17}$$

де $a_{1j}^{(1)} = a_{1j}^{(0)} / a_{11}^{(0)}$ ($j = 2, 3, 4$). Якщо тепер перше рівняння системи (4.17) послідовно множити на $a_{i1}^{(1)} (i = 2, 3)$ і віднімати його від другого та третього рівнянь, коефіцієнти при x_1 у двох останніх рівняннях системи (4.17) дорівнюватимуть 0, тобто змінну x_1 буде виключено з них. Перетворені рівняння матимуть вигляд

$$\begin{aligned} 1x_1 + a_{12}^{(1)} x_2 + a_{13}^{(1)} x_3 &= a_{14}^{(1)}, \\ 0x_1 + a_{22}^{(1)} x_2 + a_{23}^{(1)} x_3 &= a_{24}^{(1)}, \\ 0x_1 + a_{32}^{(1)} x_2 + a_{33}^{(1)} x_3 &= a_{34}^{(1)}. \end{aligned} \tag{4.18}$$

де $a_{ij}^{(1)} = a_{ij}^{(0)} - a_{1j}^{(0)} a_{i1}^{(1)}$ ($i = 2, 3;\ j = 2, 3, 4$).

Припустимо, що головний елемент другого рядка також не нуль, тобто $a_{22}^{(1)} \neq 0$. Тоді, поділимо на нього другий рядок системи (4.18) і отримаємо

$$\begin{aligned} 1x_1 + a_{12}^{(1)} x_2 + a_{13}^{(1)} x_3 &= a_{14}^{(1)}, \\ 0x_1 + 1x_2 + a_{23}^{(2)} x_3 &= a_{24}^{(2)}, \\ 0x_1 + a_{32}^{(1)} x_2 + a_{33}^{(1)} x_3 &= a_{34}^{(1)}. \end{aligned} \tag{4.19}$$

де $a_{23}^{(2)} = a_{23}^{(1)} / a_{22}^{(1)}$, $a_{24}^{(2)} = a_{24}^{(1)} / a_{22}^{(1)}$. Якщо тепер друге рівняння системи (4.19) послідовно множити на $a_{12}^{(1)}$ та на $a_{32}^{(1)}$ і віднімати його від першого та третього рівнянь, то в результаті коефіцієнти при x_2 у першому й третьому рівняннях дорівнюватимуть 0, тобто змінну x_2 буде виключено з них. Перетворені рівняння матимуть вигляд

$$\begin{aligned} 1x_1 + 0x_2 + a_{13}^{(2)} x_3 &= a_{14}^{(2)}, \\ 0x_1 + 1x_2 + a_{23}^{(2)} x_3 &= a_{24}^{(2)}, \\ 0x_1 + 0x_2 + a_{33}^{(2)} x_3 &= a_{34}^{(2)}, \end{aligned} \tag{4.20}$$

де $a_{1j}^{(2)} = a_{1j}^{(1)} - a_{2j}^{(2)} a_{12}^{(1)}$, $a_{3j}^{(2)} = a_{3j}^{(1)} - a_{2j}^{(2)} a_{32}^{(2)}$, $j = 3,4$.

Нарешті, якщо $a_{33}^{(2)} \neq 0$, то після ділення на нього третього рядка системи (4.20), отримаємо

$$
\begin{aligned}
1x_1 + 0x_2 + a_{13}^{(2)} x_3 &= a_{14}^{(2)},\\
0x_1 + 1x_2 + a_{23}^{(2)} x_3 &= a_{24}^{(2)},\\
0x_1 + 0x_2 + 1x_3 &= a_{34}^{(3)},
\end{aligned}
\tag{4.21}
$$

де $a_{34}^{(3)} = a_{34}^{(2)} / a_{33}^{(2)}$(j = 3,4).

Якщо тепер третє рівняння системи (4.21) послідовно множити на $a_{i3}^{(2)} (i = 1,\ 2)$ і віднімати його від першого та другого рівнянь, то коефіцієнти при x_3 у першому й другому рівняннях дорівнюватимуть 0, отже змінну x_3 буде виключено з них. Перетворені рівняння матимуть вигляд

$$
\begin{aligned}
1x_1 + 0x_2 + 0x_3 &= a_{14}^{(3)},\\
0x_1 + 1x_2 + 0x_3 &= a_{24}^{(3)},\\
0x_1 + 0x_2 + 1x_3 &= a_{34}^{(3)},
\end{aligned}
\tag{4.22}
$$

де $a_{14}^{(3)} = a_{14}^{(2)} - a_{34}^{(3)} a_{13}^{(2)}$, $a_{24}^{(3)} = a_{24}^{(2)} - a_{34}^{(3)} a_{23}^{(2)}$.

Як наслідок, розв'язок вихідної системи (4.16) отримуємо безпосередньо з (4.22), а саме:

$$x_1 = a_{14}^{(3)},\ x_2 = a_{24}^{(3)},\ x_3 = a_{34}^{(3)}.$$

Приклад 4.2. Розв’язати методом виключення Гауса-Жордана систему

$$
\begin{cases}
4x_1 + x_2 - 2x_3 = 8,\\
x_1 - 5x_2 + x_3 = -10,\\
3x_1 + x_2 - 5x_3 = 10.
\end{cases}
$$

Поділимо перше рівняння на $a_{11}^{(0)} = 4$, тоді отримаємо

$$
\begin{cases}
x_1 + 0.25x_2 - 0.5x_3 = 2,\\
x_1 - 5x_2 + x_3 = -10,\\
3x_1 + x_2 - 5x_3 = 10.
\end{cases}
$$

Помножимо перший рядок спочатку на $a_{21}^{(0)} = 1$ і далі віднімемо його від другого рядка, а потім помножимо його на $a_{31}^{(0)} = 3$ і віднімемо від третього рядка. В результаті маємо

$$
\begin{cases}
x_1 + 0.25x_2 - 0.5x_3 = 2,\\
0x_1 - 5.25x_2 + 1.5x_3 = -12,\\
0x_1 + 0.25x_2 - 3.5x_3 = 4.
\end{cases}
$$

Поділимо друге рівняння на $a_{22}^{(1)} = -5.25$

$$\begin{cases} x_1 + 0.25x_2 - 0.5x_3 = 2, \\ 0x_1 + x_2 - 0.2857x_3 = 2.2857, \\ 0x_1 + 0.25x_2 - 3.5x_3 = 4. \end{cases}$$

Помножимо другий рядок спочатку на $a_{12}^{(1)} = 0.25$ і віднімемо його від першого рядка, а потім помножимо його на $a_{32}^{(1)} = 0.25$ і віднімемо від третього рядка. В результаті маємо

$$\begin{cases} x_1 + 0.x_2 - 0.42857x_3 = 1.42857, \\ 0x_1 + x_2 - 0.2857x_3 = 2.2857, \\ 0x_1 + 0x_2 - 3.42857x_3 = 3.42857. \end{cases}$$

Поділимо третє рівняння на $a_{33}^{(12} = -3.42857$

$$\begin{cases} x_1 + 0x_2 - 0.42857x_3 = 1.42857, \\ 0x_1 + x_2 - 0.2857x_3 = 2.2857, \\ 0x_1 + 0x_2 + x_3 = -1. \end{cases}$$

Помножимо третій рядок спочатку на $a_{13}^{(2)} = -0.42857$ і віднімемо його від першого рядка, а потім помножимо його на $a_{23}^{(2)} = -0.2857$ і віднімемо від другого рядка. В результаті маємо

$$\begin{cases} x_1 + 0x_2 + 0x_3 = 1, \\ 0x_1 + x_2 + 0x_3 = 2, \\ 0x_1 + 0x_2 + x_3 = -1. \end{cases}$$

Розв'язок системи зчитуємо зі стовпчика вільних членів: $x_1 = 1$, $x_2 = 2$, $x_3 = -1$.

Псевдокод алгоритму розв'язку системи лінійних алгебраїчних рівнянь методом виключення Гауса-Жордана

Вхідні дані:
n – порядок системи
A – матриця коефіцієнтів системи;
B – стовпчик вільних членів системи;
Вихід: B – вектор розв'язків системи.
1. k = 1.
2. Якщо $a_{kk} = 0$ *- вихід*
3. Поділити k–й рядок системи на a_{kk}.
4. for(i = 1; i < n; i = i + 1){
    a. if(i == k)
        continue;
    b. Помножити рядок і на a_{kk}.
    c. Відняти рядок і від рядка k.
}

5. *k = k + 1.*

6. Якщо k <= n, перейти до кроку 2.

4.4. Метод з вибором головного елемента

Для методу єдиного поділу та методу виключення Гаус-Жордана у випадку, коли черговий головний елемент дорівнює нулю, подальше перетворення матриці стає неможливим. Утім, це ще не означає, що система несумісна й не має єдиного розв'язку. Цей недолік не властивий наступному методу (схемі) з вибором головного елемента. Крім того, ця схема менш чутлива до похибок округлення.

Отже, маємо систему

$$a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = a_{1,n+1},$$

$$a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = a_{2,n+1},$$

.....

$$a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = a_{n,n+1}.$$

Розглянемо розширену матрицю M з коефіцієнтів системи й вільних членів

$$M = \begin{vmatrix} a_{11}a_{12}...a_{1q}...a_{1n}a_{1,n+1} \\ a_{21}a_{21}...a_{2q}...a_{2n}a_{2,n+1} \\ ..... \\ a_{p1}a_{p2}...a_{pq}...a_{pn}a_{p,n+1} \\ ..... \\ a_{n1}a_{n2}...a_{nq}...a_{nn}a_{n,n+1} \end{vmatrix}.$$

Виберемо ненульовий і найбільший за абсолютною величиною коефіцієнт a_{pq}, що не належить до стовпчика вільних членів $(q \neq n+1)$, і обчислимо множники

$$m_i = -\frac{a_{iq}}{a_{pj}},\ i = 1,2,...,n,\ i \neq p.$$

Рядок з номером p матриці M називають ***головним рядком***. Далі до кожного неголовного рядка з номером i додамо головний, помножений на відповідний множник m_i. Як результат, отримаємо матрицю, в якій q -й стовпчик складається з нулів (за винятком a_{pq}). Відкидаючи цей стовпчик і головний рядок, одержимо матрицю $M^{(1)}$ з меншою на одиницю кількістю рядків і стовпчиків. Із матриці $M^{(1)}$ в такий самий спосіб отримаємо $M^{(2)}$ і т.д.:

$M, M^{(1)}, M^{(2)}, ..., M^{(n-1)}$. Остання матриця є двочленною матрицею-рядком, її також вважаємо головним рядком. Для визначення невідомих x_i об'єднуємо в систему всі головні рядки, починаючи з останнього $M^{(n-1)}$.

Далі невідомі обчислюють з трикутної матриці – так само як і за схемою єдиного поділу. Після відповідної зміни нумерації невідомих отримуємо розв'язок вихідної системи. Метод завжди приводить до розв'язку, якщо головний визначник системи не дорівнює нулю

$$\det A = \begin{vmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{vmatrix} \neq 0 .$$

Приклад 4.3. Розв'язати методом з вибором головного елемента систему

$$\begin{cases} 4x_1 + x_2 - 2x_3 = 8, \\ x_1 - 5x_2 + x_3 = -10, \\ 3x_1 + x_2 - 5x_3 = 10. \end{cases}$$

Маємо розширену матрицю M

$$\begin{array}{cccc} 4 & 1 & -2 & 8 \\ 1 & -5 & 1 & -10 \\ 3 & 1 & -5 & 10 \end{array} .$$

Виберемо ненульовий і найбільший за абсолютною вличиною коефіцієнт, що не належить до стовпчика вільних членів, $a_{22}^{(0)} = -5$. Отже $p = q = 2$. Обчислимо множники

$$m_i = -\frac{a_{iq}}{a_{pj}}, \; i = 1, 2, ..., n, \; i \neq p .$$

В результаті маємо $m_1 = 0.2$, $m_3 = 0.2$. При цьому першим головним рядком є

$$x_1 - 5x_2 + x_3 = -10. \qquad (4.23)$$

Тепер до кожного неголовного рядка i додамо головний, помножений на відповідний множник m_i. Отримаємо таку матрицю $M^{(1)}$

$$\begin{array}{ccc} 4.2 & -1.8 & 6 \\ 3.2 & -4.8 & 8 \end{array} .$$

Для цієї матриці головним елементом, вочевидь, є $a_{22}^{(1)} = -4.8$. Отже $p = q = 2$. Обчислимо $m_1 = -0.375$. При цьому, оскільки на попередньому кроці був видалений другий стовпчик (коефіцієнти при x_2), другим головним рядком є

$$3.2x_1 - 4.8x_3 = 8 \quad . \qquad (4.24)$$

Тепер до першого (неголовного) рядка додамо головний, помножений на множник m_1. В результаті отримаємо двочленну матрицю-рядок $M^{(2)}$

3 0 -3

Ця матриця є останнім головним рядком

$$3x_3 = -3. \tag{4.25}$$

Об'єднаємо рівняння (4.23) - (4.25) в систему з трикутною матрицею

$$\begin{cases} x_1 - 5x_2 - x_3 = -10, \\ 3.2x_1 - 4.8x_3 = 8, \\ 3x_3 = -3. \end{cases}$$

Виконуючи зворотний хід, отримуємо розв'язок системи: $x_3 = -1$, $x_1 = 1$, $x_2 = 2$.

Псевдокод алгоритму до методу з вибором головного елемента

Вхідні дані:
n – порядок системи
A – матриця коефіцієнтів системи;
B – стовпчик вільних членів системи;
Вихід: X – вектор розв'язків системи.

1. *Побудувати розширену матрицю M, розміром n×(n + 1)*
2. *k = 1.*
3. *Обрати найбільший за абсолютною вличиною коефіцієнт a_{pq}, що не належить до стовпчика вільних членів*
4. *Якщо $a_{pq} = 0$ - вихід*
5. *for(i = 1; i < n; i = i + 1){*
   if(i == p)
   continue;
   $m_i = -\frac{a_{iq}}{a_{pj}}$
   }
6. *for(i = 1; i < n; i = i + 1){*
   a. if(i == p)
   continue;
   b. Помножити рядок p на m_i.
   c. Відняти рядок p від рядка i.
   }
7. *Запам'ятати головний рядок p.*
8. *Видалити з матриці M рядок p і стовпчик q.*
9. *k = k + 1.*
10. *Якщо k < n, перейти до кроку 3.*

11. k = 1, t = n.

12. З головного рядка t обчислити $x_t = (m_{t,t+1} - \sum_{j=t+1}^{n} m_{kj} x_j)/m_{tt}$,

де $m_{i,j}$ *- коефіцієнти матриці* $M^{(t)}$.

13. k = k + 1, t = t – 1.

14. Якщо k < n, перейти до кроку 12.

15. Змінити нумерацію невідомих.

Зауваження. Щоб крок 6.*c* виконувався коректно, треба мати копію рядка *p* і відновлювати його щоразу перед виконанням кроку 6.*b*.

4.5. Метод простої ітерації

Нехай є система *n* лінійних рівнянь $A\vec{X} = \vec{B}$

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + ... + a_{1j}x_j + ... + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2j}x_j + ... + a_{2n}x_n = b_2, \\ ..... \\ a_{i1}x_1 + a_{i2}x_2 + ...a_{ij}x_j + ... + a_{in}x_n = b_i, \\ ..... \\ a_{n1}x_1 + a_{n2}x_2 + ...a_{nj}x_j + ... + a_{nn}x_n = b_n, \end{cases}$$

де $i = \overline{1,n}$; $j = \overline{1,n}$; $i, j > 0$. Вважаючи, що коефіцієнти головної діагоналі системи відмінні від нуля, перетворимо її до вигляду

$$\vec{X} = \vec{\beta} + \alpha\vec{X}, \tag{4.26}$$

де елементи вектора $\vec{\beta}$ і матриці α обчислюють наступним чином

$\beta_i = a_{i,n+1}/a_{ii}$, $\alpha_{ij} = -a_{ij}/a_{ii}$ при $i \neq j$ та $\alpha_{ij} = 0$ при $i = j$, $(i, j = 1,2,...,n)$.

Тобто формі (4.26) відповідає система

$$\begin{cases} x_1 = \beta_1 + \alpha_{12}x_2 + ... + \alpha_{1j}x_j + ... + \alpha_{1n}x_n, \\ x_2 = \beta_2 + \alpha_{21}x_1 + + ... + \alpha_{2j}x_j + ... + \alpha_{2n}x_n, \\ ..... \\ x_n = \beta_n + \alpha_{n1}x_1 + \alpha_{n2}x_2 + ...\alpha_{n,n-1}x_{n-1}. \end{cases} \tag{4.27}$$

Систему (4.27) розв'яжемо методом послідовних наближень, обраючи як нульове наближення довільний вектор $X^{(0)} = (x_1^0, x_2^0, ..., x_n^0)$. Чергове $(k+1)$-е наближення $k = (0,1,2,...)$ будемо обчислювати, використовуючи попереднє наближення $X^{(k)} = (x_1^{(k)}, x_2^{(k)}, ..., x_n^{(k)})$:

$$\begin{cases} x_1^{(k+1)} = \beta_1 + \alpha_{12}x_2^{(k)} + ... + \alpha_{1j}x_j^{(k)} + ... + \alpha_{1n}x_n^{(k)}, \\ x_2^{(k+1)} = \beta_2 + \alpha_{21}x_1^{(k)} + +... + \alpha_{2j}x_j^{(k)} + ... + \alpha_{2n}x_n^{(k)}, \\ ..... \\ x_n^{(k+1)} = \beta_n + \alpha_{n1}x_1^{(k)} + \alpha_{n2}x_2^{(k)} + ...\alpha_{n,n-1}x_{n-1}^{(k)}. \end{cases} \quad (4.28)$$

Якщо послідовність наближень $X^{(0)}$, $X^{(1)}$,…, $X^{(k)}$, $X^{(k+1)}$,... має границю, то ця границя є точним розв'язком системи (4.28).

Достатньою умовою збіжності обчислень за (4.28) є виконання принаймні однієї з умов

$$\| \alpha \|_m = \max \sum_{j=1}^{n} \left| \alpha_{ij} \right| < 1, \ i = 1,2,...,n, \quad (4.29)$$

або

$$\| \alpha \|_l = \max \sum_{i=1}^{n} \left| \alpha_{ij} \right| < 1, \ \ j = 1,2,...,n,$$

або

$$\| \alpha \|_k = \sqrt{\sum_i \sum_j (\alpha_{ij})^2}.$$

Тобто збіжність забезпечується, якщо хоча б одна з канонічних норм матриці α менша одиниці. При цьому процес ітерації збігається до єдиного розв'язку незалежно від вибору початкового наближення.

На практиці зручніше користуватися *m*–нормою. З умови (4.29) випливає, що методу простої ітерації забезпечена збіжність, якщо $|a_{ii}| > \sum_{i \neq j} |a_{ij}|$, $(i = 1, 2,..., n)$, тобто абсолютні величини коефіцієнтів на головній діагоналі в кожному з рівнянь більші за суму абсолютних величин решти коефіцієнтів (без урахування вільних членів). Це дозволяє визначити, чи виконується умова збіжності, зважаючи безпосередньо на вигляд матриці A вихідної системи.

Зрозуміло, що далеко не кожна система відповідає зазначеним вимогам. Але кожна система, головний визначник якої не дорівнює нулю, може бути зведена до вигляду, де ці умови виконуються. Зробити це можна наступним чином. Із заданої системи вилучають рівняння з коефіцієнтами, абсолютні величини яких більші за суму абсолютних величин решти коефіцієнтів рівняння. Кожне виділене рівняння записують у такий рядок нової системи, щоб найбільший за абсолютною величиною коефіцієнт став діагональним.

З решти невикористаних і виділених на першому етапі рівнянь системи утворюють лінійно-незалежні комбінації так, щоб нові рівняння відповідали (4.29), і кожне невикористане на першому етапі рівняння потрапило б принаймні в одну лінійну комбінацію, яка є рівнянням нової системи.

Наприклад, підготуємо до розв'язання методом простої ітерації систему

$$\begin{aligned} &(1) \quad 10x_1 - 7x_2 - 91x_3 = -12, \\ &(2) \quad 41x_1 - 92x_2 + 12x_3 = 60, \\ &(3) \quad -21x_1 - 96x_2 - 95x_3 = 49. \end{aligned} \quad (4.30)$$

У першому рівнянні системи (4.30) третій коефіцієнт за абсолютною величиною більший за суму абсолютних величин решти коефіцієнтів, отже у перетвореній системі це рівняння посяде місце третього рівняння. У дугому рівнянні коефіцієнт при x_2 більший за суму абсолютних величин решти коефіцієнтів, отже у перетвореній системі це рівняння залишиться на своїй позиції. Третє рівняння утворимо як суперпозицію інших рівнянь, причому друге й третє рівняння мають бути в лінійній комбінації

$$(1)+(2)-(3) \qquad 72x_1 - 3x_2 + 16x_3 = -1,$$

$$(2) \qquad 41x_1 - 92x_2 + 12x_3 = 60,$$

$$(1) \qquad 10x_1 - 7x_2 - 91x_3 = -12.$$

Далі перетворенням системи до вигляду (4.27) отримаємо

$$x_1 = -\frac{1}{72} + \frac{3}{72}x_2 - \frac{16}{72}x_3,$$

$$x_2 = -\frac{60}{92} + \frac{41}{92}x_1 + \frac{12}{92}x_3,$$

$$x_3 = \frac{12}{91} + \frac{10}{91}x_1 - \frac{7}{91}x_2.$$

Обираючи як початковий довільний вектор $X^{(0)} = (x_1^0, x_2^0, x_3^0)$ обчислиимо наступні наближення розвя'зку відповідно до (4.28).

Ітераційний процес продовжується доти, доки чергове наближення $X^{(k+1)}$ не досягне заданої точності ε, тобто не буде виконана умова

$$\left\| X^{(k+1)} - X^{(k)} \right\|_m \le \frac{1-q}{q}\varepsilon,$$

де $q = \| \alpha \|_m < 1$, $\|\alpha\|_m$ - m-норма матриці α,

$$\|\alpha\|_m = \max_i \sum_{j=1}^{n} |\alpha_{ij}|, \; i = 1,2,...,n.$$

Кількість ітерацій k, необхідна для досягнення заданої точності ε, може бути визначена із умови

$$\left\| X - X^{(k)} \right\|_m \le \frac{\|\alpha\|_m^{(k+1)}}{1 - \|\alpha\|_m} \|\beta\|_m,$$

де X - точний розв'язок. При цьому вважається, що як початкове наближення обирається вектор $\vec{\beta}$.

Оцінимо обчислювальну складність методу простих ітерацій. Очевидно, що виконання одної ітерації потребує виконання $n(n+1)$ алгебраїчних множень. Якщо для досягнення заданої точності знадобилося k ітерацій, то маємо оцінку $O(kn^2)$. Звідси випливає, що у випадку $k < n$, ітераційний метод має меншу обчислювальну складність. Наприклад, $n = 1000$, а $k = 100$. Тоді схема єдиного поділу має складність $O(10^9)$, а метод простої ітерації - $O(10^8)$.

Приклад 4.4. Знайти методом простої ітерації перше наближення розв'язку системи лінійних рівнянь

$$\begin{cases} 5x_1 - 4x_2 - x_3 = -2, \\ 4x_1 + x_2 - 2x_3 = 8, \\ 3x_1 + x_2 - 5x_3 = 10. \end{cases}$$

Розв'язок. Перетворимо систему до вигляду, придатного для ітерації. Оскільки третій коефіцієнт третього рівняння за абсолютною величиною більший за суму решти коефіцієнтів, залишаємо його на третій позиції перетвореної системи. Далі поміняємо місцями перше та друге рівняння

$$\begin{cases} 4x_1 + x_2 - x_3 = 8, \\ 5x_1 - 4x_2 - x_3 = -2, \\ 3x_1 + x_2 - 5x_3 = 10. \end{cases}$$

Тепер перше й третє рівняння відповідають умовам збіжності. Залишається побудувати друге рівняння, яке відповідатиме умові збіжності. Для цього помножимо перше рівняння на (-1) і додамо його до другого. В результаті отримуємо

$$\begin{cases} 4x_1 + x_2 - x_3 = 8, \\ 1x_1 - 5x_2 + x_3 = -10, \\ 3x_1 + x_2 - 5x_3 = 10. \end{cases}$$

Отже перетворена система має вигляд

$$\begin{cases} x_1 = 2 + 0x_2 - 0.25x_3 + 0.5x_3, \\ x_2 = -2 + 0.2x_1 + 0x_2 + 0.2x_3, \\ x_3 = -2 + 0.6x_1 + 0.2x_2 + 0x_3. \end{cases}$$

Підставляючи в систему довільне початкове наближення $X^{(0)} = (1,2,1)$ дістаємо

$$\begin{cases} x_1^{(1)} = 2 - 0.25 \cdot 2 + 0.5 \cdot 1 = 2, \\ x_2^{(1)} = -2 + 0.2 \cdot 1 + 0.2 \cdot 1 = -1.6, \\ x_3^{(1)} = -2 + 0.6 \cdot 1 + 0.2 \cdot 2 = -1. \end{cases}$$

Псевдокод алгоритму до методу простої ітерації

Вхідні дані:
n – порядок системи
α *– матриця коефіцієнтів перетвореної системи (4.27);*
$\vec{\beta}$ *– стовпчик перетвореної системи (4.27);*

eps – задана точність.
Вихід: X – вектор розв'язків системи.

1. *Оберати як початкове наближення* $X^{(0)}$ *вектор* β.
2. *Обчислити* $q = \|\alpha\|_m$.
3. *k = 1.*
4. *while(1){*
   a. *Обчислити наступне наближення* $X^{(k)}$

$$x_i^{(k)} = \beta_i + \sum_{j=1, j\neq i}^{n} \alpha_{ij} x_j^{(k-1)}, \; i = 1,2,...n$$

   b. *Обчислити m-норму norm вектора* $\| X^{(k)} - X^{(k-1)} \|$
   c. *if(norm ≤ eps*(1 - q) /q)*
      вихід.
   d. $X^{(k-1)} = X^{(k)}$.
   e. *k = k + 1*
   } // end while.

4.6. Метод ітерацій Зейделя

М***етод ітерацій Зейделя*** є модифікацією методу простої ітерації. Як і у методі простої ітерації систему перетворюють до вигляду

$$\vec{X} = \vec{\beta} + \alpha \vec{X}$$

за тими самими правилами.

Основна ідея методу полягає в тому, що при обчисленні $(k+1)$-го наближення невідомого $x_i^{(k+1)}$ враховуються $(k+1)$-і наближення $x_1^{(k+1)}, x_2^{(k+1)}, ..., x_{i-1}^{(k+1)}$. Вважаючи, що k-і наближення $x_j^{(k)}$, $j = 1, 2, ..., i,$ коренів відомі, будемо обчислювати $(k+1)$-е наближення за формулами

$$\begin{cases} x_1^{(k+1)} = \beta_1 + \sum_{j=1}^{n} \alpha_{1j} x_j^{(k)}, \\ x_2^{(k+1)} = \beta_2 + \alpha_{21} x_1^{(k+1)} + \sum_{j=2}^{n} \alpha_{2j} x_j^{(k)}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_i^{(k+1)} = \beta_i + \sum_{j=1}^{i-1} \alpha_{ij} x_j^{(k+1)} + \sum_{j=i}^{n} \alpha_{ij} x_j^{(k)}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n^{(k+1)} = \beta_n + \sum_{j=1}^{n-1} \alpha_{nj} x_j^{(k+1)} + \alpha_{nn} x_n^{(k)}. \end{cases}$$

Досить часто, але не завжди, метод Зейделя характеризується вищою збіжністю, ніж метод простої ітерації. Одна з умов збіжності ітерацій Зейделя визначається такою теоремою.

Теорема. *Я*кщо для системи лінійних рівнянь

$$\vec{X} = \vec{\beta} + \alpha \vec{X} \tag{4.31}$$

виконується умова $\| \alpha \|_m < 1$, де $|| \alpha || = \max\limits_i \sum\limits_{j=1}^{n} | \alpha_{ij} |$, то процес ітерації Зейделя для системи (4.31) збігається до єдиного розв'язку за будь-якого вибору початкового вектора $X^{(0)}$.

Отже, при використанні m-норми в умові збіжності ітераційного процесу, оцінка точності отриманого наближення обчислюється як

$$\left\| X^{(k+1)} - X^{(k)} \right\| \le \frac{1-q}{q} \varepsilon,$$

де $q = \| \alpha \|_m < 1$.

Приклад 4.5. Знайти методом ітерації Зейделя перше наближення розв'язку системи лінійних рівнянь

$$\begin{cases} 5x_1 - 4x_2 - x_3 = -2, \\ 4x_1 + x_2 - 2x_3 = 8, \\ 3x_1 + x_2 - 5x_3 = 10. \end{cases}$$

Розв'язок. Систему перетворимо до вигляду, коли виконуються умови збіжності, в такий же спосіб, як і у прикладі 4.7. Тоді маємо

$$\begin{cases} x_1^{(1)} = 2 + 0x_1^{(0)} - 0.25x_2^{(0)} + 0.5x_3^{(0)}, \\ x_2^{(1)} = -2 + 0.2x_1^{(1)} + 0x_2^{(0)} + 0.2x_3^{(0)}, \\ x_3^{(1)} = -2 + 0.6x_1^{(1)} + 0.2x_2^{(1)} + 0x_3^{(0)}. \end{cases}$$

Як початкове наближення оберемо вектор $X^{(0)} = (1, 2, 1)$, після підстановки якого отримуємо перше наближення

$$\begin{cases} x_1^{(1)} = 2 + 0 \cdot 1 - 0.25 \cdot 2 + 0.5 \cdot 1 = 2, \\ x_2^{(1)} = -2 + 0.2 \cdot 2 + 0 \cdot 2 + 0.2 \cdot 1 = -1.4, \\ x_3^{(1)} = -2 + 0.6 \cdot 2 + 0.2 \cdot (-1.4) + 0 \cdot 1 = -1.08. \end{cases}$$

Псевдокод алгоритму до методу ітерації Зейделя

Вхідні дані:

n – порядок системи

α – матриця коефіцієнтів перетвореної системи (4.31);

β – стовпчик перетвореної системи (4.31);

eps – задана точність.

Вихід: $\vec{X}$ *– вектор розв'язків системи.*

1. *Оберемо як початкове наближення* $X^{(0)}$ *вектор* $\vec{\beta}$.
2. *Обчислимо* $q = \|\alpha\|_m$.
3. *k = 1.*
4. *while(1){*
   a. *Обчислити наступне наближення* $X^{(k)}$

$$x_i^{(k)} = \beta_i + \sum_{j=1, j\neq i}^{i-1} \alpha_{ij} x_j^{(k)} + \sum_{j=i}^{n} \alpha_{ij} x_j^{(k-1)}, \ i = 1, 2, ...n.$$

   b. *Обчислити m-норму norm вектора* $\| X^{(k)} - X^{(k-1)} \|$
   c. *if(norm ≤ eps*(1 - q) / q)*
      вихід.
   d. $X^{(k-1)} = X^{(k)}$.
   e. *k = k + 1*
   } // end while.

4.7. Розв'язання систем лінійних алгебраїчних рівнянь методом Монте-Карло

Загальноприйнятого визначення методів Монте-Карло поки що немає. Звичайно методами Монте-Карло називаються чисельні методи розв'язання математичних задач за допомогою моделювання випадкових величин. Найбільш досліджені застосування методу Монте-Карло для таких задач: обчислення визначених інтегралів, розв'язання систем лінійних алгебраїчних рівнянь, обчислення обернених матриць, власних значень і власних векторів матриць і т.д.

Загальна ідея методів Монте-Карло полягає у зведенні задачі до розрахунку математичного сподівання. Щоб приблизно обчислити деяку скалярну величину a, необхідно знайти таку випадкову величину ξ, що $M\xi = a$, тоді маючи N незалежних значень $\xi_1, \xi_2, ..., \xi_N$ величини ξ можна скористатися законом великих чисел, відповідно до якого середнє арифметичне збігається до математичного сподівання. Тому можна вважати, що

$$a \approx \frac{1}{N}(\xi_1 + \xi_2 + ..., + \xi_N).$$

Нехай, наприклад, треба обчислити об'єм деякої обмеженої просторової фігури G. Побудуємо паралелепіпед P, що містить в собі G, об'єм якого $V_Đ$ відомий. Виберемо N випадкових точок, рівномірно розподілених у P, і позначимо як N' кількість точок, що потрапили в G. Якщо N велике, то очевидно, що

$$\frac{N'}{N} \approx \frac{V_G}{V_Đ}.$$

Звідси маємо

$$V_G \approx V_{Đ} \frac{N'}{N}.$$

У цьому прикладі випадкова величина ξ дорівнює $V_{Đ}$, якщо точка потрапляє в G, і ξ дорівнює 0, якщо точка не потрапляє в G. Тоді $M\xi = V_G$, а середнє арифметичне

$$\frac{1}{N}(\xi_1 + \xi_2 + ... + \xi_N) \approx V_P \frac{N'}{N}$$

визначає шуканий об'єм.

Ітераційні методи розв'язання систем лінійних алгебраїчних рівнянь вимагають збереження в оперативній пам'яті всіх компонентів розв'язку, що за великих порядків системи є обтяжливим. Тому у випадку дуже великого порядку системи може бути використаний метод Монте-Карло.

Нехай маємо систему лінійних алгебраїчних рівнянь

$$\sum_{j=1}^{n} à_{ij} x_{ij} = b_i, (i = 1,2,...,n).$$

Зведемо її до спеціального вигляду

$$\vec{X} = \vec{\beta} + \alpha \vec{X},$$

тобто до вигляду

$$\begin{cases} x_1 = \beta_1 + \alpha_{12} x_2 + ... + \alpha_{1j} x_j + ... + \alpha_{1n} x_n, \\ x_2 = \beta_2 + \alpha_{21} x_1 + + ... + \alpha_{2j} x_j + ... + \alpha_{2n} x_n, \\ ..... \\ x_n = \beta_n + \alpha_{n1} x_1 + \alpha_{n2} x_2 + ... \alpha_{n,n-1} x_{n-1}, \end{cases} \quad (4.32)$$

в такий спосіб, як і у випадку ітераційних методів розв'язання систем лінійних алгебраїчних рівнянь, причому для матриці α виконуються умови збіжності за m-нормою.

Підберемо систему множників v_{ij} таким чином, щоб числа p_{ij}, які визначаються з рівняння

$$\alpha_{ij} = p_{ij} v_{ij}, \; i, j = 1,...,n, \quad (4.33)$$

задовольняли таким умовам:

1) $p_{ij} \geq 0$, причому $p_{ij} > 0$, при $\alpha_{ij} \neq 0$;

2) $\sum_{j=1}^{n} p_{ij} < 1$, $i = 1,...,n$.

Нехай $p_{i,n+1} = 1 - \sum_{j=1}^{n} p_{ij}$, $i = 1,...,n$, крім того $p_{n+1,j} = 0$ при $j < n+1$ і $p_{n+1,n+1} = 1$.

Найпростіший спосіб визначення величин v_{ij}, які задовольняють рівнянню (4.33), полягає в тому, щоб вибрати $v_{ij} = 1$, якщо $\alpha_{ij} > 0$, і $v_{ij} = -1$, якщо $\alpha_{ij} < 0$.

Розглянемо деяку абстрактну «блукаючу» частинку, що має скінченну кількість можливих і несумісних станів $S_1, S_2, ..., S_n, S_{n+1}$. Ця частинка з імовірністю p_{ij}, $i, j = 1, ..., n+1$ переходить із стану S_i до стану S_j незалежно від попередніх станів. Стан $S_{n+1} = \Gamma$ (границя) є особливим і відповідає повній зупинці частинки, оскільки за умови $p_{n+1,j} = 0$ перехід із стану S_{n+1} до стану S_j ($j < n+1$) є неможливим. Процес переходів («блукання») припиняється, коли частинка вперше потрапляє на границю Г. Описану зміну станів називають ***дискретним ланцюгом Маркова*** із скінченною кількістю станів. Імовірності p_{ij} називають ***перехідними імовірностями***, а матрицю П цих імовірностей - ***матрицею переходу станів*** $\{S_i\}$

$$\Pi = \begin{bmatrix} p_{11} & \cdots & p_{1n} & p_{1,n+1} \\ \cdots & \cdots & \cdots & \cdots \\ p_{n1} & \cdots & p_{nn} & p_{n,n+1} \\ 0 & \cdots & 0 & 1 \end{bmatrix} \qquad (4.34)$$

Нехай S_i – деякий фіксований стан, що відрізняється від граничного. Розглянемо «блукання» частинки, яке починається зі стану $S_i = S_{i_0}$ і після ряду проміжних станів $S_{i_1}, S_{i_2}, ..., S_{i_m}$ закінчується на границі $S_{i_{m+1}} = \Gamma$. Сукупність станів

$$T_i = \{ S_{i_0}, S_{i_i}, ..., S_{i_m}, S_{i_{m+1}} \} \qquad (4.35)$$

назвемо ***траєкторією.*** Нехай X_i – випадкова величина, що залежить від випадкової траєкторії T_i, яка починається у стані S_i, і набуває для траєкторії (4.35) таких значень

$$\xi(T_i) = \beta_{i_0} + v_{i_0 i_1}\beta_{i_1} + v_{i_0 i_1}v_{i_1 i_2}\beta_{i_2} + ... + v_{i_0 i_1}...v_{i_{m-1} i_m}\beta_{i_m}, \qquad (4.36)$$

де β_j $(j = i_0, i_1, ..., i_m)$ - елементи вектора $\vec{\beta}$ системи (4.32). Зокрема, якщо всі $v_{ij} = 1$, то

$$\xi(T_i) = \beta_{i_0} + \beta_{i_i} + ... + \beta_{i_m}.$$

З теорії імовірностей відомо, що математичні сподівання випадкових величин, які обчислюють за формулою (4.36), є коренями системи (4.32).

Таким чином, якщо виконати *N* випадкових «блукань» з випадковими траєкторіями $T_i^{(k)}$ *(k = 1, 2,…,N)* і початковим станом S_i, а також щоразу обчислювати значення $\xi(T_i^{(k)})$, то тоді згідно з теоремою Чебишова (законом великих чисел) маємо

$$x_i = \frac{1}{N}\sum_{k=1}^{N}\xi(T_i^{(k)}) \qquad (4.37)$$

.

Розглянемо тепер, яким чином можна забезпечити «блукання» частинки. Нехай початковий стан частинки S_i, а $\{l\}$- випадкові числа, рівномірно розподілені на інтервалі [0,1]. Згенеруємо випадкове число l. Якщо далі виявиться, що виконані нерівності $0 \le l < p_{i1}$, будемо вважати, що частинка переходить із стану S_i у стан S_1. Якщо виконані нерівності $p_{i1} \le l < p_{i1} + p_{i2}$, то вважаємо, що частка переходить із S_i у S_2 і т.д. Врешті, частка потрапляє на границю $S_{n+1} = \Gamma$, якщо випадкове число l таке, що

$$p_{i1} + p_{i2} + \ldots + p_{in} \le l < p_{i1} + p_{i2} + \ldots + p_{i,n+1}.$$

Послідовно вибираючи початкові стани і виконуючи по *N* «блукань» у кожному випадку, обчислюємо згідно (4.37) корені системи (4.32).

Приклад 4.6. Розв'язати методом Монте-Карло систему рівнянь

$$\begin{cases} x_1 = 0,1x_1 + 0,2x_2 + 0,7, \\ x_2 = 0,2x_1 - 0,3x_2 + 1,1. \end{cases}$$

Розв'язок. Виберемо $v_{11} = 1$, $v_{12} = 1$, $v_{21} = 1$, $v_{22} = -1$. Тоді матриця переходів відповідно до (4.34) матиме вигляд

$$\Pi = \begin{bmatrix} 0,1 & 0,2 & 0,7 \\ 0,2 & 0,3 & 0,5 \\ 0 & 0 & 1 \end{bmatrix}.$$

Для обчислення невідомого x_1 виконаємо N = 20 «блукань» частинки, траєкторії якої починається в стані S_1. Траєкторії та відповідні їм випадкові величини наведені в табл. 4.1.

Таблиця 4.1. **Знаходження невідомого X_1 за методом Монте-Карло**

№ п/п	Випадкове число l	Траєкторія блукання частки	Значення випадкової величини T_i
1	0,5	$S_1 \to \Gamma$	0,7
2	0,7	$S_1 \to \Gamma$	0,7
3	0,7	$S_1 \to \Gamma$	0,7
4	0,0 0,5	$S_1 \to S_1 \to \Gamma$	0,7+0,7
5	0,7	$S_1 \to \Gamma$	0,7
6	0,1 0,6	$S_1 \to S_2 \to \Gamma$	0,7+1,1
7	0,1 0,8	$S_1 \to S_2 \to \Gamma$	0,7+1,1

8	0,7	$S_1 \to \Gamma$	0,7
9	0,3	$S_1 \to \Gamma$	0,7
10	0,7	$S_1 \to \Gamma$	0,7
11	0,1 0,0 0,7	$S_1 \to S_2 \to S_1 \to \Gamma$	0,7+1,1+0,7
12	0,0 0,1 0,3 0,1 0,1 0,6	$S_1 \to S_1 \to S_2 \to S_2 \to S_1 \to S_2 \to I$	0,7+0,7+1,1-1,1-0,7-1,1
13	0,9	$S_1 \to \Gamma$	0,7
14	0,6	$S_1 \to \Gamma$	0,7
15	0,1 0,5	$S_1 \to S_2 \to \Gamma$	0,7+1,1
16	0,3	$S_1 \to \Gamma$	0,7
17	0,3	$S_1 \to \Gamma$	0,7
18	0,2 0,4 0,4 0,3 0,1 0,6	$S_1 \to S_2 \to S_2 \to S_2 \to S_2$ $\to S_1 \to \Gamma$	0,7+1,1-1,1+1,1-1,1-0,7
19	0,6	$S_1 \to \Gamma$	0,7
20	0,2 0,6	$S_1 \to S_2 \to \Gamma$	0,7+1,1
$\sum$		$21 \cdot 0,7 + 4 \cdot 1,1$	

Згідно з теоремою Чебишова маємо

$$x_1 = \frac{1}{N} \sum_{k=1}^{N} \xi(T_1^{(k)}) \approx \frac{1}{20}(20 \cdot 0,7 + 0,7 + 4 \cdot 1,1) = 0.96 .$$

Для обчислення x_2 треба виконати таку ж кількість випадкових траєкторій, що починаються зі стану S_2. Зазначимо, що розв'язком заданої системи є $x_1 = x_2 = 1$.

Псевдокод алгоритму розв'язку систем лінійних алгебраїчних рівнянь за методом Монте-Карло

Вхідні дані:
n – порядок системи
α *– матриця коефіцієнтів перетвореної системи (4.32);*
β *– стовпчик перетвореної системи (4.32);*
N – кількість траєкторій для кожного невідомого.
Вихід: X – вектор розв'язків системи.

1. *Побудувати матрицю* Π *ймовірностей переходів* p_{ij}.
2. *Побудувати матрицю V множників* v_{ij}.
3. *for(i = 1; i < n; i = i+1){*
   sum = 0;
   for(j = 1; j < N; j = j+1){
   a. *Побудувати траєкторію* T_i^j, *яка починається в стані* S_i.
   b. *Обчислити значення* $\xi(T_i^j)$ *траєкторії* T_i^j *у відповідності з* (4.36).
   c. *sum = sum +* $\xi(T_i^j)$,
   d. *Обчислити невідоме* $x_i = sum / N$.
   }
   }

Контрольні запитання та завдання до розділу 4

1. Що таке система лінійних алгебраїчних рівнянь?
2. Що є розв'язком системи?
3. На які групи поділяють методи розв'язання систем лінійних рівнянь?
4. Що таке прямий та зворотний хід методу Гауса?
5. Наведіть оцінку обчислювальної складності схеми єдиного поділу.
6. Що є головним недоліком схеми єдиного поділу?
7. Систему рівнянь

$$\begin{cases} 2x_1 + 2x_2 - x_3 = 3, \\ 2x_1 + 3x_2 - x_3 = 4, \\ 8x_1 + 5x_2 - 3x_3 = 10. \end{cases}$$

   розв'язати за схемою єдиного поділу.
8. У чому полягає ідея виключення Гауса-Жордана?
9. Систему рівнянь

$$\begin{cases} 2x_1 + 2x_2 - x_3 = 3, \\ 2x_1 + 3x_2 - x_3 = 4, \\ 8x_1 + 5x_2 - 3x_3 = 10. \end{cases}$$

розв’язати з використанням виключення Гауса-Жордана.

10 .Які переваги має схема з вибором головного елемента?

11. У чому полягає суть методу простої ітерації?

12. Наведіть оцінку складності обчислень за методом простої ітерації.

13. Яким чином виконують перетворення системи до вигляду, придатного для ітерації?

14. Систему рівнянь

$$\begin{cases} x_1 - 3x_2 + x_3 = 1, \\ x_1 - x_2 + 4x_3 = 4, \\ x_1 - 2x_2 + x_3 = 0. \end{cases}$$

звести до вигляду, придатного для ітерації.

15. Наведіть оцінку точності отриманого наближення для методу простої ітерації.

16. У чому полягає відмінність ітерації Зейделя від простої ітерації?

17. На якій основній ідеї базуються методи Монте-Карло?

18. Як метод Монте-Карло застосовують для розв’язання систем лінійних рівнянь?

РОЗДІЛ 5. НАБЛИЖЕННЯ ФУНКЦІЙ ЗА ДОПОМОГОЮ ІНТЕРПОЛЯЦІЇ

5.1. Постановка задачі інтерполяції

Однією з основних задач чисельних методів є задача відновлення значень функції, тобто обчислення її приблизних значень в певних точках. На практиці в багатьох випадках функція задається у вигляді таблиці, яка є результатом деяких вимірювань. При цьому часто виникає потреба обчислити значення таблично заданих функцій у проміжних точках (тобто відсутніх в таблиці). Крім того, може виникнути потреба в обчисленні похідної таблично заданої функції. Наприклад, якщо в таблиці задано шлях, пройдений матеріальною точкою в певні моменти часу і треба овизначити швидкість точки. Одним із способів роз'вязання таких задач є інтерполяція.

Іншим прикладом застосування інтерполяції є ситуація, коли маємо дуже складну щодо обчислення функцію. В цьому випадку може мати сенс побудова таблиці значень цієї функції в потрібному діапазоні і подальше відновлення її значень з використанням інтерполяції.

Нехай відомі значення $f_i = f(x_i),\ i = 0,1,...,n,$ деякої функції $f(x)$ в $n+1$ різних точках $x_0,\ x_1,...,\ x_n$ і виникає задача наближеного відновлення функції $f(x)$ у довільній точці $x\ x \in (x_0; x_n)$. Для її роз'вязання необхідно побудувати алгебраїчний поліном $L_n(x)$ степеня n, який у точках x_i набуває заданих значень, тобто

$$L_n(x_i) = f_i,\ i = 0,1,...,n. \tag{5.1}$$

Такий поліном називають ***інтерполяційним*** (рис. 5.1), а точки $x_i,\ i = 0,1,...,n$ називають ***вузлами інтерполяції***. Наближене відновлення $f(x)$ за формулою

$$f(x) \approx L_n(x) \tag{5.2}$$

називають ***інтерполяцією***. Якщо точка x розміщена поза відрізком, що містить всі вузли інтерполяції $x_0, x_1, ..., x_n$, то заміну (5.2) називають ***екстраполяцією***. Доведено, що існує єдиний інтерполяційний многочлен n–го степеня, що задовольняє умовам (5.1).

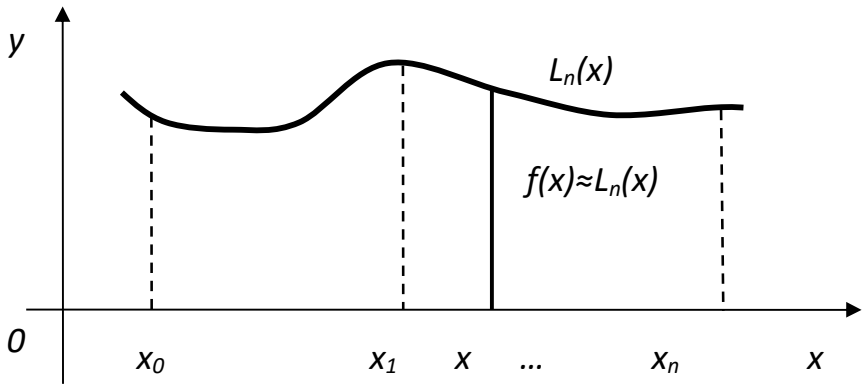


Рис.5.1. Геометрична інтерпретація інтерполяції

5.2. Інтерполяційний поліном Лагранжа. Похибки інтерполяції

Алгебраїчний поліном вигляду

$$L_n(x) = \sum_{i=0}^{n} p_{ni} f_i, \tag{5.3}$$

де

$$p_{ni} = \frac{(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}, \tag{5.4}$$

називають ***інтерполяційним поліномом Лагранжа***, а функції (5.4) – лагранжевими коефіцієнтами.

Зауваження. Оскільки інтерполяційний поліном Лагранжа лінійно залежить від f_i, інтерполяційний многочлен суми двох функцій дорівнює сумі двох інтерполяційних поліномів.

Приклад 5.1. Побудувати поліном Лагранжа для функції, заданої таблицею,

x_i	0	2	3	5
f_i	1	3	2	5

Розв'язок. У цьому випадку $n=3$, відповідно, поліном Лагранжа набуде вигляду

$$L_3(x) = \frac{(x-2)(x-3)(x-5)}{(0-2)(0-3)(0-5)}\cdot 1 + \frac{(x-0)(x-3)(x-5)}{(2-0)(2-3)(2-5)}\cdot 3 +$$
$$\frac{(x-0)(x-2)(x-5)}{(3-0)(3-2)(3-5)}\cdot 2 + \frac{(x-0)(x-2)(x-3)}{(5-0)(5-2)(5-3)} = 1 + \frac{62}{15}x - \frac{13}{6}x^2 + \frac{3}{10}x^3$$

Приклад 5.2. Для функції, заданої далі таблицею,

x_i	-3	-1	2
f_i	-5	-11	10

обчислити значення функції в точці $x = 0$.

Розв'язок. Оскільки $n = 2$, маємо

$$L_2(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} f_0 + \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} f_1 + \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} f_2.$$

Отже для $x = 0$

$$L_2(0) = \frac{(0+1)(0-2)}{(-3+1)(-3-2)}(-5) + \frac{(0+3)(0-2)}{(-1+3)(-1-2)}(-11) +$$
$$+ \frac{(0+3)(0+1)}{(2+3)(2+1)}(10) = -8.$$

Якщо у виразі для $L_2(x)$ розкрити дужки і звести подібні доданки, то $L_2(x) = 2x^2 + 5x - 8$. Насправді вихідну таблицю для цього прикладу побудовано для функції $f(x) = 2x^2 + 5x - 8$. Звідси випливає важливе спостереження: якщо функція, яку інтерполюють, є алгебраїчним поліномом *n*–го степеня, то інтерполяційний поліном того ж самого степеня, побудований на її основі, тотожно збігається із самою функцією, тобто

$$L_n(x) \equiv f(x).$$

Похибка інтерполяції. Говорити про похибку інтерполяції має сенс тільки у випадку, коли відоме аналітичне подання функції $f(x)$. Очевидно, що існує рівність

$$f(x) = L_n(x) + R_n(x),$$

де $R_n(x)$ – залишковий член, або похибка інтерполяції. Виявляється, що

$$R_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \Pi_{(n+1)}(x),$$

де $\Pi_{(n+1)}(x) = (x - x_0)(x - x_1)...(x - x_n)$, а $\xi \in [x_0, x_n]$ – невідома точка. Тоді маємо

$$f(x) = L_n(x) + \frac{f^{(n+1)}(\xi)}{(n+1)!} \Pi_{(n+1)}(x). \qquad (5.5)$$

Якщо максимізувати перший співмножник в (5.5), отримаємо верхню границю залишкового члена - оцінку похибки інтерполяції в заданій точці x:

$$\left| f(x) - L_n(x) \right| \le \frac{M_{n+1}}{(n+1)!} \left| \prod_{n+1}(x) \right|,$$

де

$$M_{n+1} = \max_{x_0 \le x \le x_n} | f^{(n+1)}(x) | .$$

Щоб отримати верхню границю залишкового члена на всьому проміжку $[x_0, x_n]$, треба максимізувати другий співмножник. Тоді оцінка максимальної похибки на інтервалі інтерполювання $[x_0, x_n]$ матиме вигляд

$$\max \left| f(x) - L_n(x) \right| \le \frac{M_{n+1}}{(n+1)!} \max_{[x_0, x_n]} \left| \Pi_{n+1}(x) \right|. \qquad (5.6)$$

Приклад 5.3. Оцінити похибку наближення функції $f(x) = \sqrt{x}$ в точці $x = 116$ і на проміжку інтерполювання [100,144] за допомогою інтерполяційного полінома Лагранжа другого степеня, побудованого в точках $x_0 = 100,\ x_1 = 121,\ x_2 = 144.$

Розв'язок. Знаходимо похідні

$$f'(x) = \frac{1}{2} x^{-1/2};\ f''(x) = -\frac{1}{4} x^{-3/2};\ f'''(x) = \frac{3}{8} x^{-5/2}.$$

Отже,

$$M_3 = \max_{[100,144]} | f'''(x) | = \frac{3}{8} 100^{-5/2} = \frac{3}{8} 10^{-5}.$$

На основі формули (5.5) отримуємо:

$$\left| \sqrt{116} - L_2(116) \right| \le \frac{3}{8} 10^{-5} \frac{1}{3!} (116-100)(116-121)(116-144) = 1.4 \cdot 10^{-3}.$$

Відповідно до (5.6) остаточно маємо:

$$\max_{[100,144]} \left| \sqrt{x} - L_2(x) \right| \le \frac{10^{-5}}{16} \cdot \left| (x-100)(x-121)(x-144) \right| = 2.5 \cdot 10^{-3}.$$

Відзначимо, що в цьому прикладі оцінки похибок отримані без побудови самого інтерполяційного полінома. Отже, у разі потреби можна заздалегідь визначити необхідний степінь полінома, що забезпечує допустиму похибку інтерполяції.

Лінійна інтерполяція. У випадку $n = 1$ інтерполяцію, що виконується за допомогою полінома Лагранжа, називають ***лінійною***. Позначаючи $h = x_1 - x_0$, $q = \dfrac{(x - x_0)}{h}$, формулу лінійної інтерполяції можна звести до вигляду

$$f(x) \approx L_1(x) = L_1(x_0 + qh) = (1-q) f_0 + q f_1.$$

Величину q називають ***фазою інтерполяції***, яка змінюється в межах від 0 до 1, коли x пробігає діапазон від x_0 до x_1. Геометрично лінійна інтерполяція – це хорда (рис. 5.2), що проходить через точки (x_0, f_0) і (x_1, f_1). При цьому $\ddot{I}_2(x) = (x - x_0)(x - x_1)$ і, відповідно,

$$\max_{[x_0, x_1]} | \Pi_2(x) | = \frac{h^2}{4}.$$

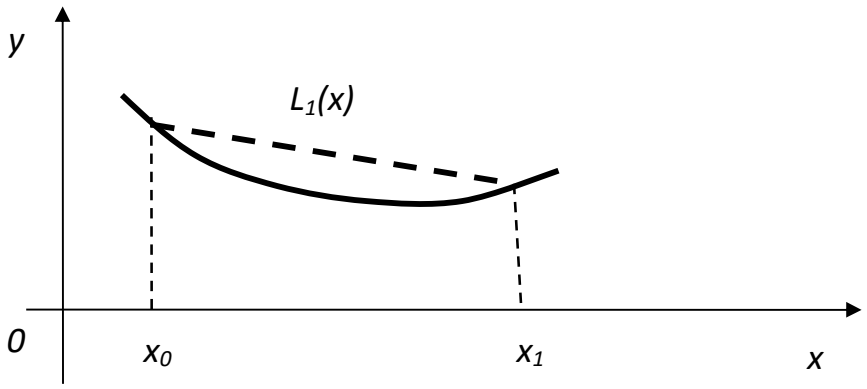


Рис. 5.2. Геометрична інтерпретація лінійної інтерполяції

Тоді оцінка максимальної похибки лінійної інтерполяції на $[x_0, x_1]$ матиме вигляд

$$\max_{[x_0,x_1]} | f(x) - L_1(x) | \le h^2 \frac{M_2}{8}.$$

Таблиці, які задають вузли інтерполяції, містять велику їх кількість. У випадку лінійної інтерполяції обирають два найближчих до x вузла. Лівий з них – x_0, правий – x_1.

Лінійну інтерполяцію вважають допустимою, якщо внесена нею додаткова похибка, менша за похибку таблиці, що задає функцію. Якщо похибка таблиці дорівнює $0.5 \cdot 10^{-m}$, де m – номер останнього розряду в таблиці, достатньою умовою застосування лінійної інтерполяції є $\frac{M_2 h^2}{8} < 0,5 \cdot 10^{-m}$ або $M_2 h^2 < 4 \cdot 10^{-m}$. Крок і точність таблиці слід обирати таким чином, щоб лінійна інтерполяція була допустимою.

Схема Ейткена. Якщо необхідно обчислити наближене значення функції $f(x)$ для деякого значення x без явної побудови інтерполяційного полінома, зручно використовувати спосіб Ейткена. Нехай

$$\varphi_{i,i+1}(x) = \frac{1}{x_{i+1} - x_i} \begin{vmatrix} f_i & x_i - x \\ f_{i+1} & x_{i+1} - x \end{vmatrix},$$

$$\varphi_{i,i+1,i+2}(x) = \frac{1}{x_{i+2} - x_i} \begin{vmatrix} \varphi_{i,i+1}(x) & x_i - x \\ \varphi_{i+2,i+1}(x) & x_{i+2} - x \end{vmatrix},$$

$$\varphi_{i,i+1,i+2,i+3}(x) = \frac{1}{x_{i+3} - x_i} \begin{vmatrix} \varphi_{i,i+1,i+2}(x) & x_i - x \\ \varphi_{i+1,i+2,i+3}(x) & x_{i+3} - x \end{vmatrix} \text{ і т.д.}$$

Тоді інтерполяційний поліном n-го степеня, що набуває в точках $x_0, x_1, ..., x_n$ значень $f_0, f_1, ..., f_n$, можна записати наступним чином

$$\varphi_{0,1...n}(x) = \frac{1}{x_n - x_0} \begin{vmatrix} \varphi_{0,1...n-1}(x) & x_0 - x \\ \varphi_{0,1...n}(x) & x_n - x \end{vmatrix}.$$

Звідси випливає схема обчислень функції в заданій точці (табл. 5.1). Умова припинення обчислень визначається порівнянням послідовних значень $\varphi(x)$. Обчислення потрібно припинити, коли в межах заданої точності збільшення степеня многочлена не призводить до уточнення $\varphi(x)$. Це збезпечує можливість автоматичного управління обчисленнями.

Таблиця 5.1. **Схема Ейткена для обчислення значень функції в заданій точці**

x_i	f_i	$x_i - x$	$\varphi_{i-1,i}$	$\varphi_{i-2,i-1,i}$	$\varphi_{i-3,i-2,i-1,i}$	…
x_0	f_0	$x_0 - x$				
x_1	f_1	$x_1 - x$	$\varphi_{01}(x)$			
x_2	f_2	$x_2 - x$	$\varphi_{12}(x)$	$\varphi_{012}(x)$		
x_3	f_3	$x_3 - x$	$\varphi_{23}(x)$	$\varphi_{123}(x)$	$\varphi_{0123}(x)$	
x_4	f_4	$x_4 - x$	$\varphi_{34}(x)$	$\varphi_{234}(x)$	$\varphi_{1234}(x)$	φ_{01234}

Приклад 5.4. Знайти наближене значення функції, заданої таблицею, в точці $x = 1$.

x_i	0	2	3	4
y_i	0	2	3	4

Розв'язок. Побудуємо таблицю для схеми Ейткена (табл. 5.2).

Таблиця 5.2- **Обчислення за схемою Ейткена**

x_i	f_i	$x_i - x$	$\varphi_{i,i+1}$	$\varphi_{i,i+1,i+2}$
0	0	-1		
2	2	1		
3	3	2	1	
4	4	3	1	1

Отже

$$\varphi_{01} = \frac{1}{x_1 - x_0} \begin{vmatrix} 0 & -1 \\ 2 & 1 \end{vmatrix} = \frac{1}{2}(0 + 2) = 1,$$

$$\varphi_{12} = \frac{1}{x_2 - x_1}\begin{vmatrix} f_1 & x_1 - 1 \\ f_2 & x_2 - 1 \end{vmatrix} = \frac{1}{1}\begin{vmatrix} 2 & 0 \\ 3 & 2 \end{vmatrix} = 1,$$

$$\varphi_{012} = \frac{1}{x_2 - x_0}\begin{vmatrix} \varphi_{01} & x_0 - 1 \\ \varphi_{12} & x_2 - 1 \end{vmatrix} = \frac{1}{3}\begin{vmatrix} 2 & 0 \\ 3 & 2 \end{vmatrix} = \frac{1}{3} \cdot 3 = 1.$$

Тобто $f(1) = 1.$

5.3. Інтерполяційний поліном Ньютона

Введемо поняття скінченної та розділеної різниці. Нехай функція $f(x)$ задана у рівновіддалених точках $x_k = x_0 + kh,$ де k – ціле, $h>0.$ Величину

$$\Delta f_k = f(x_k + h) - f(x_k) = f(x_{k+1}) - f(x_k) = f_{k+1} - f_k$$

називають ***скінченною різницею першого порядку*** функції $f(x)$ в точці x_k, а величину

$$\Delta^2 f_k = \Delta(\Delta f_k) = \Delta f_{k+1} - \Delta f_k$$

- ***скінченною різницею другого порядку*** точці x_k.

Скінченною різницею n***-го порядку*** називають різницю між двома сусідніми скінченними різницями $(n-1)$-го порядку і визначають за рекурентною формулою:

$$\Delta^n f_k = \Delta^{n-1} f_{k+1} - \Delta^{n-1} f_k.$$

Скінченні різниці є функціями, які звичайно задають у вигляді таблиць (наприклад , табл.5.3).

Таблиця 5.3- **Запис скінченних різниць**

x_0	f_0				
x_1	f_1	Δf_0			
x_2	f_2	Δf_1	$\Delta^2 f_0$		
x_3	f_3	Δf_2	$\Delta^2 f_1$	$\Delta^3 f_0$	
x_4	f_4	Δf_3	$\Delta^2 f_2$	$\Delta^3 f_1$	$\Delta^4 f_0$

Приклад 5.5. Необхідно побудувати скінченні різниці для функції $f = x^3$ з кроком $h = 1$.

Розв'язок.

$$\Delta f = (x+1)^3 - x^3 = 3x^2 + 3x + 1,$$

$$\Delta^2 f = [3(x+1)^2 + 3(x+1) + 1] - (3x^2 + 3 + 1) = 6x + 6,$$

$$\Delta^3 f = [6(x+1) + 6] - (6x + 6) = 6,$$

$$\Delta^k f = 0 \text{ при } k > 3.$$

Взагалі, якщо $f(x) = a_0 x^n + a_1 x^{n-1} + ... + a_0 -$ поліном n-го степеня, то $\Delta^n f = n! a_0 h^n = const$.

Властивості скінченних різниць багато в чому аналогічні властивостям похідних. По-перше, скінченні різниці першого порядку від полінома n-го степеня, є многочленом степеня $(n-1)$, а скінченні різниці n-го порядку від цього многочлена - сталими. По-друге,

$$\Delta^m (C_1 f \pm C_2 g) = C_1 \Delta^m f \pm C_2 \Delta^m g .$$

Скінченна різниця і похідна m-го порядку пов'язані відношенням

$$\Delta^m f_k = h^m f^{(m)}(\xi),\ \xi \subset [x_k, x_{k+m}].$$

При обчисленні скінченних різниць, виходячи з наближених значень функції, похибка різниць швидко зростає. Так, якщо похибка табличного значення функції дорівнює половині одиниці останнього розряду, то похибка скінченної різниці першого порядку складає одиницю останнього розряду, скінченна різниця другого порядку – дві одиниці, третього порядку – чотири одиниці і т.д. Похибка m-го порядку складе 2^{m-1} одиниць останнього порядку. Тому, якщо на деякому інтервалі таблиці всі різниці m-го порядку відрізняються не більше, аніж на 2^m одиниць останнього розряду, то такі різниці вважають практично постійними.

Розділені різниці. Розділені різниці застосовуються у випадку нерівновіддалених інтерполяційних вузлів. Нехай $x_0, x_1, ..., x_n$ - довільні вузли осі x, причому $x_i \neq x_j$ при $i \neq j$. Значення $f(x_0), f(x_1), ..., f(x_n)$ функції $f(x)$ називають ***розділеними різницями*** 0-го порядку. Число

$$f(x_0; x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0}$$

називають розділеною різницею 1-го порядку функції $f(x)$. Очевидно, що $f(x_0; x_1) = f(x_1; x_0)$, тобто, розділена різниця 1-го порядку є симетричною функцією аргументів x_0 та x_1. Розділена різниця 2-го порядку визначається через розділені різниці 1-го порядку в такий спосіб

$$f(x_0; x_1; x_2) = \frac{f(x_1; x_2) - f(x_0; x_1)}{x_2 - x_0}.$$

Нарешті розділена різниця *n*-го порядку визначається через розділені різниці *(n -1)*-го порядку

$$f(x_0; x_1; ...; x_n) = \frac{f(x_1; x_2; ..; x_n) - f(x_0; x_1; ...; x_{n-1})}{x_n - x_0}.$$

Подібно до скінченних, розділені різниці можна подавати у вигляді таблиці (приклад - табл.5.4).

Таблиця 5.4. **Запис розділених різниць**

x_0	$f(x_0)$			
x_1	$f(x_1)$	$f(x_0;x_1)$		
x_2	$f(x_2)$	$f(x_1;x_2)$	$f(x_0;x_1;x_2)$	
x_3	$f(x_3)$	$f(x_2;x_3)$	$f(x_1;x_2;x_3)$	$f(x_0;x_1;x_2;x_3)$

Властивості розділених різниць аналогічні властивостям скінченних різниць. Більше того, якщо $x_k = x_0 + kh$, $k = 0,1,...,m$, $h > 0$, то

$$f(x_0;x_1;...;x_n) = \frac{\Delta^n f_0}{n!h^n}.$$

Нехай $x_0, x_1, ..., x_n$ – довільні попарно відмінні вузли, для яких відомі значення функції $f(x)$. Алгебраїчний многочлен *n*-го степеня

$$P_n(x)=f(x_0)+(x-x_0)f(x;x_1)+(x-x_0)(x-x_1)f(x;x_1;x_2)+\ldots$$

$$\ldots+(x-x_0)(x-x_1)\ldots(x-x_{n-1})f(x;x_1;\ldots;x_n) \qquad (5.7)$$

є інтерполяційним, тобто

$$P_n(x_i) = f_i, \quad i = 0,1,2,...,n.$$

Многочлен (5.7) називають також **інтерполяційним поліномом Ньютона для нерівних проміжків**.

Приклад 5.6. Побудувати інтерполяційний поліном Ньютона другого степеня для функції, заданою таблицею

x_i	-3	-1	2
f_i	-5	-11	10

Розв'язок. Складемо таблицю розділених різниць.

Таблиця 5.5- **Таблиця розділених різниць**

x_i	$f(x_i)$	$f(x_i;x_{i+1})$	$f(x_i;x_{i+1};x_{i+2})$
-3	-5		
-1	-11	-3	
2	10	7	2

Отже, інтерполяційний поліном Ньютона має вигляд

$$P_2(x) = -5 + (x-(-3))\cdot(-3) + (x-(-3))(x-(-1))\cdot 2 = -5 - 3\cdot(x+3) + 2\cdot(x+3)(x+1) =$$

$$= -5 - 3x - 9 + 2x^2 + 2x + 6x + 6 = 2x^2 + 5x - 8.$$

Як і очікувалося, отриманий результат за побудовою ідентичний поліному, отриманому за допомогою формули Лагранжа. Різниця тільки полягає в тому, що, як видно з (5.4), в інтерполяційному поліномі Лагранжа кожен з його коефіцієнтів залежить від всіх значень x_i, $i = 0,1,...,n$. Це означає, що при збільшені кількості вузлів інтерполяції необхідно перераховувати всі лагранжеві коефіцієнти. В той же час, як видно з (5.7), при збільшені степеня полінома Ньютона достатньо обчислити лише наступний доданок.

Випадок рівновіддалених вузлів. Високої точності наближення функції можна досягти двома шляхами:

1) додавати нові вузли інтерполяції й інтерполювати функцію по всіх вузлах поліномом максимально високого степеня;

2) після додавання нових вузлів значення функції обчислюють шляхом інтерполяції за найближчими вузлами, а степінь інтерполяційного полінома при не цьому збільшують.

Вузли таблиці звичайно вибирають рівновіддаленими. Якщо вузли обираються в околі точки x, де обчислюється значення функції, то проміжна точка ξ в оцінці залишкового члена також знаходиться в околі точки x, і таким чином, величина $f^{(n+1)}(\xi)$ змінюється не дуже сильно на проміжку інтерполювання. Оскільки похибка оцінюється як

$$R_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\cdot \prod_{n+1}(x),$$

то, вирішальний вплив на її значення має величина $|\Pi_{n+1}(x)| = |(x-x_0)(x-x_1)...(x-x_n)|$, тобто добуток відстаней від точки x до вузлів інтерполяції. Величина $\Pi_{n+1}(x)$ буде мінімальною, якщо взяти n вузлів, найближчих до x (в ідеалі – по обидві сторони від x). Нехай $x_k = x_0 + kh$, $k = 0,1,...,n$, $f_k = f(x_k)$ і $q = \dfrac{x - x_0}{h}$. Враховуючи зв'язок розділених та скінченних різниць, інтерполяційний поліном (5.7) можна переписати як

$$P_n(x) = l_n(x_0 + qh) = f_0 + q\frac{\Delta f_0}{1!} + q(q-1)\frac{\Delta^2 f_0}{2!} + \cdots + q(q-1)\ldots(q-n+1)\frac{\Delta^n f_0}{n!} \quad . \quad (5.8)$$

Цей інтерполяційний поліном називають також ***першою інтерполяційною формулою Ньютона***. Застосовують інтерполяційний поліном (5.8) для так званої «інтерполяції вперед», коли точка x розташована ближче до початку таблиці, або для «екстраполяції назад» на один-два кроки. В останньому випадку мається на увазі, що точка x розташована на невеликій відстані лівіше x_0. Залишковий член цього інтерполяційного полінома

$$R_n(x) = h^{n+1}\cdot \Pi_{n+1}(q)\cdot\frac{f^{(n+1)}(\xi)}{(n+1)!},$$

де $\Pi_{n+1}(q) = q(q-1)...(q-n)$.

Змінимо спосіб нумерації вузлів. Інтерполяційний поліном з вузлами $x_{-n}, x_{-(n-1)}, ..., x_{-2}, x_{-1}, x_0$, де $x_{-k} = x_0 - kh$ має вигляд

$$Q_n(x) = Q_n(x_0 + qh) = f_0 + q\frac{\Delta f_{-1}}{1!} + q(q+1)\frac{\Delta^2 f_{-2}}{2!} + ...$$
$$... + q(q+1)...(q+n-1)\frac{\Delta^n f_{-n}}{n!}. \qquad (5.9)$$

Його залишковий член

$$R_n(x) = h^{n+1} \cdot q(q+1)...(q+n)\frac{\Delta^{n+1} f(\xi)}{(n+1)!}.$$

Інтерполяційний поліном (5.9) називають ***другою інтерполяційною формулою Ньютона.*** Застосовується він для «інтерполяції назад», коли точка x розташована ближче до кінця x_0 таблиці, або для «екстраполяції вперед» на один-два кроки. В останньому випадку мається на увазі, що точка x розташована правіше x_0 на невеликій відстані, а в (5.9) змінюється нумерація вузлів: x_{-n} - це самий лівий вузол, x_0 - самий правий. У зв'язку з цим таблиця скінченних різниць має вигляд, відмінний від табл.5.4, а саме: скінченні різниці в (5.9) розташовані на діагоналі зліва вверх.

Таблиця 5.6 - **Таблиця скінченних різниць для другої інтерполяційної формули Ньютона**

x_i	f_i	Δf_i	$\Delta^2 f_i$	$\Delta^3 f_i$
x_{-4}	f_{-4}	Δf_{-4}	$\Delta^2 f_{-4}$	$\Delta^3 f_{-4}$
x_{-3}	f_{-3}	Δf_{-3}	$\Delta^2 f_{-3}$	$\Delta^3 f_{-3}$
x_{-2}	f_{-2}	Δf_{-2}	$\Delta^2 f_{-2}$	
x_{-1}	f_{-1}	Δf_{-1}		
x_0	f_0			

Приклад 5.7. За таблицею значень функції $f(x) = \lg x$ обчислити $\lg 1044$.

	x_i	f_i
x_{-5}	1000	3.0000000
x_{-4}	1010	3.0043214
x_{-3}	1020	3.0086002
x_{-2}	1030	3.0128372
x_{-1}	1040	3.0170333
x_0	1050	3.0211893

Розв'язок. Виберемо $x_0 = 1050$, тоді $q = \dfrac{x - x_0}{h} = \dfrac{1044 - 1050}{10} = -0.6$, $h = 10$. Складемо таблицю скінченних різниць.

Таблиця 5.7=**Таблиця скінченних різниць**

x_i	f_i	Δf_i	$\Delta^2 f_i$	$\Delta^3 f_i$
1000	3.0000000	43214	-426	8
1010	3.0043214	42788	-418	9
1020	3.0086002	42370	-409	8
1030	3.0128372	41961	-401	
1040	3.0170333	41560		
1050	3.0211893			

Згідно з формулою (5.9) маємо

$$\lg 1044 = 3.0211893 + (-0.6)\cdot 0.0041560 + \frac{(-0.6)(-0.6+1)}{2}\cdot 0.0000401 +$$
$$+ \frac{(-0.6)(-0.6+1)(-0.6+2)}{6}\cdot 0.0000008 = 3.0187005$$

5.4. Інтерполяційні формули Гауса

Якщо в таблиці є достатнє число вузлів з кожної сторони від x, доцільно вузли $x_0, x_1, ..., x_n$ обрати так, щоб точка x виявилася якомога ближче до середини мінімального відрізка, що містить $x_0, x_1, ..., x_n$. Тоді маємо таблицю ***центральних різниць*** (табл.5.8).

Перша інтерполяційна формула Гауса. Нехай є $2n+1$ рівновіддалених вузлів $x_{-n}, x_{-(n-1)}, ..., x_{-1}, x_0, x_1, ..., x_{n-1}, x_n$, де $x_{i+1} - x_i = \Delta x_i = h - const$, $(i = -n, -(n-1), ..., n-1, n)$. Тоді інтерполяційний поліном набуває вигляду

$$T_n(x) = f_0 + q\Delta f_0 + \frac{q(q-1)}{2!}\Delta^2 f_{-1} + \frac{(q+1)q(q-1)}{3!}\Delta^3 f_{-1} +$$
$$+ \frac{(q+1)q(q-1)(q-2)}{4!}\Delta^4 f_{-2} + \frac{(q+2)(q+1)q(q-1)(q-2)}{5!}\Delta^5 f_{-2} + \cdots$$
$$\ldots + \frac{(q+n-1)\ldots(q-n+1)}{(2n-1)!}\Delta^{2n-1} f_{-(n-1)} + \frac{(q+n-1)\ldots(q-n)}{(2n)!}\Delta^{2n} f_{-n}.$$

Неважко помітити, що в цій формулі містяться центральні різниці $\Delta f_0, \Delta^2 f_{-1}, \Delta^3 f_{-1}, \Delta^4 f_{-2}, \Delta^5 f_{-2} \ldots$ (нижня ламана в табл.5.8).

Друга інтерполяційна формула Гауса містить у собі скінчені різниці $\Delta f_{-1}, \Delta^2 f_{-1}, \Delta^4 f_{-2}, \Delta^5 f_{-3}$ (верхня ламана в табл.5.8).

$$P_n(x) = f_0 + q\Delta f_{-1} + \frac{q(q+1)}{2!}\Delta^2 f_{-1} + \frac{(q+1)q(q-1)}{3!}\Delta^3 f_{-2} + \frac{(q+2)(q+1)q(q-2)}{4!}\Delta^4 f_{-2} + \ldots$$

$$\ldots + \frac{(q+n-1)\ldots(q-n+1)}{(2n-1)!}\Delta^{2n-1} f_{-n} + \frac{(q+n)(q+n-1)\ldots(q-n+1)}{(2n)!}\Delta^{2n} f_{-n}.$$

Таблиця 5.8 - **Таблиця центральних різниць**

x	F	Δf	$\Delta^2 f$	$\Delta^3 f$	$\Delta^4 f$	$\Delta^5 f$	
x_{-3}	f_{-3}						
		Δf_{-3}					
x_{-2}	f_{-2}		$\Delta^2 f_{-3}$				
		Δf_{-2}		$\Delta^3 f_{-3}$			
x_{-1}	f_{-1}		$\Delta^2 f_{-2}$		$\Delta^4 f_{-3}$		
		Δf_{-1} ↘	↗	$\Delta^3 f_{-2}$ ↘	↗	$\Delta^5 f_{-3}$ ↘	*2-га*
x_0	f_0 ↗↘		$\Delta^2 f_{-1}$ ↘		$\Delta^4 f_{-2}$ ↘	↗	
		Δf_0 ↗		$\Delta^3 f_{-1}$ ↗		$\Delta^5 f_{-2}$	*1-ша*
x_1	f_1		$\Delta^2 f_0$		$\Delta^4 f_0$		
		Δf_1		$\Delta^3 f_0$			
x_2	f_2		$\Delta^2 f_1$				
		Δf_2					
x_3	f_3						

Таким чином перша формула Гауса здійснює «інтерполяцію вперед» всередині таблиці, друга формула Гауса – «інтерполяцію назад».

5.5. Зворотна інтерполяція

Задача зворотної інтерполяція полягає в тому, щоб за заданими значеннями функції $f(x)$ визначити відповідні значення аргументу x (рис. 5.3). В задачі зворотної інтерполяції слід розрізняти два випадки. Якщо функція $f(x)$ на проміжку наближення є монотонною, тобто

існує взаємно однозначна відповідність між x_i і $f(x_i)$, можна побудувати інтерполяційний поліном $L_n(y)$. Фактично, це означає, що треба "перевернути" таблицю, яка задає функцію,

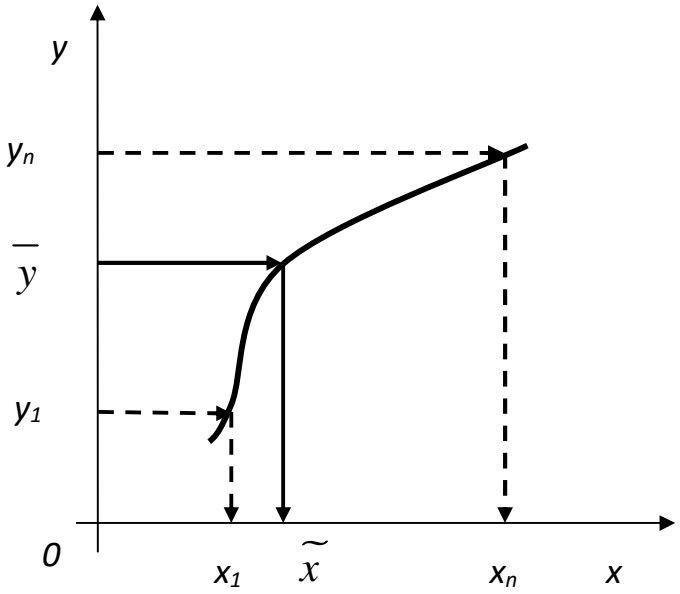


Рис. 5.3. Геометрична інтерпретація зворотної інтерполяції

тобто вважати f_i значеннями незалежної змінної, а x_i - значеннями функції. Тоді, щоб для заданого $\tilde{y} = f(\tilde{x})$ обчислити відповідне $\tilde{x}$ достатньо скористатися формулою $\tilde{x} = L_n(\bar{y})$.

У випадку немонотонної функції слід виписати той чи інший поліном $Z_n(x)$ для функції $f(x)$, а потім з рівняння $Z_n(x) = \bar{y}$ знайти значення $\tilde{x}$, яке і буде наближеним до шуканого $\bar{x}$. Його точність визначається різницею

$$\left|\bar{x} - \tilde{x}\right| \le \frac{M_{n+1}}{(n+1)!\,m_1}\left|m_1(\tilde{x})\right|,$$

де $M_{n+1} \ge \max\left|f^{(n+1)}(x)\right|$, $m_1 \le \min\left|f'(x)\right|$.

Якщо задана точність не досягнута, необхідно підвищити степінь полінома.

Розв'язання рівнянь з одним невідомим за допомогою зворотної інтерполяції. Найпростіший спосіб знаходження коренів рівняння методом зворотної інтерполяції полягає у тому, що для рівняння $f(x) = 0$ на інтервалі відокремлення кореня складають таблицю значень функції $f(x)$, а потім застосовують метод зворотного інтерполювання для $\bar{y} = 0$. Це є можливим оскільки на інтервалі відокремлення кореня перша похідна $f'(x)$ зберігає постійний знак, тобто $f(x)$ є монотонною.

Перша обставина, яку при цьому треба приймати до уваги, полягає в тому, що навіть якщо таблиця значень $f(x)$ складена для рівновіддалених вузлів, в "перевернутій" таблиці вузли стають нерівновіддаленими (за винятком випадку, коли $f(x)$ є лінійною функцією). Отже для виконання зворотної інтерполяції слід застосовувати інтерполяційний поліном для нерівних проміжків.

По-друге, постає задача визначення степеня полінома, необхідного для забезпечення заданої точності обчислення кореня. Для інтерполяційних поліномів, побудованих на розділених різницях, існує практичне правило, за яким похибка інтерполяції не перевищує абсолютної величини останнього доданка, який утримується в запису полінома.

Отже процес уточнення кореня методом зворотної інтерполяції виглядає наступним чином. Обирають початковий степінь полінома в залежності від властивостей $f(x)$. Насправді можна починати з кількості вузлів n = 3, тобто з полінома другого степеня. Далі складають таблицю значень $f(x)$ на проміжку відділення кореня, будують "перевернуту" таблицю розділених різниць, на її основі формують інтерполяційний поліном і якщо останній доданок в його складі за абсолютною величиною не більший ніж задана точність, корінь рівняння обчислюють як значення полінома в точці 0. Інакше, значення n збільшують на одиницю і процес повторюють з самого початку.

Приклад 5.8. Обчислити корінь рівняння $2x^2 + 5x - 8 = 0$, який належить проміжку $[1;1.5]$, методом зворотної інтерполяції з точністю $\varepsilon = 0.001$.

Розв'язок. Виберемо n = 3 та побудуємо таблицю значень функції $f(x) = 2x^2 + 5x - 8.$

x_i	f_i
1.000000	-1.000000
1.250000	1.375000
1.500000	4.000000

Далі складемо "перевернуту" таблицю розділених різниць.

f_i	x_i	Δf_i	$\Delta^2 f_i$
-1.000000	1.000000		
1.375000	1.250000	0.105263	
4.000000	1.500000	0.095238	-0.002005

Скориставшись (5.7) отримаємо інтерполяційний поліном

$$P_2(0) = 1.0 + (0 - (-1))0.105263 + 0 - (-1))(0 - 1.375)(-0.002005)$$

Значення останнього доданка за абсолютною вличиною дорівнює 0.002757 і перевищує задане ε. Отже, збільшуємо n на одиницю і повторюємо процес.

x_i	f_i
1.000000	-1.000000
1.166667	0.555556
1.333333	2.222222
1.500000	4.000000

Тепер маємо

f_i	x_i	Δf_i	$\Delta^2 f_i$	$\Delta^3 f_i$
-1.000000	1.000000			
0.555556	1.166667	0.107143		
2.222222	1.333333	0.100000	-0.002217	
4.000000	1.500000	0.093750	-0.001815	0.000080

$$P_3(0) = 1.0 + (0-(-1))0.107143 + (0-(-1))(0-0.555556)(-0.002217) +$$
$$+(0-(-1))(0-0.555556)(0-2.22222)0.00008$$

Значення останнього доданка за абсолютною величиною дорівнює 0.000099, тобто менше за ε. Таким чином, шукане значення кореня $x = P_3(0) =$ 1.108473.

5.6. Наближення функцій за допомогою інтерполяційних сплайнів

Підвищення степеня інтерполяційного полінома не завжди призводить до зменшення похибки наближення функції. Типовим прикладом цього є так званий ***феномен Рунге*** , який полягає у наступному. Спроби інтерполювати функцію Рунге $f(x) = \frac{1}{1+25x^2}$ на проміжку $[-1,1]$ у рівновіддалених вузлах $x_i = -1 + (i-1)\frac{2}{n}$, $i \in \{1,\ 2,...,\ n+1\}$ за допомогою інтерполяційного полінома $P_n(x)$ призводять до того, що на краях проміжку має місце осциляція полінома, причому тим більша, чим вищий степінь полінома (рис. 5.4).

Подолати цей недолік можна застосовуючи інтерполяційний сплайн. Нехай відрізок $[a,b]$ розбито на N рівних часткових відрізків $[x_i, x_{i+1}]$.

Сплайном називають функцію, яка разом з кількома її похідними є неперервною на відрізку $[a,b]$, а на кожному частковому відрізку $[x_i, x_{i+1}]$, є певним алгебраїчним многочленом.

Максимальний по всіх часткових відрізках степінь цих многочленів називається степенем сплайна, а різниця між степенем сплайна і порядком найвищої безперервної на $[a,b]$ похідної - дефектом сплайна.

Наприклад, неперервна кусково-лінійна функція (ламана) є сплайном першого степеня з дефектом рівним одиниці, оскільки безперервною є тільки сама функція (нульова похідна).

На практиці найбільше поширення одержали сплайни третього степеня, що мають на $[a,b]$ принаймні безперервну першу похідну. Такі сплайни називають ***кубічними*** і позначають $S_3(x)$.

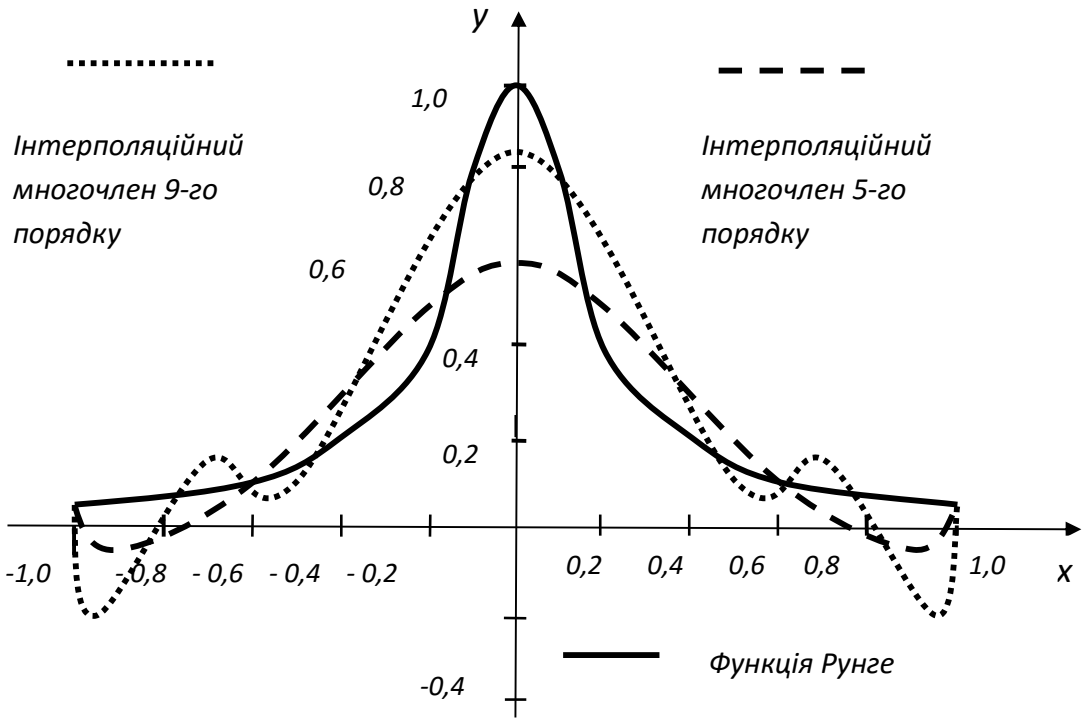


Рис. 5.4. Феномен Рунге

Інтерполяційні кубічні сплайни. Нехай на $[a,b]$ визначено неперервну функцію $f(x)$. Введемо вузли (сітку) $a = x_0 < x_1 < ... < x_{n-1} < x_n = b$, і позначимо $f_i = f(x_i),\ i = \overline{0,\ n}.$

Інтерполяційним кубічним сплайном, що відповідає функції $f(x)$ на заданих вузлах, називають функцію $S(x)$, яка задовольняє наступним умовам:

1) на кожному сегменті $[x_{i-1}, x_i],\ i = \overline{1,\ n}$, функція $S(x)$ є поліномом третього степеня;

2) функція $S(x)$, а також її перша й друга похідні неперервні на $[a,b]$;

3) $S(x_i) = f(x_i),\ i = \overline{0,\ n}$.

Останню умову називають також ***умовою інтерполяції***.

На кожному з відрізків $[x_{i-1}, x_i],\ i = \overline{1,\ n}$ будемо шукати функцію $S(x) = S_i(x)$ у вигляді полінома третього степеня

$$S_i(x) = a_i + b_i(x - x_i) + \frac{c_i}{2}(x - x_i)^2 + \frac{d_i}{6}(x - x_i)^3, \qquad (5.10)$$

де $x_{i-1} \le x \le x_i,\ i = \overline{1,\ n},,\ a_i, b_i, c_i, d_i$ – коефіцієнти, які потрібно визначити.

З'ясуємо зміст введених коефіцієнтів. Маємо

$$S'_i(x) = b_i + c_i(x - x_i) + \frac{d_i}{2}(x - x_i)^2,\ S''_i(x) = c_i + d_i(x - x_i),\ S'''_i(x) = d_i.$$

Тому $a_i = S_i(x_i),\quad b_i = S'_i(x_i),\quad c_i = S''_i(x_i),\quad d_i = S'''_i(x_i).$

З умов інтерполяції $S_i(x_i) = f(x_i),\quad i = \overline{1,\ n},$ отримуємо $a_i = f(x_i),\ i = \overline{1,\ n}.$ Крім того, необхідно додатково визначити $a_0 = f(x_0)$. Далі, вимога неперервності функції $S(x)$ приводить до умов

$$S_i(x) = S_{i+1}(x_i),\ i = \overline{1,\ n-1}.$$

Звідси, беручи до уваги вирази для функцій $S_i(x)$, отримуємо при $i = \overline{0, n-1}$ рівняння

$$a_i = a_{i+1} + b_{i+1}(x_i - x_{i+1}) + \frac{c_{i+1}}{2}(x_i - x_{i+1})^2 + \frac{d_{i+1}}{6}(x_i - x_{i+1})^3.$$

Позначаючи $h_i = x_i - x_{i-1}$, перепишемо ці рівняння у вигляді

$$h_i b_i - \frac{h_i^2}{2}c_i + \frac{h_i^3}{6}d_i = f_i - f_{i-1},\ i = \overline{1,n}. \qquad (5.11)$$

З умов неперервності першої похідної $S'_i(x_i) = S'_{i+1}(x_i),\ i = \overline{1,\ n-1}$ отримуємо рівняння

$$c_i h_i - \frac{d_i}{2}h_i^2 = b_i - b_{i-1},\ i = \overline{2,n}\ , \qquad (5.12)$$

А з умов неперервності другої похідної - рівняння

$$d_i h_i = c_i - c_{i-1}\ \ i = \overline{2,n}. \qquad (5.13)$$

Об'єднання виразів (5.11) – (5.13) приводить до системи з $3n-2$ лінійних рівнянь відносно $3n$ невідомих b_i, c_i, d_i, $(i = \overline{1,n}$). Два рівняння, яких не вистачає, отримують, задаючи тим чи іншим способом граничні умови для $S(x)$.

Припустимо, наприклад, що функція $f(x)$ задовольняє умові $f''(a) = f''(b) = 0$. Тоді природно вимагати, щоб $S''(a) = S''(b) = 0.$ Звідси $S''_1(x_0) = 0,\quad S''_n(x_n) = 0,$ тобто $c_1 - d_1 h_1 = 0,\quad c_n = 0.$ Остаточно для визначення коефіцієнтів c_i маємо систему рівнянь

$$h_i c_{i-1} + 2(h_i + h_{i+1})c_i + h_{i+1}c_{i+1} = 6\left(\frac{f_{i+1} - f_i}{h_{i+1}} - \frac{f_i - f_{i-1}}{h_i}\right),\quad i = \overline{1,n-1},\ c_0 = c_n = 0. \quad (5.14)$$

Внаслідок діагонального переважання система (5.14) має єдиний розв'язок. Оскільки матриця системи є тридіагональною, її розв'язок можна знайти методом прогону. За знайденими значеннями c_i, коефіцієнти d_i і b_i визначаємо за формулами

$$d_i = \frac{c_i - c_{i-1}}{h_i},\ \ b_i = \frac{h_i}{2}c_i - \frac{h_i^2}{6}d_i + \frac{f_i - f_{i-1}}{h_i},\quad i = \overline{1,n}\ .$$

Таким чином, існує єдиний кубічний сплайн, що визначається умовами 1)-3) і граничними умовами $S''(a) = S''(b) = 0$. Визначений в такий спосіб сплайн називають ***натуральним кубічним сплайном***. Зауважимо, що можна розглядати й інші граничні умови.

5.7. Розв'язання систем лінійних алгебраїчних рівнянь методом прогону

Система (5.14) є системою з «тридіагональною матрицею», тобто матрицею вигляду

$$A = \begin{Vmatrix} C_1 & B_1 & & \dots & 0 & 0 & 0 \\ A_2 & C_2 & B_2 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & A_{n-1} & C_{n-1} & B_{n-1} \\ 0 & 0 & 0 & \dots & 0 & A_n & C_n \end{Vmatrix}$$

Окрім обчислення коефіцієнтів кубічного сплайну, системи лінійних рівнянь з такими матрицями зустрічаються при розв'язанні багатьох задач математики й фізики. Найбільш ефективним методом розв'язання таких систем є метод прогону.

Метод прогону для розв'язання систем вигляду $A\vec{X} = \vec{F}$ ґрунтується на припущені, що шукані невідомі пов'язані рекурентним співвідношенням

$$x_i = \alpha_{i+1}x_{i+1} + \beta_{i+1}, \tag{5.15}$$

де $i = \overline{1,n}$.

Використовуючи співвідношення (5.15), виразимо x_{i-1} та x_i через x_{i+1} і підставимо в i-е рівняння:

$$(A_i\alpha_i\alpha_{i+1} + C_i\alpha_{i+1} + B_i)x_{i+1} + A_i\alpha_i\beta_{i+1} + A_i\beta_i + C_i\beta_{i+1} - F_i = 0, \tag{5.16}$$

де F_i – права частина i-го рівняння.

Співвідношення (5.15) та (5.16) будуть виконуватися незалежно від результатів розв'язку, якщо вимагати щоб

$$A_i\alpha_i\alpha_{i+1} + C_i\alpha_{i+1} + B_i = 0,$$
$$A_i\alpha_i\beta_{i+1} + A_i\beta_i + C_i\beta_{i+1} - F_i = 0.$$

Звідси випливає

$$\alpha_{i+1} = \frac{-B_i}{A_i\alpha_i + C_i}, \tag{5.17}$$

$$\beta_{i+1} = \frac{F_i - A_i\beta_i}{A_i\alpha_i + C_i}. \tag{5.18}$$

Із рівняння (5.17) отримаємо $\alpha_1 = \frac{-B_1}{C_1}$, із рівняння (5.18) - $\beta_1 = \frac{F_1}{C_1}$. Після знаходження коефіцієнтів α_i і β_i, скориставшись рівнянням (5.15), отримаємо розв'язок системи, при цьому $x_n = \frac{F_n - A_n\beta_n}{A_n\alpha_n + C_n}$.

Для того, щоб система з тридіагональною матрицею мала єдиний розв'язок необхідно, щоб її матриця відповідала вимозі діагонального переважання.

Матриця A є матрицею ***з діагональним переважанням***, якщо

$$|a_{ii}| \geq \sum_{j=1}^{n} |a_{ij}|, \ j \neq i, i = 1,...,n,$$

причому хоча б одна нерівність є строгою. Якщо всі нерівності строгі, то кажуть, що матриця A є матрицею ***зі строгим діагональним переважанням***.

Контрольні запитання та завдання до розділу 5

1. Що таке інтерполяція?
2. В яких випадках доцільно застосувувати інтерполяцію?
3. Навести формулу полінома Лагранжа.
4. Для функції, заданої таблицею

x_i	2	4	6
f_i	5	3	1

обчислити її значення в точці $x = 2.5$ за допомогою формули Лагранжа.

5. Чим визначається похибка інтерполяції?
6. Як оцінити похибку інтерполяції в точці й на інтервалі?
7. Оцінити похибку наближення функції $f(x) = \sin(x)$ в точці $x = \pi/3$ і на проміжку інтерполювання $[0;\pi/2]$ за допомогою інтерполяційного полінома Лагранжа другого степеня, побудованого в точках $x_0 = 0$, $x_1 = \pi/4$, $x_2 = \pi/2$.
8. Навести визначення скінченних різниць.
9. Навести визначення розділених різниць.
10. Навести формулу інтерполяційного полінома Ньютона.
11. Побудувати інтерполяційний поліном Ньютона для функції, заданої таблицею

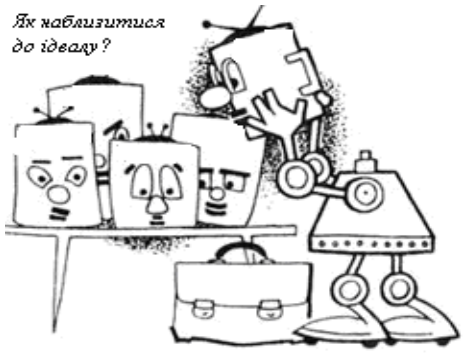
x_i	1	-2	-4
f_i	1	2	4

12. В яких випадках доцільно застосовувати першу й другу інтерполяційні формули Ньютона?
13. Що таке зворотна інтерполяція?
14. Для монотонної функції, заданої таблицею,

x_i	0.000000	0.456932	0.913864	1.370796
f_i	0.000000	0.441197	0.791869	0.980067

методом зворотної інтерполяції обчислити $\tilde{y}$ для $x = 0.5$.

15. Яким чином за допомогою зворотної інтерполяції можна обчислювати корені рівнянь?
16. За допомогою зворотної інтерполяції обчислити корінь рівняння $\ln(x) - 1 = 0$, розміщзений на проміжку $[2.5; 2.8]$ з похибкою не більше за 10^{-3}.
17. Яку функцію називають сплайном ?
18. Як визначають кубічний інтерполяційний сплайн?
19. Які існують способи визначення крайових умов?
20. Який кубічний сплайн називають натуральним?
21. У чому полягає метод прогону?
22. Сформулюйте умови існування та єдиності розв'язку системи з тридіагональною матрицею.



РОЗДІЛ 6. СЕРЕДНЬОКВАДРАТИЧНЕ НАБЛИЖЕННЯ ФУНКЦІЙ

6.1. Постановка задачі середньоквадратичного наближення

Інтерполяція функції полягає в заміні заданої функції $f(x)$ іншою функцією $L_n(x)$ за умови, що функції $f(x)$ і $L_n(x)$ співпадають на заданій послідовності точок. Однак подавати функцію за допомогою інтерполяційного полінома не завжди зручно: зі збільшенням кількості вузлів зростає степінь полінома, що не завжди дозволяє поліпшувати наближене подання функції на заданому відрізку. Крім того, близькість ординат кривих $f(x)$ і $L_n(x)$ на заданому відрізку ще не гарантує близькості на ньому похідних $f'(x)$ і $L_n'(x)$, тобто малої різниці кутових коефіцієнтів дотичних до цих кривих. Тому постає задача такого подання функції $f(x)$ за допомогою функції $g(x)$, яке б характеризувало функцію $f(x)$ на заданому відрізку в цілому, без копіювання її локальних особливостей.

Такі міркування приводять до доцільності середньоквадратичного наближення функції.

У найбільш загальній формі цю задачу можна сформулювати наступним чином.

Для функції $f(x)$, заданої на відрізку $[a,b]$, потрібно підібрати апроксимуючу функцію $g(x)$ таку, щоб значення інтеграла

$$\int_a^b \big(f(x)-g(x)\big)^2\,dx \tag{6.1}$$

було якнайменшим.

Якщо значення інтеграла (6.1) є малим, це означає, що в середньому на більшій частині відрізка $[a,b]$ функції $f(x)$ і $g(x)$ близькі одна до одної, хоча в окремих точках або на дуже малій його частині різниця $f(x)-g(x)$ може бути досить істотною (рис. 6.1).

Таким чином, середньоквадратичне наближення зумовлює згладжування місцевих похибок. Величина

$$\Delta=\sqrt{\Big(\int_a^b (f(x)-g(x))^2\,dx\Big)/(b-a)}$$

називається ***середньоквадратичним відхиленням*** функцій $f(x)$ і $g(x)$ і характеризує похибку наближення функції $f(x)$ за допомогою $g(x)$ як ***середньоквадратичного***. Якщо

функцію задано її значеннями в $(n+1)$ точках $x_0, x_1, x_2, ..., x_n$ (тобто невідома її аналітична форма), то природно замість інтеграла (6.1) розглядати суму вигляду

$$\sum_{i=0}^{n} (f(x_i) - g(x_i))^2 ,$$

а середньоквадратичне відхилення визначати за формулою

$$\Delta_n = \sqrt{(\sum_{i=0}^{n} (f(x_i) - g(x_i))^2)/(n+1)}.$$

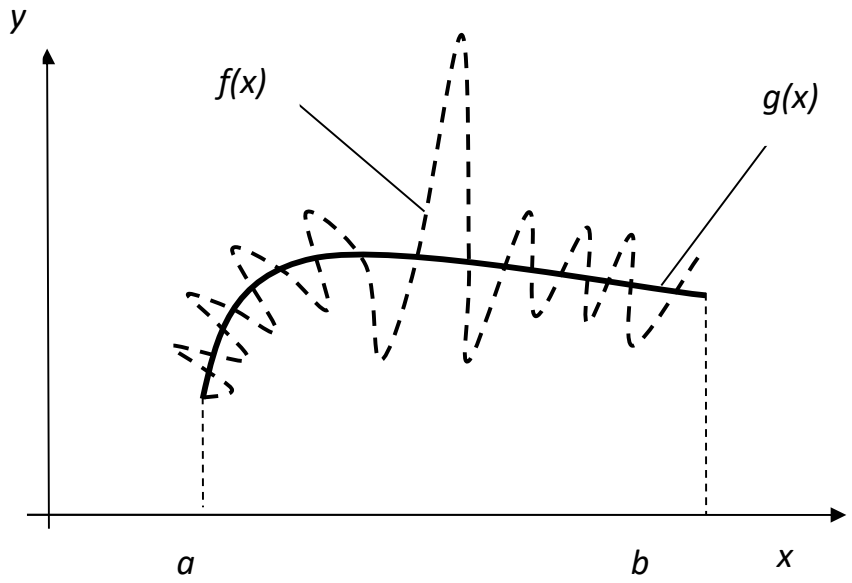


Рис. 6.1. Середньоквадратичне наближення функції $f(x)$ за допомогою апроксимуючої функції $g(x)$.

Для розв'язання практичних задач як апроксимуючі функції $g(x)$ найчастіше вибирають степеневий або тригонометричний поліном. Тоді ***задача середньоквадратичного наближення*** функцій формулюється так.

Нехай $\varphi_0(x), \varphi_1(x), ..., \varphi_m(x)$ – задана на відрізку $[a,b]$ система лінійно незалежних і неперервних функцій. Узагальненим поліномом $P_m(x)$ m-го степеня за системою (базисом) $\{\varphi_i(x)\}$ будемо називати вираз

$$P_m(x) = a_0\varphi_0(x) + a_1\varphi_1(x) + ... + a_m\varphi_m(x), \qquad (6.2)$$

де $a_0, a_1, ..., a_m$ – деякі константи.

Потрібно підібрати такі значення $a_0, a_1, ..., a_m$ в узагальненому поліномі (6.2), щоб значення інтеграла

$$I(a_0, a_1, ..., a_m) = \int_a^b (f(x) - P_m(x))^2 dx$$

було якнайменшим.

6.2. Нормальна система

Для визначення $a_0, a_1, ..., a_m$ прирівняємо до нуля частинні похідні інтеграла $I(a_0, a_1, ..., a_m)$ за всіма a_k. Тоді маємо систему рівнянь, яка називають ***нормальною системою***

$$\sum_{i=0}^{m} a_i(\varphi_i, \varphi_k) = (f, \varphi_k),\ k = 0,1,..,m,$$

де $(\varphi_i, \varphi_k) = \int_a^b \varphi_i(x)\varphi_k(x)dx$, а $(f, \varphi_k) = \int_a^b f(x)\varphi_k dx$.

Таким чином, для аналітично заданої функції нормальна система має вигляд

$$\begin{cases} a_0\int_a^b \varphi_0^2(x)dx + a_1\int_a^b \varphi_1(x)\varphi_0(x)dx + ... + a_m\int_a^b \varphi_m(x)\varphi_0(x)dx = \int_a^b f(x)\varphi_0(x)dx, \\ a_0\int_a^b \varphi_0(x)\varphi_1(x)dx + a_1\int_a^b \varphi_1^2(x)dx + ... + a_m\int_a^b \varphi_m(x)\varphi_1(x) = \int_a^b f(x)\varphi_1(x)dx, \\ ..... \\ a_0\int_a^b \varphi_0(x)\varphi_m(x) + a_1\int_a^b \varphi_1(x)\varphi_m(x) + ... + a_m\int_a^b \varphi_m^2(x)dx = \int_a^b f(x)\varphi_m(x)dx. \end{cases}$$

Доведено, що нормальна система завжди має розв'язок, і притому єдиний. Цей розв'язок і є шуканими значеннями $a_0, a_1, ..., a_m$.

Виберемо як систему базисних функцій $\varphi_0(x), \varphi_1(x), ..., \varphi_m(x)$ степеневий ряд $1, x, x^2, ..., x^m$ і побудуємо за методом найменших квадратів многочлен другого степеня для наближеного подання функції $f(x) = |x|$ на відрізку [-1;1]. Поліном (6.2) у цьому випадку набуває вигляду

$$P_2(x) = a_0 + a_1 x + a_2 x^2.$$

Визначимо коефіцієнти a_0, a_1, a_2.

$$\begin{aligned} a_0\int_{-1}^{1} dx + a_1\int_{-1}^{1} xdx + a_2\int_{-1}^{1} x^2dx &= \int_{-1}^{0}(-x)dx + \int_{0}^{1} xdx, \\ a_0\int_{-1}^{1} xdx + a_1\int_{-1}^{1} x^2dx + a_2\int_{-1}^{1} x^3dx &= \int_{-1}^{0}(-x)xdx + \int_{0}^{1} x^2dx, \\ a_0\int_{-1}^{1} x^2dx + a_1\int_{-1}^{1} x^3dx + a_2\int_{-1}^{1} x^4dx &= \int_{-1}^{0}(-x)x^2dx + \int_{0}^{1} x^3dx. \end{aligned}$$

Після інтегрування маємо

$$\begin{cases} 2\cdot a_0 + 0\cdot a_1 + \frac{2}{3}\cdot a_2 = 1, \\ 0\cdot a_0 + \frac{2}{3}\cdot a_1 + 0\cdot a_2 = 0, \\ \frac{2}{3}\cdot a_0 + 0\cdot a_1 + \frac{2}{5} a_2 = \frac{1}{2}. \end{cases}$$

Звідси $a_0 = 0{,}1875$, $a_1 = 0$, $a_2 = 0{,}9375$. Отже, шуканим поліномом є $P_2(x) = 0{,}1875 + 0{,}9275x^2$

Розглянемо випадок, коли функцію $f(x)$ задан таблицею. Тоді кажуть, що функцію ***задано в точках***, на відміну від попереднього випадку, де функцію було ***задано на інтервалі***. Побудуємо поліном

$$P_m(x) = a_0 x^m + a_1 x^{m-1} + ... + a_{m-1}x + a_m$$

таким чином, щоб сума

$$\Sigma = \sum_{i=0}^{m} (P_m(x_i) - f_i)^2$$

була найменшою. Для виконання цієї умови досить, щоб

$$\frac{\partial \Sigma}{\partial a_0} = 0, \frac{\partial \Sigma}{\partial a_1} = 0, ..., \frac{\partial \Sigma}{\partial a_m} = 0,$$

що дає нормальну систему з $(m+1)$ рівняннями відносно $(m+1)$ невідомих. Нехай, наприклад, маємо функцію, задану за допомогою таблиці

x_i	1	2	3	4
f_i	0	1	3	5

Побудуємо поліном першого степеня з використанням степеневого базису $P_1(x) = a_0 x + a_1$.

Маємо нормальну систему

$$\frac{\partial \sum_i (a_0 x_i + a_1 - f_i)^2}{\partial a_0} = 0,$$

$$\frac{\partial \sum_i (a_0 x_i + a_1 - f_i)^2}{\partial a_1} = 0.$$

Виконаємо диференціювання, тоді отримаємо

$$2\sum_i (a_0 x_i + a_1 - f_i) x_i = 0,$$

$$2\sum_i (a_0 x_i + a_1 - f_i) = 0.$$

Або

$$a_0\sum_i x_i^2 + a_1\sum_i x_i - \sum_i f_i x_i = 0,$$
$$a_0\sum_i x_i + 4a_1 - \sum_i f_i = 0.$$

Остаточно маємо

$$30a_0 + 10a_1 = 31,$$
$$10a_0 + 4a_1 = 9.$$

Звідси $a_0 = 1.7; a_1 = -2$. Отже $P_1(x) = 1.7x - 2.$

6.3. Ортогональні базиси.

Крім степеневої системи функцій для побудови узагальненого многочлена використовують також систему многочленів Лежандра та систему многочленів Чебишова першого роду.

Система многочленів Лежандра задається формулою

$$P_n(x) = \frac{1}{2^n n!}\frac{d^n}{dx^n}(x^2-1)^n,\ n = 0,1,2....$$

Перші два многочлени Лежандра ($n = 0$ і $n = 1$) дорівнюють відповідно 1 та x. Решта многочленів може бути обчислена за рекурентним співвідношенням

$$P_{n+1}(x) = xP_n(x) + \frac{x^2-1}{n+1}\frac{dP_n}{dx} = \frac{2n+1}{n+1}xP_n(x) - \frac{n}{n+1}P_{n-1}(x).$$

Многочлени Лежандра утворюють ортогональну систему многочленів на відрізку [-1;1] , тобто

$$\int_{-1}^{1} P_n(x)P_m(x)dx = \begin{cases} 0, m \neq n, \\ 2/(2n+1), m = n. \end{cases}$$

Система многочленів Чебишова першого роду визначається таким чином

$$T_n(x) = \cos(n \cdot \text{arc}\cos x),\ n = 0,1,2....$$

Перші два многочлени Чебишова дорівнюють відповідно 1 та x. Решту многочленів можна визначити за допомогою рекурентної формули

$$T_{n+1}(x) = 2x \cdot T_n(x) - T_{n-1}(x).$$

Для многочленів Чебишова першого роду виконуються рівності

$$\int_{-1}^{1} \frac{T_n^2(x)}{\sqrt{1-x^2}}dx = \begin{cases} \pi, n = 0, \\ \pi/2, n \neq 0, \end{cases} \quad \text{та} \quad \int_{-1}^{1} \frac{T_n(x)T_m(x)}{\sqrt{1-x^2}}dx = 0,\ n \neq m,$$

тобто вони є ортогональними на відрізку [-1;1] з вагою

$$t(x) = \frac{1}{\sqrt{1-x^2}}.$$

Наведені системи є ***повними***, тобто $P_m(x) \to f(x)$ при $m \to \infty$ як середньоквадратичне, або

$$\int_a^b t(x)(f(x) - P_m(x))^2 dx \to 0, \ (m \to \infty) \ .$$

Неважко помітити, що у випадку ортогональної системи функцій $\{\varphi_k(x)\}$ на відрізку $[a,b]$ всі $\int_a^b \varphi_k(x)\varphi_i(x)dx = 0,$ при $k \neq i$ і, як наслідок, нормальна система стає діагональною, що значно спрощує її розв'язання, а саме

$$a_k = \frac{(f,\varphi_k)}{(\varphi_k,\varphi_k)} = \frac{\int_a^b f(x)\varphi_k(x)dx}{\int_a^b \varphi_k^2(x)dx}, \ k = 0,1,...,m, \qquad (6.3)$$

Коефіцієнти a_k, обчислені за формулою (6.3), називаються ***коефіцієнтами Фур'є*** функції $f(x)$ щодо ортогональної системи функцій $\{\varphi_k(x)\}$.

Многочлени Лежандра та Чебишова ортогональні лише на проміжку [-1;1], тому, якщо середньоквадратичний многочлен будують на проміжку, відмінному від [-1;1], нормальна система не матиме діагональної матриці при використанні цих могочленів.

Розв'язати цю проблему можна таким чином. За допомогою лінійної заміни $x = \frac{b+a}{2} + t\frac{b-a}{2}$ перейдемо від змінної x до змінної t. При цьому, коли змінна x пробігає значення від a до b, t відповідно змінюється від -1 до 1. Тепер можна побудувати узагальнений многочлен $P_m(t)$ для функції $g(t)$ на проміжку [-1;1], наприклад, за системою многочленів Лежандра, скориставшись нормальною системою з діагональною матрицею. Потім, виконуючи в отриманому многочлені зворотну заміну t на x, отримаємо $P_m(x)$.

Приклад 6.1.. Апроксимувати у сенсі середньоквадратичного функцію $f(x) = |x|$ на відрізку [-1;2] многочленом $P_2(x)$ в базисі многочленів Лежандра.

Розв'язок. Щоб отримати діагональну нормальну систему вводимо заміну $x = 1,5t + 0,5$, де $t \in [-1;1]$, тоді $g(t) \equiv |1,5t + 0,5|$, а

$$P_2(t) = a_0L_0(t) + a_1L_1(t) + a_2L_2(t) = a_0 + a_1t + a_2(3t^2 -)/2.$$

Коефіцієнти a_i знайдемо за формулою (6.3)

$$a_0 = 5/6, \ a_1 = 13/18, \ a_2 = 20/27,$$

$$P_2(t) = 5/6 + 13/18t + 20/21(3t^2 -)/2 .$$

Повертаючись до змінної x отримаємо

$$P_2(x) = 0,34 + 0,02x + 0,49x^2.$$

Нехай в деяких частинах відрізка наближення функції $f(x)$ за методом найменших квадратів з певних міркувань необхідно отримати більш точне, порівняно з іншими частинами його наближення. Це можна зробити наступним чином: інтеграл I, визначений рівністю (6.1), слід замінити інтегралом більш загального вигляду

$$I = \int_a^b t(x)(f(x) - \varphi(x))^2 dx,$$

де $t(x)$ – певним чином підібрана невід'ємна функція, яку називають вагою. При цьому значення $t(x)$ у заданій точці x має бути тим більшим порівняно зі значеннями в інших точках, відмінних від x, чим більша точність наближення потрібна в точці x.

6.4. Наближення періодичних функцій

Наближення періодичних функцій, визначених на всій осі $\tilde{o}$, найбільш раціонально виконувати за допомогою тригонометричних поліномів.

Нехай задано неперервну функцію $f(x)$, яка має період 2π. Будемо будувати апроксимуючий поліном у вигляді

$$T_m(x) = \sum_{k=0}^{m} (a_k \cos kx + b_k \sin kx).$$

Відомо, що тригонометрична система $1, \sin x, \cos x, \sin 2x, \cos 2x$ ортогональна з вагою $t(x) = 1$ на будь-якому проміжку довжиною 2π. Коефіцієнти a_k, b_k обчислюють за формулами

$$a_k = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\cos kx dx,\ b_0 = 0,\ b_k = \frac{1}{\pi}\int_{-\pi}^{\pi} f(x)\sin kx dx.$$

Це відомі з аналізу коефіцієнти Фур'є для функції $f(x)$, причому для парної функції $f(x)$, $b_k = 0$, а для непарної – $a_k = 0$.

Приклад 6.2. За допомогою тригонометричного полінома дев'ятого степеня отримати наближення періодичної функції (рис. 6.2), аналітично заданої як

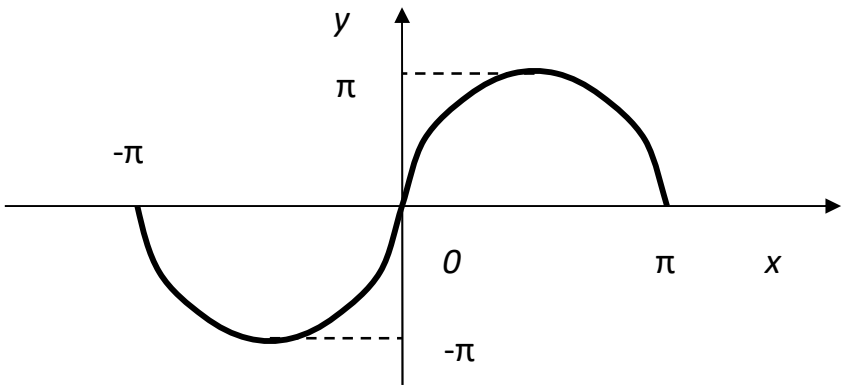


Рис.6.2. Графік періодичної функції до прикладу.

$$f(x)=\begin{cases} x(\pi-x), 0\le x\le \pi, \\ x(\pi+x), -\pi\le x\le 0. \end{cases}$$

Розв'язок. Внаслідок непарності заданої функції a_k=0, k=0,1,2,…. Тоді

$$b_k=\frac{2}{\pi}\int_0^{\pi} x(\pi-x)\sin kx dx=\frac{4}{\pi k^3}[1-(-1)^k].$$

Відповідно $b_{2m}=0$, $b_{2m-1}=\frac{8}{\pi(2m-1)^3}$, а шуканий тригонометричний поліном можна записати у вигляді

$$T_9(x)=\frac{8}{\pi}[\sin x+\frac{\sin 3x}{27}+\frac{\sin 5x}{125}+\frac{\sin 7x}{343}+\frac{\sin 9x}{729}].$$

6.5. Контрольні запитання та завдання до розділу 6

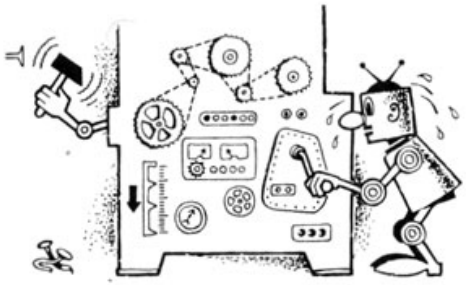
1. У чому полягає принципова відмінність інтерполяції та середньоквадратичного наближення?
2. Як формулюють задачу середньоквадратичного наближення?
3. Яким чином оцінюють якість середньоквадратичного наближення?
4. Яким вимогам повинна відповідати система базисних функцій?
5. Яким чином будують нормальну систему у випадку, якщо функцію задана на інтервалі?
6. Яким чином будують нормальну систему у випадку, якщо функцію задано в точках?
7. Побудуйте $P_2(x)$ в степеневому базисі для функції, заданої таблицею

x_i	1	2	3
f_i	1	4	9

8. Побудуйте $P_2(x)$ в базисі многочленів Лежандра для функції, заданої таблицею

x_i	1	2	3
f_i	1	4	9

9. Що таке ортогональна система функцій і які переваги вона забезпечує при використанні її як базису?
10. Яким чином можна зберігати переваги ортогонального базису у разі, якщо інтервал ортогональності не збігається з інтервалом наближення функції?
11. У чому полягають особливості середньоквадратичного наближення періодичних функцій?



РОЗДІЛ 7. ЧИСЕЛЬНЕ ДИФЕРЕНЦІЮВАННЯ ТА ІНТЕГРУВАННЯ

7.1.Розв'язання задачі чисельного диференціювання

На практиці часто виникає необхідність диференціювати функцій, задані таблицями (наприклад, при визначенні швидкості, прискорення, тощо). Іноді таке диференціювання виявляється корисним і для функцій з досить складним аналітичним поданням.

Інтерполяційні формули чисельного диференціювання ґрунтуються на заміні функції $f(x)$ деяким інтерполяційним поліномом $P_n(x)$ при цьому вважають, що

$$f^{(k)}(x) \approx P_n^{(k)}(x), \quad k = 0, 1, 2, \dots, n \geq k, \tag{7.1}$$

тобто k –та похідна $f(x)$ приблизно дорівнює k–й похідній $P_n(x)$, а похибка її обчислення приблизно дорівнює k–й похідній залишкового члена або похибки інтерполяції $R_n^{(k)}(x)$. Однак з малості залишкового члена інтерполяції зовсім не випливає малість його похідної. Взагалі наближене диференціювання є операцією менш точною, ніж інтерполяція. Близькість ординат кривих $f(x)$ і $P_n(x)$ на відрізку $[a;b]$ ще не гарантує близькості на цьому відрізку похідних $f'(x)$ і $P_n'(x)$, тобто малої різниці кутових коефіцієнтів дотичних до цих кривих.

Розглянемо, наприклад, наближену формулу для обчислення $f'(x_1)$ за значеннями функції у вузлах x_0, x_1, x_2 та оцінимо її похибку. Скориставшись поліномом Лагранжа, маємо

$$f(x) = f_0 \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} + f_1 \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} +$$
$$+ f_2 \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} + \frac{f^{III}(\xi)}{3!}(x-x_0)(x-x_1)(x-x_2).$$

Звідси отримаємо

$$f'(x) = P_2'(x) = f_0 \frac{x_1 - x_2}{(x_0 - x_1)(x_0 - x_2)} + f_1 \frac{2x_1 - x_0 - x_2}{(x_1 - x_0)(x_1 - x_2)} + f_2 \frac{x_1 - x_2}{(x_2 - x_0)(x_2 - x_1)}.$$

Похибка цієї формули дорівнює $-\frac{f'''(\xi)}{6}(x_1 - x_0)(x_1 - x_2)$. Якщо функція задана для рівновіддалених вузлів, формула спрощується:

$$f'(x_1) = \frac{f_2 - f_0}{2h},$$

а її похибка дорівнює $-\frac{f'''(\xi)}{3}h^2$.

Більш зручними для практичного використання є формули зі скінченними різницями. Візьмемо, наприклад, інтерполяційний поліном Ньютона

$$f(x) = f_0 + q\Delta f_0 + \frac{q(q-1)}{2!}\Delta^2 f_0 + \frac{q(q-1)(q-2)}{3!}\Delta^3 f_0 + \dots \quad ,$$

де $q = \frac{x - x_0}{h}$; $h = x_{i+1} - x_i$. Перемножуючи поліноми, маємо

$$f(q) = f_0 + q\Delta f_0 + \frac{q^2 - q}{2}\Delta^2 f_0 + \frac{q^3 - 3q^2 + 2q}{6}\Delta^3 f_0 + \dots$$

Очевидно, що

$$\frac{df}{dx} = \frac{df}{dq}\frac{dq}{dx} = \frac{1}{h}\frac{df}{dq},$$

тому

$$f'(x) = \frac{1}{h}(\Delta f_0 + \frac{2q-1}{2}\Delta^2 f_0 + \frac{3q^2 - 6q + 2}{6}\Delta^3 f_0 + \dots \qquad (7.2)$$

Аналогічно, оскільки

$$f''(x) = \frac{d(f'(x))}{dx} = \frac{d(f'(x))}{dq} \cdot \frac{dq}{dx},$$

то

$$f''(x) = \frac{1}{h^2}(\Delta^2 f_0 + (q-1)\Delta^3 f_0 + ...). \tag{7.3}$$

Відзначимо, що оскільки похибка різниць швидко збільшується з ростом їх порядку, не має сенсу утримувати в формулах (7.2) та (7.3) багато доданків.

Приклад 7.1. У точці $x = 3.217$ обчислити першу похідну функції, заданої таблицею

x_i	f_i	Δf_0	$\Delta^2 f_0$	$\Delta^3 f_0$
3,217	4,72343			
3,218	4,72430	87×10^{-4}		
3,219	4,72516	86×10^{-4}	-1×10^{-5}	
3,220	4,72603	87×10^{-4}	1×10^{-5}	2×10^{-6}

Розв'язок. За формулою (7.3) знаходимо

$$f'(x) = \frac{1}{0.001}(0.00087 - 0.5\cdot 0.00001 + \frac{0.00002}{3} + ...).$$

Для визначення довжини ряду користуються практичним правилом: грубо похибка дорівнює абсолютній величині першого відкинутого доданка. Якщо утримувати один доданок, то маємо $f'(3.217) = 0.87$ з похибкою 0.005. Якщо утримувати два члени, то $f'(3.217) = 0.865$, а похибка дорівнюватимемо 0.015, отже перша оцінка точніше.

Практично за формулами чисельного диференціювання можна визначити першу й другу похідні, а третю й четверту - лише задовільно. Більше високі похідні рідко вдається обчислити із заданою точністю. Кращі результати можуть бути отримані диференціюванням попередньо згладжених функцій (наприклад, методом найменших квадратів).

7.2. Загальний підхід до задачі чисельного інтегрування

Обчислення визначеного інтеграла за формулою Ньютона-Лейбніца

$$\int_a^b f(x)dx = F(b) - F(a), \qquad F'(x) \equiv f(x), \tag{7.4}$$

може бути надто складним або взагалі неможливим через те, що первісна $F(x)$ не може бути виражена через елементарні функції, або такий вираз є занадто складним. У цих випадках доводиться користуватися чисельним інтегруванням. Нарешті, функція може бути табличною і тоді застосування формули (7.4) також є неможливим. Отже, в загальному випадку, задача чисельного інтегрування функції полягає в обчисленні значення визначеного інтеграла на підставі ряду значень підінтегральної функції.

Обчислення однократного інтегралу називають ***механічною квадратурою***, а подвійного - ***механічною кубатурою***. Відповідні формули називають ***квадратурними*** або ***кубатурними***.

Найпростіші квадратурні формули можуть бути отримані з наочних міркувань. Нехай необхідно обчислити інтеграл

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

Якщо $f(x) \approx const$, то можна вважати $I = (b-a)f(c)$, де c – довільна точка відрізка $[a;b]$. Цілком природно як значення c вибрати середину відрізка $[a;b]$, тоді для обчислення інтеграла маємо ***формулу прямокутників***

$$I = (b-a)f(\frac{a+b}{2}).$$

Якщо функція $f(x)$ близька до лінійної (рис. 7.1), природно замінити інтеграл площею

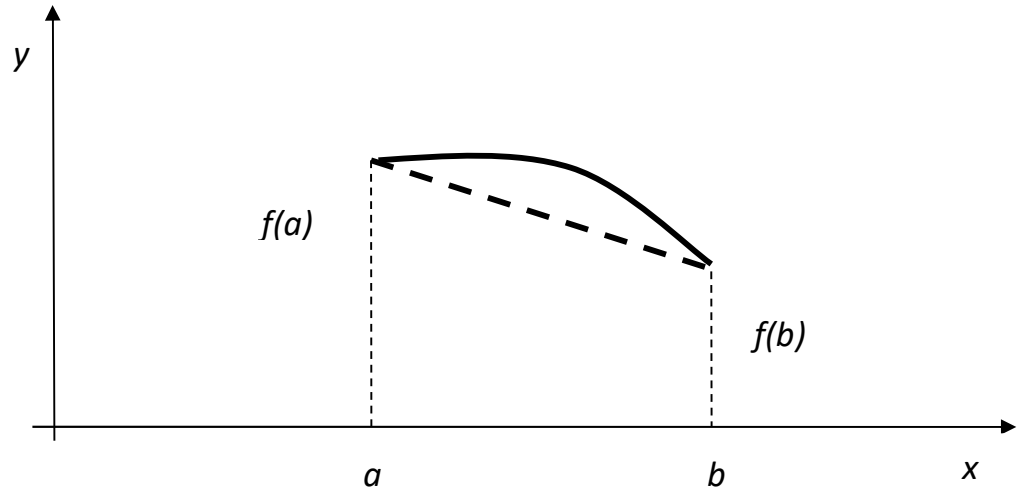


Рис. 7.1. Обчислення інтеграла за формулою трапецій

трапеції висотою $(b-a)$ і сторонами $f(a)$ і $f(b)$. Тоді маємо ***формулу трапецій***

$$I = (b-a)\frac{f(a)+f(b)}{2}.$$

Загальний прийом механічної квадратури полягає в тому, що функцію $f(x)$ на відрізку $[a;b]$ заміняють інтерполяційною функцією $\varphi(x)$ і вважають, що

$$\int_a^b f(x)dx \approx \int_a^b \varphi(x)dx. \qquad (7.5)$$

Якщо $f(x)$ задано аналітично, треба розглянути питання про похибки при використанні формули (7.5). Для цього застосуємо замість $\varphi(x)$ поліном Лагранжа у дещо зміненому вигляді:

$$L_n(x) = \sum_{i=0}^{n} \frac{\Pi_{n+1}(x)}{(x-x_i)\Pi'_{n+1}(x_i)} f_i,$$

де

$$\Pi_{n+1}(x) = (x-x_0)(x-x_1)...(x-x_n). \qquad (7.6)$$

Шляхом диференціювання виразу (7.6) маємо

$$\Pi'_{n+1}(x) = \sum_{j=0}^{n} (x-x_0)(x-x_1)...(x-x_{j-1})(x-x_{j+1})...(x-x_n).$$

Виберемо $x = x_i$, $i = 0,1,...,n$, тоді

$$\Pi'_{n+1}(x_i) = (x_i-x_0)(x_i-x_1)...(x_i-x_{i-1})(x_i-x_{i+1})...(x_i-x_n),$$

$$L_n(x) = \sum_{i=0}^{n} \frac{\Pi_{n+1}(x)}{(x-x_i)\Pi'_{n+1}(x_i)} y_i.$$

Нехай для функції $f(x)$, заданій в $(n+1)$ точці $x_0, x_1, ..., x_n$ на проміжку $[a,b]$

$$L_n(x_i) = f_i, \ (i = 0,1,...,n).$$

Підставляючи $L_n(x)$ замість $\varphi(x)$ в (7.3) отримаємо

$$\int_a^b f(x)dx = \int_a^b L_n(x)dx + R_n[f], \qquad (7.7)$$

де $R_n[f]$ – похибка (залишковий член) квадратурної формули (7.7). Використовуючи формулу (7.5), отримуємо наближену квадратурну формулу

$$\int_a^b f(x)dx \approx \sum_{i=0}^{n} A_i f_i, \qquad (7.8)$$

де

$$A_i = \int_a^b \frac{\Pi_{n+1}(x)}{(x-x_i)\Pi'_{n+1}(x_i)} dx, \ (i = 0,1,...,n). \qquad (7.9)$$

Таким чином, інтеграл замінюють приблизною сумою, причому значення A_i не залежать від функції $f(x)$ і їх можна використовувати для будь-якої підінтегральної функції $f(x)$.

Будемо вважати, що інтерполяційний поліном побудовано для рівновіддалених вузлів $x_k = x_0 + kh$, де $h = (b-a)/n$. Нехай $q = (x - x_0)/h$ і $q^{[n+1]} = q(q-1)...(q-n)$, тоді

$$L_n(x) = \sum_{i=0}^{n} \frac{(-1)^{n-i}}{i!(n-i)!} \cdot \frac{q^{[n+1]}}{q-i} f_i \cdot .$$

Підставляючи останній вираз в (7.9) остаточно маємо

$$A_i = h \frac{(-1)^{n-i}}{i!(n-i)!} \int_0^n \frac{q^{[n+1]}}{q-i} dq \,, \; (i = 0,1,2,..,n).$$

Оскільки $h = (b-a)/n$ і $dq = dx/h$, виберемо $A_i = (b-a)H_i$, де

$$H_i = h \frac{(-1)^{n-i}}{i!(n-i)!} \int_0^n \frac{q^{[n+1]}}{q-i} dq, \tag{7.10}$$

H_i – сталі, які називаються ***коефіцієнтами Котеса***.

7.3. Інтерполяційні квадратурні формули чисельного інтегрування (*Формули Ньютона-Котеса*)

Нехай для функції $y = f(x)$ потрібно обчислити інтеграл

$$I = \int_a^b y(x)dx.$$

Розіб'ємо інтервал $[a,b]$ на рівні проміжки з кроком $h = (b-a)/n$, тобто маємо рівновіддалені вузли $x_0, x_1, ..., x_n$, де $x_k = x_0 + kh$. Для обчислення наближеного значення інтеграла скористуємося формулою (7.7). Якщо при обчисленні $A_i = (b-a)H_i$, $n = 1$, то відповідно до (7.10) маємо коефіцієнти Котеса

$$H_0 = -\int_0^1 \frac{q(q-1)}{q} dq = \frac{1}{2}, \; H_1 = \int_0^1 q dq = \frac{1}{2},$$

і, повертаючись до (7.7) маємо

$$\int_{x_0}^{x_1} y dx \approx \frac{h}{2}(y_0 + y_1). \tag{7.11}$$

Це так звана ***квадратурна формула трапецій***. Її залишковий член R_1 (помилка обчислення інтеграла) дорівнює

$$R_1 = \int_{x_0}^{x_1} y dx - \frac{h}{2}(y_0 + y_1) = -\frac{h^3}{12} y''(\xi),$$

де $\xi \in (x_0, x_1)$.

Звідси, якщо $y'' > 0$, то формула для R_1 дає значення інтеграла ***з надлишком***, якщо $y'' < 0$ – ***з недостачею.***

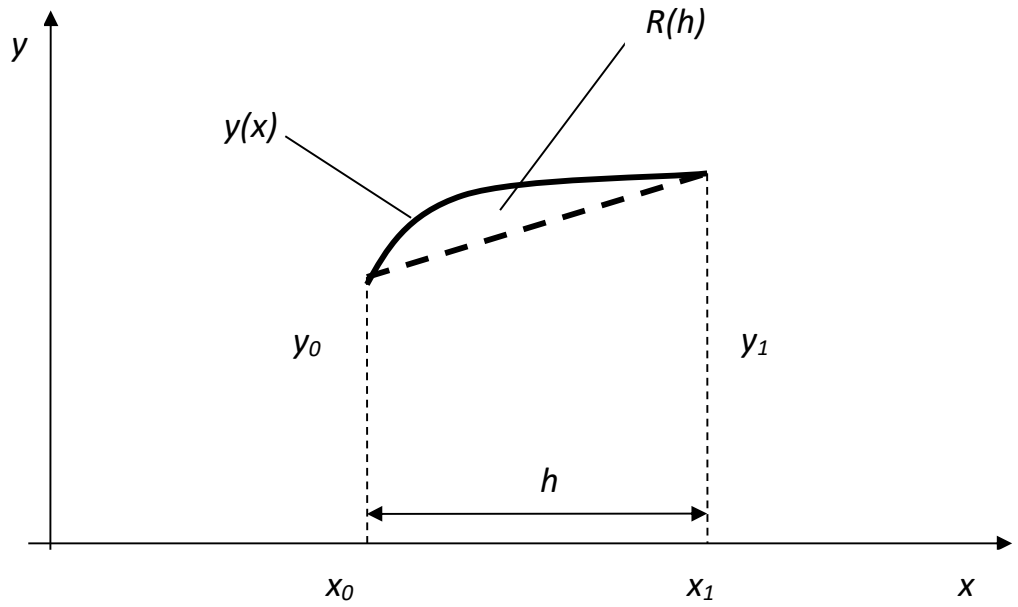


Рис. 7.2. Геометрична інтерпретація квадратурної формули трапецій

Для обчислювальної практики найважливішим є граничне значення абсолютної величини залишкового члена, а саме:

$$\left|R_1\right| \le \frac{M_2 (b-a)^3}{12},$$

де $M_2 = \max(|\, y''(x)\,|)$ при $x \in [a;b]$. Для $n = 2$ маємо коефіцієнти Котеса

$$H_0 = \frac{1}{2}\frac{1}{2}\int_0^2 (q-1)(q-2)dq = \frac{1}{4}(\frac{8}{3} - 6 + 4) = \frac{1}{6},$$

$$H_1 = -\frac{1}{2}\frac{1}{1}\int_0^2 q(q-2)dq = \frac{2}{3},$$

$$H_2 = \frac{1}{2}\frac{1}{2}\int_0^2 q(q-1)dq = \frac{1}{6}.$$

Оскільки тепер $x_2 - x_0 = 2h$, маємо

$$\int_{x_0}^{x_2} ydx \approx \frac{h}{3}(y_0 + 4y_1 + y_2). \qquad (7.12)$$

Це ***квадратурна формула Сімпсона***, геометрична інтерпретація якої приведена на рис.7.3, а її залишковим членом є

$$R_2 = -\frac{h^5}{90} y^{IV}(\xi),$$

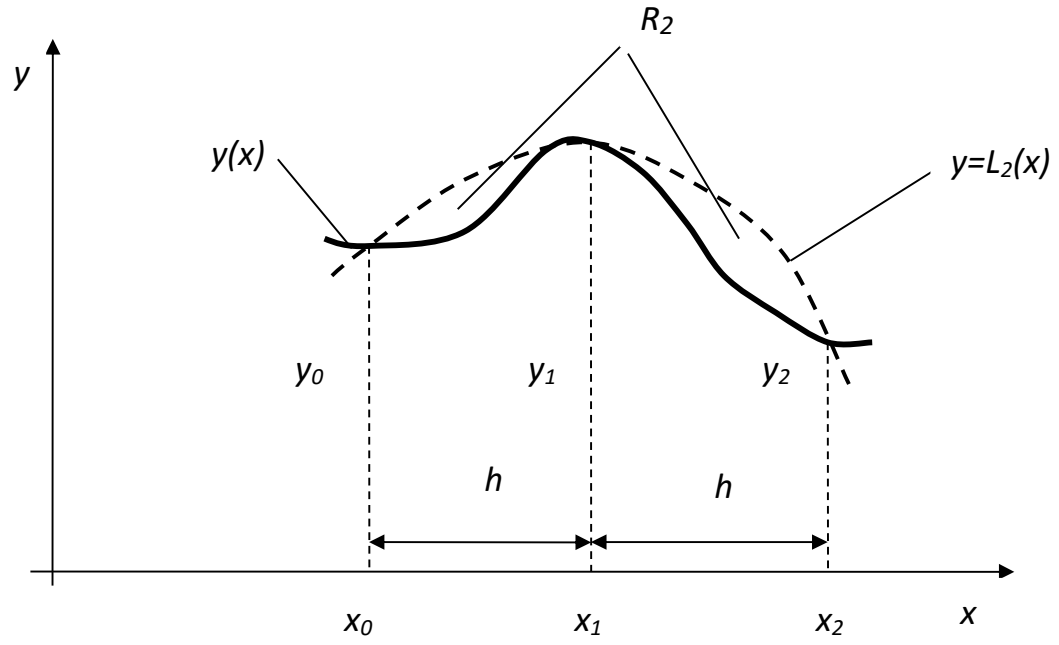


Рис. 7.3. Геометрична інтерпретація квадратурної формули Сімпсона

де $\xi \in (x_0, x_2)$. Відповідно, граничне значення залишкового член дорівнює

$$|R_2| \le \frac{M_4(b-a)}{180},$$

де $M_4 = \max(|y^{IV}(x)|)$ при $x \in [a;b]$.

Розглянуті ***інтерполяційні квадратурні формули*** на практиці мають обмежене використання. Сподіватися, що залишковий член відповідної формули буде достатньо малим, щоб забезпечити задану похибку обчислення інтеграла, можна лише тоді, коли відрізок [*a,b*] і похідна, що входить у залишковий член, дуже малі. Тому забезпечити задовільну похибку інтегрування на більш-менш великих інтервалах [*a,b*] неможливо. На перший погляд здається, що труднощів можна уникнути збільшенням значення *n.* Але коефіцієнти Котеса за великої кількості вузлів обчислювати складно і, крім того, в процесі розрахунків до методичної похибки R_n додається значна обчислювальна похибка. Цих недоліків позбавлені так звані ***узагальнені*** або ***неінтерполяційні формули*** чисельного інтегрування.

7.4. Узагальнені формули чисельного інтегрування

Ідея побудови цих формул ґрунтується на тому, що інтервал інтегрування розбивають на досить велику кількість малих проміжків, до кожного з яких застосовують інтерполяційну квадратурну формулу з малим h. При цьому отримують формули простої структури, точність яких може бути досить високою.

Узагальнена формула трапецій. Розіб'ємо інтервал інтегрування [*a*, *b*] на *n* рівних проміжків: $[x_0, x_1]$, $[x_1, x_2]$, ... ,$[x_{n-1}, x_n]$ і до кожного з них застосуємо формулу (7.11). Тоді матимемо

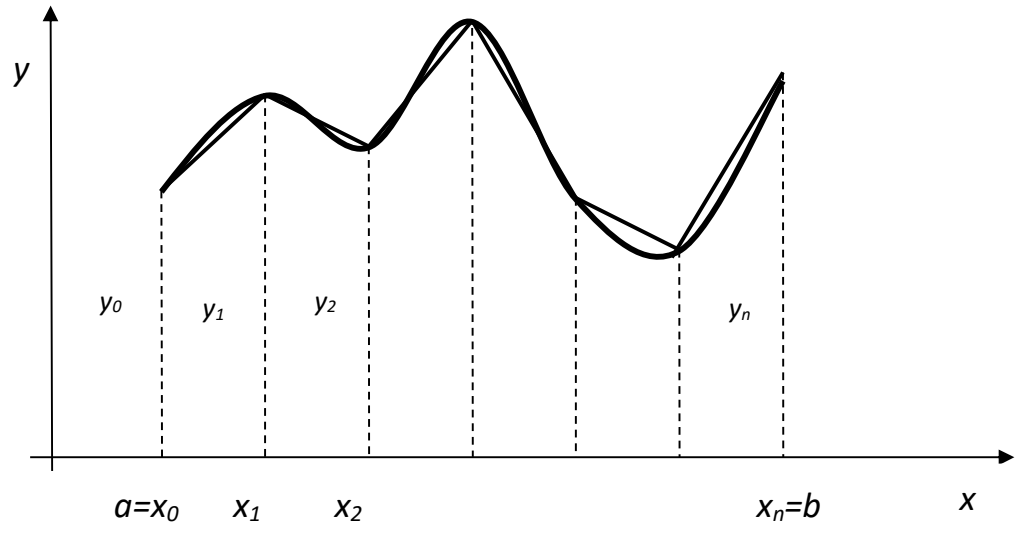


Рис. 7.4. Геометрична інтерпретація узагальненої формули трапецій

$$\int_a^b y(x)dx = h\left(\frac{y_0}{2} + y_1 + y_2 + \ldots + y_{n-1} + \frac{y_n}{2}\right). \qquad (7.13)$$

Геометрична інтерпретація узагальненої формули трапецій приведена рис.7.4. Залишковий член формули (7.13) дорівнює

$$R(h) = -\frac{nh^3}{12} y''(\zeta) = -\frac{(b-a)h^2}{12} y''(\zeta), \qquad (7.14)$$

де $\zeta \in [a,\ b]$.

Узагальнена формула Сімпсона. Нехай $n = 2m$ і задано значення функції $y(x)$ у рівновіддалених вузлах $x_0=a,\ x_1,\ x_2,\ \ldots,\ x_n=b$ з кроком $h = \frac{b-a}{n} = \frac{b-a}{2m}$. Якщо тепер застосувати формулу (7.12) до кожного з проміжків $[x_0,\ x_2],\ [x_2,\ x_4],\ \ldots,[x_{2m-2},\ x_{2m}]$, завдовжки $2h$ кожний, то отримаємо

$$\int_a^b y(x)dx = \frac{h}{3}[(y_0 + y_{2m}) + 4(y_1 + y_3 + \ldots + y_{2m-1}) + 2(y_2 + y_4 + \ldots + y_{2m-2})]. \qquad (7.15)$$

Формулу (7.15) частіше записують у вигляді

$$\int_a^b y(x)dx = \frac{h}{3}[y_0 + y_n + 4\sigma_1 + 2\sigma_2],$$

де $\sigma_1 = y_1 + y_3 + \ldots + y_{2m-1},\ \sigma_2 = y_2 + y_4 + \ldots + y_{2m-2}$. Залишковий член узагальненої формули Сімпсона дорівнює

$$R(h) = -\frac{nh^5}{90} y^{IV}(\zeta) = -\frac{(b-a)h^4}{180} y^{IV}(\zeta), \qquad (7.16)$$

де $\zeta \in [a,\ b]$. Геометричну інтерпретацію узагальненої формули Сімпсона показано на рис.7.5. Маючи формулу для залишкового члена, можна визначити крок h, який забезпечує точність чисельного інтегруванн, не меншу за $\varepsilon > 0$. Для цього на інтервалі інтегрування потрібно знайти найбільше за абсолютною величиною значення відповідної похідної, яка входить до залишкового члена, а потім абсолютну величину залишкового члена прирівняти до ε. Наприклад, для (7.16) необхідно спочатку обчислити

$$M_4 = \max_{[a,b]} \left| y^{IV}(x) \right|$$

і далі з нерівності $(b-a)\frac{h^4}{180}M_4 \le \varepsilon$ визначити крок h, який забезпечує потрібну точність ε

$$h \le \sqrt[4]{\frac{180\varepsilon}{(b-a)M_4}}.$$

Для виразу (7.14) відповідно маємо $M_2 = \max\limits_{[a,b]} \left| y''(x) \right|$ і $h \le \sqrt{\frac{12\varepsilon}{(b-a)M_2}}$.

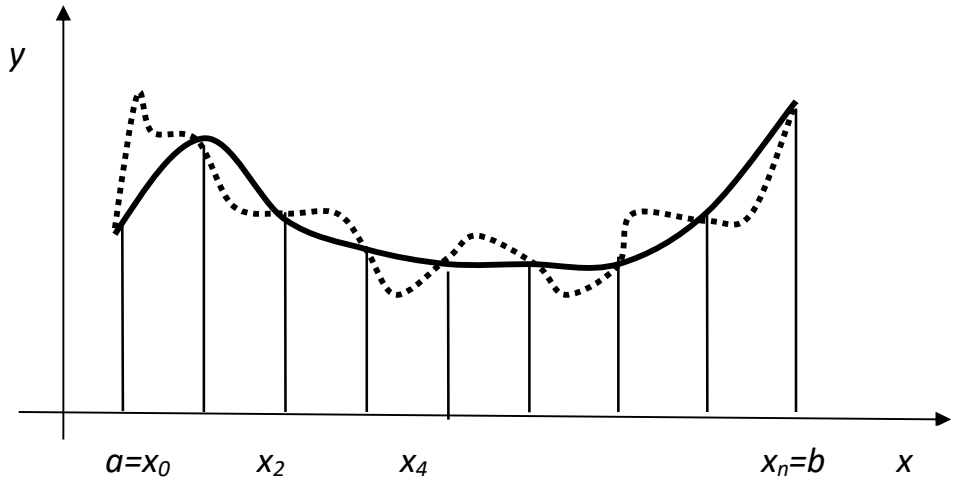


Рис. 7.5. Геометрична інтерпретація узагальненої формули Сімпсона

5.5. Метод подвійного перерахунку

Розглянутий вище спосіб обчислення кроку h потребує визначення максимуму відповідної похідної. Очевидно, що бажано мати спосіб автоматичного визначення кроку, який забезпечує потрібну точність. Таку можливість надає ***принцип Рунге*** , або так званий ***метод подвійного перерахунку***, який полягає в наступному.

Обчислимо $\int\limits_a^b f(x)dx$, наприклад, за узагальненою формулою трапецій з кількістю часткових проміжків n

$$\int\limits_a^b f(x)dx = I_n - \frac{f''(\xi_n)(b-a)^3}{12n^2}, \qquad (7.17)$$

а потім - з числом проміжків $2n$

$$\int_a^b f(x)dx = I_{2n} - \frac{f''(\xi_2)(b-a)^3}{12(2n)^2}. \tag{7.18}$$

Віднімаючи рівняння (7.17) від (7.18) маємо

$$0 = I_n - I_{2n} - 3\frac{f''(\xi)(b-a)^3}{12(2n)^2}. \tag{7.19}$$

Зрозуміло, що у виразі (7.19) доданок $3\frac{f''(\xi)(b-a)^3}{12(2n)^2}$ насправді є залишковим членом обчислення I_{2n} помноженим на три. Інакше кажучи,

$$R_{2n} = \frac{|f''(\xi)(b-a)^3}{12(2n)^2} = \frac{1}{3}|I_n - I_{2n}|.$$

Аналогічно для формули Сімпсона маємо

$$R_{2n} = \frac{|f^{(IV)}(\xi)(b-a)^5}{180(4n)^2} = \frac{1}{15}|I_n - I_{2n}|.$$

Таким чином, спочатку обчислюють шуканий інтеграл I_n за вибраною узагальненою формулою з кількістю часткових проміжків n, потім число n подвоюють і обчислюють I_{2n}. Залишковий член оцінюють за формулою

$$R_n = \frac{1}{2^r - 1}\left|I_n - I_{2n}\right|,$$

де $r = 2$ для формули трапецій і $r = 4$ для формули Сімпсона. Якщо $R_n \le \varepsilon$, то вважають, що шуканий інтеграл дорівнює I_{2n}. У противному випадку обчислення повторюють з кількістю часткових проміжків $2n$ та $4n$. Цей процес продовжують, доки не буде виконуватися нерівність $R_n \le \varepsilon$. За початкове значення n найчастіше беруть число, близьке до $1/\sqrt[r]{\varepsilon}$.

Приклад 7.2. Обчислити $\int_0^1 e^{-x^2}dx$ за допомогою узагальненої формули Сімпсона з точністю 0.001.

Розв'язок. Візьмемо спочатку $n = 2$, тоді

$$\int_0^1 e^{-x^2} dx = \frac{1}{6}(1 + 4e^{-0.25} + e^{-1}) = \frac{1}{6}(1.36788 + 4 \cdot 0.77880 = 0.74718.$$

Подвоїмо кількість часткових проміжків і складемо таблицю значень підінтегральної функції

x_i	0	0.25	0.5	0.75	1
y_i	1,00000	0.93941	0.77880	0.56978	0.36786

Далі

$$\int_0^1 e^{-x^2} dx = \frac{1}{12}(1.36788 + 4(0.93941 + 0.56978) + 2 \cdot 0.77880 = 0.74685.$$

Відповідно до принципу Рунге залишковий член

$$\left| \frac{0.74718 - 0.74685}{15} \right| \cong 2.2 \cdot 10^{-5}$$

і є меншим за 0.001. Тобто 0.74685 є шуканим значенням інтеграла.

Псевдокод алгоритму обчислення інтеграла методом подвійного перерахунку за допомогою узагальненої формули трапецій

Вхідні дані:
a,b – границі інтегрування;
eps – задана точність;
f(x) – підінтегральна функція;
Вихід: I – значення інтеграла з заданою точністю.

1. *Обчислити початкову кількість часткових проміжків інтегрування* $n = (b - a) / \sqrt{eps}$.

2. *Обчислити значення інтеграла* I_n *за допомогою узагальненої формули трапецій.*

3. *n = n * 2.*

4. *Обчислити значення інтеграла* I_{2n} *за допомогою узагальненої формули трапецій.*

5. `while(abs(`I_n`-` I_{2n}`) > 3 * eps){`
   I_n `=` I_{2n}`;`
   `n *= 2;`
   Обчислити значення інтеграла I_{2n}

}

6. `I =` I_{2n}.

Зауваження. Для того, щоб обчислити інтеграл за допомогою узагальненої формули Сімпсона, достатньо замінити обчислення початкової кількості часткових проміжків інтегрування на $n=(b-a)/\sqrt[4]{eps}$, а умову зупинки – на `while(abs(`I_n `-` I_{2n}`) > 15 * eps)`.

7.6. Контрольні запитання та завдання до розділу 7

1. У чому полягає загальний підхід до розв'язання задачі чисельного диференціювання?
2. Як оцінюється похибка чисельного диференціювання?
3. Якими інтерполяційними поліномами зручніше користуватися для розв'язання задачі чисельного диференціювання?
4. Для функції $f(x)=0.5\sin(x)$, заданої на проміжку $[0.1;0.2]$, обчислити значення її похідної в точці $x=0.1$ з похибкою не більшою за 10^{-2}.
5. Для функції, заданої таблицею,

x_i	0.100000	0.133333	0.166667	0.200000
f_i	0.049917	0.066469	0.082948	0.099335

обчисліть значення її похідної в точці $x=0.15$ з точністю не гірше за 10^{-2}.

6. Для функції, заданої таблицею,

x_i	0.500000	0.566667	0.633333	0.700000
f_i	-0.693147	-0.567984	-0.456758	-0.356675

обчислити значення її похідної в точці $x=0.55$ з точністю не гірше за 10^{-2}.

7. Як з наочних міркувань побудувати перші дві квадратурні формули?
8. У чому полягає загальний підхід до розв'язання задачі чисельного інтегрування?
9. Як обчислюються коефіцієнти Котеса?
10. Що визначає залишковий член інтерполяційної квадратурної формули?
11. У чому полягає головний недолік інтерполяційних квадратурних формул?
12. У чому полягає ідея побудови узагальнених квадратурних формул?
13. Які переваги дають узагальнені квадратурні формули?
14. За допомогою узагальненої формули трапецій обчислити приблизне значення інтегралу $\int_1^2 \sin(x)dx$ з кроком $h=0.1$.

15. Визначити крок h, який забезпечить обчислення інтеграла $\int_1^2 sin(x)dx$ за допомогою узагальненої формули трапецій з похибкою не більшою за 10^{-3}.
16. У чому полягає принцип подвійного перерахунку?
17. З яких міркувань обирається початковий крок для подвійного перерахунку?

РОЗДІЛ 8. ЧИСЕЛЬНІ МЕТОДИ РОЗВ'ЯЗАННЯ ЗВИЧАЙНИХ ДИФЕРЕНЦІАЛЬНИХ РІВНЯНЬ

Існують дві групи методів наближеного розв'язання звичайних диференціальних рівнянь: аналітичні та чисельні. У свою чергу чисельні методи поділяються на одно- та багатокрокові.

Звичайним диференціальним рівнянням порядку r називається рівняння

$$F(x, y(x), y'(x), ..., y^{r}(x)) = 0.$$

Розв'язання диференціального рівнянн полягає у знаходженні функцій $y(x)$, які відповідають цьому рівнянню для всіх значень x на визначеному скінченному чи нескінченному інтервалі (a,b). Загальний розв'язок рівняння при цьому має вигляд

$$y = y(x, C_1, C_2, ..., C_r),$$

де $C_1, C_2, ..., C_r$ - сталі інтегрування. Кожен вибір цих сталих дає частковий розв'язок.

Задача Коші зводиться до знаходження часткового розв'язку, який відповідає r початковим умовам

$$y(x_0) = y_0,\ y'(x_0) = y'_0, ..., y^{(r-1)}(x_0) = y_0^{(r-1)},$$

за якими обчислюють r сталих $C_1, C_2, ..., C_r$. ***Чисельний розв'язок задачі Коші*** отримують у вигляді таблиці наближених значень функції $y(x)$.

8.1. Аналітичні методи розв'язання задачі Коші

Одним з аналітичних методів розв'язання задачі Коші є так званий метод степеневих рядів. Нехай задано рівняння

$$y^{(n)}(x) = f(x, y, y', ..., y^{(n-1)}), \tag{8.1}$$

з початковими умовами

$$y(x_0) = y_{00}, y(x_1) = y_{01}, ... y(x_{n-1}) = y_{0,n-1}. \tag{8.2}$$

Якщо права частина рівняння (8.1) в околі точки $(x_0, y_{00}, y_{01}, ..., y_{0,n-1})$ є аналітичною функцією, то при значеннях x, близьких до x_0, існує єдиний розв'язок задачі Коші, причому цей розв'язок може бути розкладено у ряд Тейлора

$$y(x) = y(x_0) + \frac{y'(x_0)}{1!}(x - x_0) + \frac{y''(x_0)}{2!}(x - x_0)^2 + \\ + \frac{y^{(n)}(x_0)}{n!}(x - x_0)^n + \frac{y^{(n+1)}(x_0)}{(n+1)!}(x - x_0)^{(n+1)} + ... \quad (8.3)$$

За допомогою початкових умов визначають перші n складових (8.3). Значення $(n+1)$-ї складової визначимо з рівняння (8.1), підставляючи $x = x_0$ і використовуючи початкові умови (8.2). Тоді маємо

$$y^{(n)}(x_0) = f(x_0, y_{00}, ..., y_{0,n-1}).$$

Величини $y^{(n+1)}(x_0), y^{(n+2)}(x_0), y^{(n+3)}(x_0), ...$ послідовно визначимо шляхом диференціювання рівняння (8.1) і підстановкою

$$x = x_0, \quad y^{(k)}(x_0) = y_{0k}, \quad k = 0,1,2,...,n-1,n...$$

Отже, у такий спосіб можна визначити стільки доданків для формули (8.3), скільки їх необхідно для отримання розв'язку з потрібною точністю.

Приклад 8.1. Для функції $y = y(x)$, яка задовольняє рівняння

$$y'' + xy' + y = 0 \quad (8.4)$$

і початкові умови $y(0) = 0, y'(0) = 1$, скласти таблицю значень інтегральної кривої на відрізку [0; 0.1] з кроком $h = 0.05$. Розв'язок отримати із похибкою не більше 0.510^{-4}.

Розв'язок. Запишемо ряд Маклорена

$$y(x) = y(0) + y'(0)x + y''(0)x^2 + ... + \frac{y^{(n)}(0)}{n!}x^n,$$

при цьому $y(0) = 0, y'(0) = 1$. Із рівняння (8.4) маємо $y'' = -xy' - y$. Очевидно, що $y'' = 0$ при $x = 0$. Послідовно диференціюючи (8.4) отримаємо

$$y^{III}(x) = -xy'' - 2y'$$

$$y^{iV}(x) = -xy^{III} - 3y'',$$

$$y^{V}(x) = -xy^{IV} - 4y^{III},$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots .$$

$$y^{(n+1)}(x) = -xy^{(n)} - (n-1)y^{(n-1)}.$$

Отже $y'''(0) = -2, y^{(IV)}(0) = 0, y^{(V)} = 8$, і взагалі

$$y^{(2n)} = 0, y^{(2n+1)} = -2n, y^{(2n-1)}(0) = (-1)^n 2^n n!.$$

Таким чином,

$$y(x) = x - \frac{x^3}{3} + \frac{x^5}{15} - \ldots + \frac{(-1)^n 2^n n!}{(2n+1)!} + \ldots.$$

це знакозмінний ряд і його члени монотонно зменшуються за абсолютною величиною при $x \in [0;0.2]$. Залишок такого ряду за абсолютною величиною менше першого члена, що відкидається, отже перші два члени ряду

$$y(x) = x - \frac{x^3}{3}, \tag{8.5}$$

вочевидь, будуть давати наближений розв'язок з похибкою не гірше за $\frac{x^5}{15} \le \frac{0.2^5}{15} = 0.00002$.

Отже таблиця інтегральної кривої може бути побудована за формулою (8.5).

8.2. Чисельні методи розв'язання задачі Коші

Розрізняють дві групи чисельних методів розв'язання задачі Коші:

1) однокрокові методи (методи Рунге-Кутта), в яких для обчислення функції $y(x)$ у черговій точці потрібна інформація тільки про попередню точку;

2) багатокрокові методи (методи Адамса), у яких для визначення чергового значення $y(x)$ потрібна інформація більш ніж про одну з попередніх точок.

8.2.1. Однокрокові методи

Однокрокові методи розв'язяння задачі Коші характеризуються наступним:

1) для обчислення y_{k+1} необхідна інформація лише в точці (x_k, y_k);

2) розв'язки за цими методами узгоджуються з рядом Тейлора до членів порядку h^r включно;

3) вони не потребують обчислення похідних $f(x, y)$, а лише обчислення значень самої функції.

Нехай треба розв'язати задачу Коші для нелінійного диференціального рівняння першого порядку

$$y'(x) = f(x, y) \tag{8.5}$$

з початковими умовами $y(x_0) = y_0$. Наближений розв'язок задачі Коші будемо шукати на скінченній множині точок відрізку $[x_0, x_0 + l]$, яку називають сіткою. Виберемо сітку $w_h = \{x_j\}_{j=0}^{N}$, де $x_j = x_0 + jh$, $h = l/N$, N – додатне ціле.

Розглянемо рівняння $y' = y$. Його розв'язок $y = Ce^x$, C – довільна стала (рис. 8.1). Рівняння має вигляд $y' = f(x, y)$, тобто можна обчислити похідну інтегральної кривої у кожній точці (x, y). Тоді розв'язок диференціального рівняння знаходять наступним чином.

У початковий момент маємо лише одну точку (x_0, y_0), тангенс нахилу дотичної в якій дорівнює

$$y' = f(x_0, y_0).$$

Побудуємо дотичну у точці (x_0, y_0) і перейдемо вздовж неї на малу відстань h, тобто обчислимо

$$y' = f(x_1, y_1),$$

де $x_1 = x_0 + h, y_1 = y_0 + hf(x_0, y_0)$. Отже

$$y'_1 = f(x_1, y_1),\ x_1 = x_0 + h,\ y_1 = y_0 + hy'$$

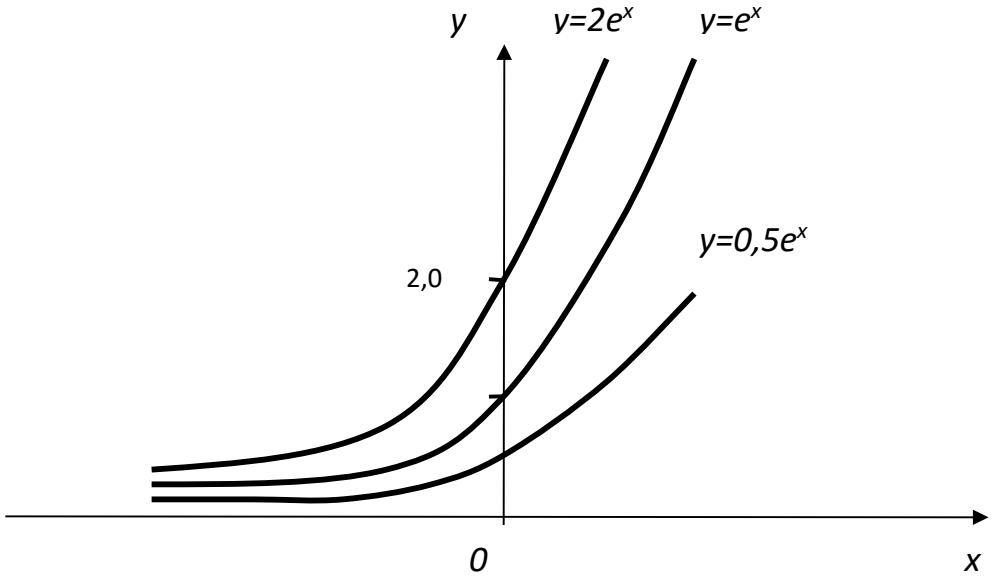


Рис. 8.1. Розв'язки рівняння $y = Ce^x$

Продовжуючи цей процес для наступних точок отримаємо ***ламану Ейлера.*** В аналізі доведено, що якщо $\varphi(x)$ – ламана Ейлера, а $\varphi^*(x)$ – точний розв'язок рівняння (8.5), то $\lim_{h->0} |\varphi(x) - \varphi^*(x)| = 0$ (рис. 8.2).

Найпростіші методи Рунге-Кутта можуть бути отримані з наочних міркувань (рис.8.3). Рівнянням прямої L є

$$y_{k+1} = y_k + y'_k(x_{k+1} - x_k) = y_k + hy'_k,$$

при цьому $y'_k = f(x_k, y_k)$. Таким чином

$$\begin{cases} y_{k+1} = y_k + hf(x_k, y_k) \\ x_{k+1} = x_k + h \end{cases} \qquad (8.6)$$

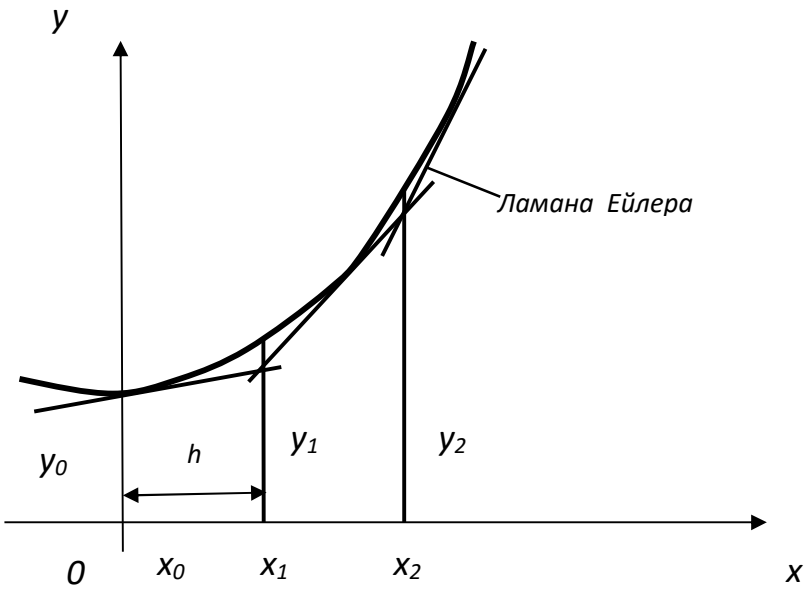


Рис. 8.2. Геометрична інтерпретація розв'язку диференціального рівняння однокроковим методом

Це розрахункові формули методу Ейлера. За відсутності похибок округлення локальна похибка (тобто похибка на кроці) становить $O(h^2)$. Глобальна (на інтервалі) похибка методу Ейлера становить $O(h)$.

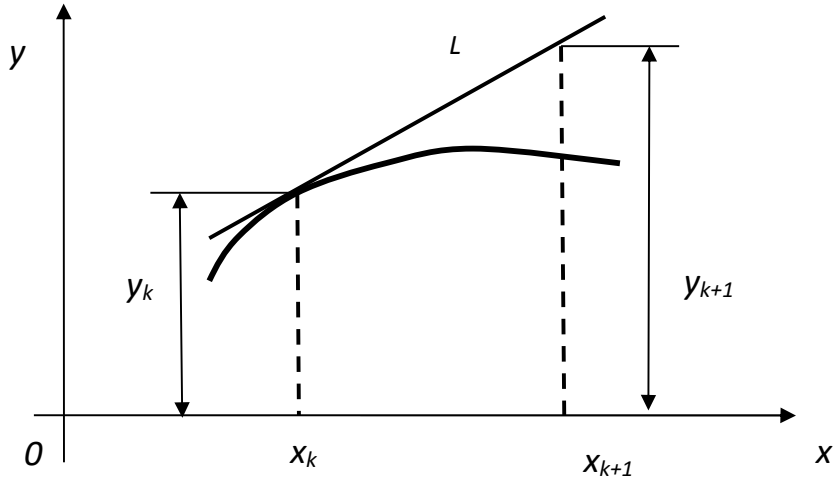


Рис. 8.3. Геометрична інтерпретація методу Ейлера

Приклад 8.2. Скласти таблицю значень інтегральної кривої рівняння $y'\,y - y^2 + 2x = 0$ на відрізку $[0;1]$ з початковою умовою $y(0) = 1$ і кроком таблиці $h = 0.2$.

Розв'язок. Із заданого рівняння маємо $y' = y - 2x/y$. Отже, $f(x, y) \equiv y - 2x/y$. Оскільки $x_0 = 0$ і $y_0 = 1$, то для y_1 маємо

$$y_1 = y_0 + h(y_0 - x_0 / y_0) = 1 + 0.2 \cdot 1 = 1.2;$$
$$x_1 = x_0 + h = 0 + 0.2 = 0.2;$$
$$y_2 = y_1 + h(y_1 - x_1 / y_1) = 1.3733;$$
$$x_2 = x_1 + h = 0,4 \quad \text{і т.д.}$$

Метод Ейлера-Коші (метод предиктор-коректор)

За цим методом знаходять тангенс середнього кута нахилу дотичних до двох точок (x_k, y_k) та (x_{k+1}, y_{k+1}) (рис. 8.4).

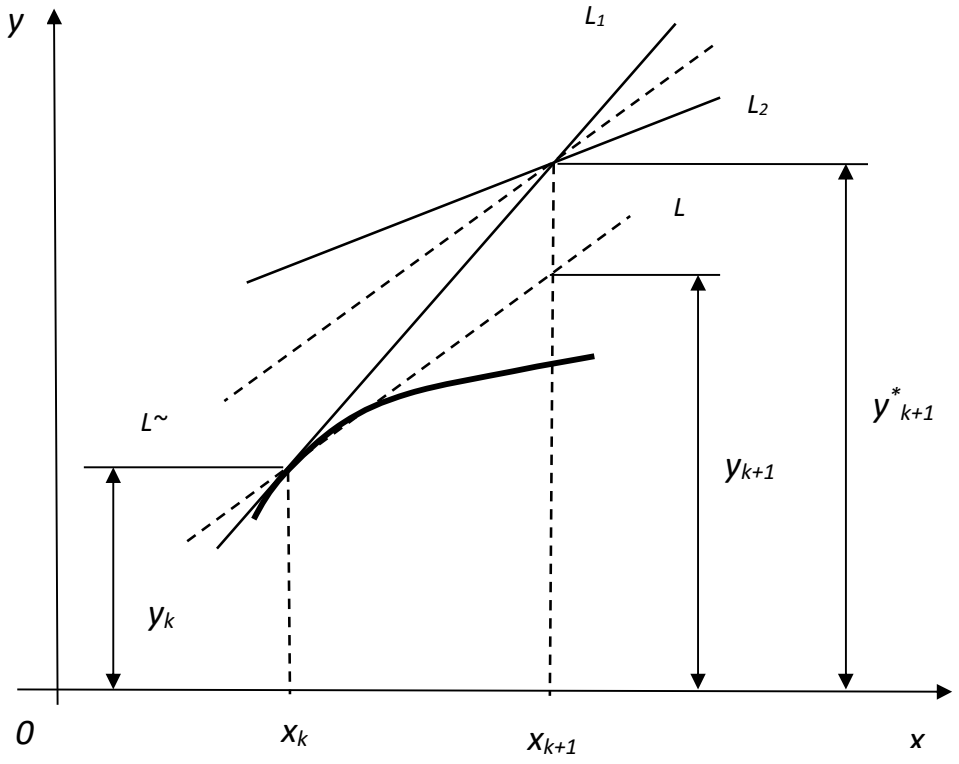


Рис. 8.4. Геометрична інтерпретація методу Ейлера-Коші

$$y^*_{k+1} = y_k + hf(x_k, y_k),$$
$$y_{k+1} = y_k + h\frac{f(x_k, y_k) + f(x_{k+1}, y^*_{k+1})}{2}. \qquad (8.7)$$

Не важко помітити, що формула (8.7) співпадає з рядом Тейлора до h^2 включно, тобто

$$y(x) = y_0 + hy'_0 + h^2 \frac{y''_0}{2!},\ y''_0 = \frac{y'_1 - y'_0}{h}.$$

Таким чином,

$$y(x) = y_0 + hy'_0 + h^2 \frac{y'_1 - y'_0}{2h} = y_0 + h\frac{y'_1 + y'_0}{2}.$$

Локальна похибка методу складає $O(h^3)$, глобальна - $O(h^2)$.

Відома модифікація цього методу, яка носить назву ***удосконалений метод Ейлера-Коші***. Відповідно до цього методу

$$\begin{cases} y^*_{(k+1)/2} = y_k + \frac{h}{2} f(x_k, y_k), \\ y_{k+1} = y_k + hf(x_k + \frac{h}{2}, y^*_{(k+1)/2}), \end{cases}$$

тобто коректор обчислюють не в точці (x_{k+1}, y^*_{k+1}), а в точці $(x_k + h/2, y_{(k+1)/2})$.

Метод Рунге-Кутта четвертого порядку точності

Загальний підхід до побудови формул Рунге-Кутта полягає в наступному. Приріст інтегральної функції Δy_k представляють у вигляді зваженої суми значень правої частини рівняння в деякій системі точок, тобто у вигляді

$$\Delta y_k = p_1 hf(\xi_1, \eta_1) + p_2 hf(\xi_2, \eta_2) + ... + p_r hf(\xi_r, \eta_r).$$

При цьому коефіцієнти p_i і точки (ξ_i, η_i) знаходять за формулою Тейлора з умов рівності приросту Δy_k сумі членів ряду Тейлора до степеня h^r включно. При $r = 1$ маємо формулу Ейлера; при $r = 2$ – формулу Ейлера-Коші; при $r = 4$ – формулу Рунге-Кутта 4-го порядку. За методом Рунге-Кутта 4-го порядку перехід від точки x_k до точки x_{k+1} здійснюють за формулами

$$k_1 = hf(x_k, y_k),$$
$$k_2 = hf(x_k + \frac{h}{2}, y_k + \frac{k_1}{2}),$$
$$k_3 = hf(x_k + \frac{h}{2}, y_k + \frac{k_2}{2}),$$
$$k_4 = hf(x_k + h, y_k + k_3),$$
$$\Delta y_k = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),$$
$$y_{k+1} = y_k + \Delta y_k,\ x_{k+1} = x_k + h.$$

Цей метод характеризується локальною похибкою $O(h^5)$ і глобальною - $O(h^4)$.

8.2.2. Метод подвійного перерахунку (принцип Рунге)

Оскільки не існує загальноприйнятної аналітичної оцінки точності однокрокових методів в залежності від кроку h, то з цією метою використовують так званий метод подвійного перерахунку, або ***принцип Рунге***, який дає можливість автоматичного вибору кроку h. Основна його ідея вже використовувалась для автоматичного вибору кроку інтегрування у розділі 7. Полягає вона в тому, що перехід з точки x_{i-1} в точку x_i спочатку здійснюють з кроком h, а потім – з вдвічі меншим кроком $h/2$. Похибку в точці x_i при цьому оцінюють за формулою

$$\frac{\left|y_i^h - y_i^{h/2}\right|}{2^r - 1} \le \varepsilon, \qquad (8.8)$$

де y_i^h - розв'язок в точці x_i, отриманий з кроком h, $y_i^{h/2}$ - розв'язок в точці x_i, отриманий з кроком $h/2$, r - порядок методу. Для методів Ейлера, Ейлера-Коші та Рунге-Кутта 4-го порядку точності значення r доівнює 1, 2 та 4 відповідно.

Якщо нерівність (8.8) не виконується, то обчислення проводять з кроком $h/4, h/8$ і т. д. Як розв'язок в точці x_i беруть значення, обчислене з найменшим кроком.

Принциповим при цьому є вибір початкового кроку h_0. Зрозуміло, що якщо крок h_0 занадто великий, то треба буде виконати багато повторів подвійного перерахунку. Природнім є вибір h_0 виходячи з порядку глобальної помилки того чи іншого методу, яка складає $O(h^r)$. Отже, якщо вважати $\varepsilon \approx h^r$, маємо слушну оцінку вибору h_0, а саме: $h_0 = \sqrt[r]{\varepsilon}$.

Наведені оцінки точності методів Рунге-Кутта справедливі, якщо похибки округлення на кожному кроці обмежені величиною Ch^{r+2}, причому C не залежить від h.

Перевірити цю умову безпосередньо досить важко. На практиці можна перевірити виконання умови

$$\left| 2^r \frac{y_h - y_{h/2}}{y_{2h} - y_h} - 1 \right| < 0{,}1$$

в точці $x_0 + l$ або $x_0 + l/2$. Якщо похибка округлення превалює над похибкою методу, ця умова грубо порушується. У цьому випадку слід перейти до обчислень з підвищеною точністю.

Для методів четвертого порядку точності можна також обчислити відхилення

$$\theta = \frac{\left|k_2 - k_3\right|}{\left|k_1 - k_2\right|}.$$

Крок h вважають прийнятним, якщо величина θ не перевищує декількох сотих (до 0.05). Нарешті, маючи наближення, отримані за двома сусідніми кроками, можна знайти уточнений (за Річардсоном) розв'язок

$$y_i^* = \frac{2^r y_i^h - y_i^{2h}}{2^r - 1},$$

що має похибку $O(h^{r+1})$, а не $O(h^r)$.

Псевдокод розв'язання задачі Коші методом подвійного перерахунку

Вхідні дані:
a,b – інтервал побудови таблиці інтегральної кривої;
eps – задана точність;
f(x,y) – права частина диференціального рівняння;
x_0, y_0 *- початкові умови;*
h – крок побудови таблиці інтегральної кривої;
N – кількість вузлів сітки;
Вихід: Y - вектор значень інтегральної кривої.

Y[0] = y_0*;*

Обчислити початкову кількість часткових проміжків інтегрування $n = (b - a)/\sqrt[r]{eps}$, $r = 2$ *для метода Ейлера-Коші й r = 4 для метода 4-го порядку точності.*

for(i = 1; i < N; i++){

Обчислити початковий крок η *обчислення значень інтегральної кривої для проміжку* $[x_{i-1}; x_i]$.

```
η = (x_i − x_{i−1})/ n; //x_i і x_{i−1}– сусідні вузли сітки.
x = x_{i−1};
//обчислити значення інтегральної кривої yk
//в наступному вузлі сітки x_i з кроком η
yk_1 =Y[i − 1];
for(k = 0; k < n; k++){
      yk = rk_method(x,yk_1,η);
      //rk_method – однокроковий метод, що застосовуєтья
      x += η;
      yk_1 = yk;
}

//запам'ятати його
y = yk;

//зменшити крок η вдвічі і обчислити
// значення інтегральної кривої yk в
// наступному вузлі сітки x_i
n = n * 2;
```

```
      η = (x_i − x_{i−1}) / n;
      yk_1 = Y[i − 1];
      for(k = 0; k < n; k++){
            yk = rk_method(x, yk_1, η);
            //rk_method – однокроковий метод, що застосовуєтья
            x += η;
            yk_1 = yk;
      }
      //перший етап подвійного перахунку для
      // інтервалу [x_{i−1}; x_i] завершено
      //далі зменьшувати крок, поки не буде
      // досягнена задана точність у вузлі x_i
      while( abs(yk − y) > (2^r − 1)* eps){
            y = yk;
            n *= 2;
            η = (x_i − x_{i−1}) / n;
            x = x_{i−1};
            yk_1 = Y[i − 1];
            for(k = 0; k < n; k++){
                  yk = rk_method(x, yk_1, η);
                  //rk_method – однокроковий метод, що застосовуєтья
                  x += η;
                  yk_1 = yk;

            }//end for
      }//end while
      Y[i] = yk;
      //перейти до наступного вузла сітки
}
```

8.2.3. Розв’язання задачі Коші для диференціальних рівнянь вищих порядків і систем диференціальних рівнянь першого порядку

Методи розв’язання задачі Коші можуть бути використані і для інтегрування систем диференціальних рівнянь першого порядку. Наприклад, для системи

$$y' = f(x, y, z), \quad z' = g(x, y, z),$$

з початковими умовами $y(x_0) = y_0$, $z(x_0) = z_0$ формули для методу Ейлера мають вигляд

$$y_{k+1} = y_k + hf(x_k, y_k, z_k),$$

$$z_{k+1} = z_k + hg(x_k, y_k, z_k),$$

$$x_{k+1} = x_k + h,$$

тобто знаходження розв'язків виконується паралельно за ідентичними формулами. Похибку розв'язку системи для наведеного прикладу визначають як

$$\max(\left| y_i^h - y_i^{2h} \right|, \left| z_i^h - z_i^{2h} \right|).$$

Для розв'язання диференціальних рівнянь вищих порядків їх необхідно привести до системи диференціальних рівнянь першого порядку, розв'язання якої виконується одним із розглянутих методів. Наприклад, рівняння другого порядку $y'' = f(x, y, y')$ за допомогою заміни $y' = t$ зводиться до системи вигляду

$$y' = t,$$

$$t' = f(x, y, t).$$

Приклад 8.3.. Отримати розрахункові формули за методом Ейлера для диференціального рівнянння другого порядку

$$y'' + y'/x + y = 0.$$

Розв'язок. Виконуючи заміну $y' = t$, маємо

$$y' = y, \; t' = -t/x - y$$

і далі

$$y_{k+1} = y_k + ht_k,$$

$$t_{k+1} = t_k + h(-t_k/x_k - y_k),$$

$$x_{k+1} = x_k + h.$$

8.2.4. Багатокрокові методи (методи Адамса)

Нехай задано рівняння

$$y'(x) = f(x, y), \tag{8.9}$$

яке відповідає задачі Коші з початковими умовами $y(x_0) = y_0$. Відомі також наближені значення $y_{j-m}, \ldots, y_{j-1}, y_j$ з кроком h, що є розв'язками задачі Коші (8.9). Позначимо

$$f_j = f(x_j, y_j).$$

Побудуємо далі інтерполяційний поліном Лагранжа степеня m

$$L_m(x) = \sum_{i=0}^{m} p_{mi} f_{i-j}\,, \tag{8.10}$$

який задовольняє умові $L_m(x_{i-j}) = f_{i-j}$, $i = 0,1,...,m$.

Оскільки для точного розв'язку $u(x)$ задачі Коші має виконуватися рівність

$$u_{j+1} = u_j + \int_{x_j}^{x_{j+1}} f(x,u(x))dx\,,$$

де $x_{j+1} = x_j + h$, $u_k = u(x_k)$, то можна припустити, що для наближеного розв'язку $y(x)$ має місце подібна рівність

$$y_{j+1} = y_j + \int_{x_j}^{x_{j+1}} L_m(x)dx\,.$$

Тоді з урахуванням (8.10) маємо

$$y_{j+1} = y_j + h\sum_{i=0}^{m} \alpha_{mi} f_{i-j}\,, \tag{8.11}$$

де f_{i-j} – значення правих частин рівняння (8.9), а $\alpha_{mi} = \frac{1}{h}\int_{x_j}^{x_{j+1}} p_{mi}(x)dx, i = 0,1,...,m$ - сталі, які не залежать ні від h, ні від j. Таким чином, значення α_{mi} можуть бути обчислені заздалегідь один раз і застосовуватися незалежно від виду правої частини (8.9).Зокрема,

при $m = 0, \alpha_{00} = 1$;

при $m = 1, \alpha_{10} = 3/2$, $\alpha_{11} = -1/2$;

при $m = 2, \alpha_{20} = 23/12$, $\alpha_{21} = -4/3$, $\alpha_{22} = 5/12$;

при $m = 3, \alpha_{30} = 55/24$, $\alpha_{31} = -59/24$, $\alpha_{32} = 37/24$, $\alpha_{33} = -3/8$.

Похибка методу Адамса має порядок $O(h^{m+1})$.

Як слідує із формули (8.1), для використання багатокрокового методу треба мати m + 1 наближень розв'язку задачі Коші. Ці наближення звичайно обчислюють однокроковим методом з точністю порядку (m + 1). У зв'язку з цим виникає питання про доцільність використання багатокрокових методів взагалі. Щоб відповісти на це питання, розглянемо обчислення за формулою (8.11) значень y_4 і y_5:

$$y_4 = y_3 + h(\alpha_{30}f_3 + \alpha_{31}f_2 + \alpha_{32}f_1 + \alpha_{33}f_0),$$
$$y_5 = y_4 + h(\alpha_{30}f_4 + \alpha_{31}f_3 + \alpha_{32}f_2 + \alpha_{33}f_1)\,. \tag{8.12}$$

.

З цих формул видно, що для переходу від точки y_4 до точки y_5 необхідно лише один раз обчислити праву чистину рівняння $f(x,y)$ для того, щоб отримати значення f_4, оскільки

значення f_1, f_2, f_3 вже відомі (див. (8.12)). Метод Рунге-Кутта 4-го порядку має точність $O(h^4)$, тобто таку ж, як і метод Адамса при $m=3$, але для переходу до наступної точки потребує чотирикратного обчислення $f(x,y)$. Принагідно зауважимо, що обчислення за методом Адамса виконуються за досить простими формулами.

8.3. Спеціальні формули розв'язання диференціальних рівнянь другого порядку

Оскільки диференціальні рівняння другого порядку зустрічаються в сучасних прикладних науково-технічних задачах частіше за інші, було розроблено спеціальні формули для їх наближеного розв'язання. Отже, маємо рівняння

$$y'' = f(x, y, y')$$

при початкових умовах

$$y(x_0) = y_0,\ y'(x_0) = y_0'.$$

Введемо наступні позначення

$$x_k = x_0 + kh,\ y(x_k) = y_k,\ y'(x_k) = y'_k,$$
$$f(x_k, y_k, y'_k) = f_k,\ k = 0,1,2...$$

1. Метод Рунге-Кутта

$$y_{k+1} = y_k + h \cdot y'_k + \frac{1}{6}(k_1 + k_2 + k_3) \cdot h,$$
$$y'_{k+1} = y'_k + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),$$

де

$$k_1 = hf(x_k, y_k, y'_k),$$
$$k_2 = hf(x_k + \frac{h}{2}, y_k + y'_k \frac{h}{2}, y'_k + \frac{k_1}{2}),$$
$$k_3 = hf(x_k + \frac{h}{2}, y_k + y'_k \frac{h}{2} + \frac{k_1}{4}h, y'_k + \frac{k_2}{2}),$$
$$k_4 = hf(x_k + h, y_k + y'_k h + \frac{k_2}{2}h, y'_k + k_3)$$

2. Метод «предиктор-коректор»

$$y'_{k+1} = y'_{k-3} + \frac{4}{3}(2f_k - f_{k-1} + 2f_{k-2})h,\ \text{(«предиктор»)},$$
$$y_{k+1} = y_{k-1} + \frac{1}{3}(y'_{k+1} + 4y_k + y'_{k-1})h,$$

$$y'_{k+1} = y'_{k-1} + \frac{1}{3}(f_{k+1} + 4f_k + f_{k-1})h, \qquad \text{(«коректор»)}.$$

Якщо $f(x, y, y')$ не залежить явно від y', то як преликтор застосовують

$$y_{k+1} = y_k + y_{k-2} - y_{k-3} + \frac{1}{4}(5f_k + 2f_{k-1} + 5f_{k-2})h^2,$$

а для корекції $y_{k+1} = 2y_k - y_{k-1} + \frac{1}{12}(f_{k+1} + 10f_k + f_{k-1})h^2$ (***формула Штермера***).

8.4. Контрольні запитання та завдання до розділу 8

1. Які методи чисельного розв'язання задачі Коші відомі?
2. В чому полягають недоліки аналітичних методів розв'язання задачі Коші?
3. Чим характеризуються однокрокові методи розв'язання задачі Коші?
4. Наведіть розрахункові формули методів Ейлера й Ейлера-Коші.
5. Чому дорівнює похибка на кроці та похибка на інтервалі для методів Ейлера й Ейлера-Коші?
6. Наведіть розрахункові формули методу Рунге-Кутта четвертого порядку.
7. Чому дорівнює похибка на кроці і похибка на інтервалі для методу Рунге-Кутта четвертого порядку?
8. За допомогою методу Ейлера побудувати таблицю значень інтегральної кривої для рівняння $y'x - x^2 - 3y = 0$ з початковими умовами $x_0 = 1, y_0 = 1$ на проміжку $[1;2]$ з кроком 0.2. Визначити похибку розв'язку в точці $x = 2$, порівнявши його зі значенням точного розв'язку цього рівняння є $y(x) = 2x^3 - x^2$.
9. За допомогою методу Рунге-Кутта 4-го порядку побудувати таблицю значень інтегральної кривої для рівняння $y'x - x^2 - 3y = 0$ з початковими умовами $x_0 = 1, y_0 = 1$ на проміжку $[1;2]$ з кроком 0.2. Визначити похибку розв'язку в точці $x = 2$, порівнявши його зі значенням точного розв'язку цього рівняння є $y(x) = 2x^3 - x^2$.
10. У чому полягає застосування методу подвійного перерахунку до розв'язання задачі Коші?
11. З яких міркувань вибирають початковий крок для подвійного перерахунку.
12. У чому полягає ідея побудови багатокрокових методів Адамса?
13. Наведіть оцінку похибки на інтервалі для багатокрокових методів.
14. Яким чином визначають перші (m + 1) наближення, необхідні для старту багатокрокового методу?
15. Порівняйте методи Рунге-Кутта й методи Адамса.

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Алберг Дж., Нильсон Э., Уолш Дж. Теория сплайнов и ее приложения. – М.: Мир, 1972, - 318 с.
2. Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. - М.: Мир, 1979. – 536 с.
3. Бабушка И., Витасек Э., Прагер М. Численные процессы решения дифференциальных уравнений. – М: Мир, 1969. – 368 с.
4. Бахвалов Н.С. Численные методы (анализ, алгебра, обыкновенные дифференциальные уравнения). – М.: Наука, 1975. – 632 с.
5. Васильев Ф. П. Численные методы решения экстремальных задач. – М.: Наука, 1980. - 520 с.
6. ГОСТ 19.701-90. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения.
7. ГОСТ 8.381-80 ГСИ. Эталоны. Способы выражения погрешностей.
8. Демидович Б.П., Марон И.А. Основы вычислительной математики. – М: Наука, 1966. – 664 с.
9. Демидович Б.П., Марон И.А., Шувалова Э.З. Численные методы анализа. Приближение функций, дифференциальные и интегральные уравнения. – М: Наука, 1967. – 368 с.
10. Калиткин Н.Н. Численные методы. – М.: Наука, 1978. – 512 с.
11. Кондаков Н.И. Логический словарь-справочник. – М.: Наука, 1975. – 720 с.
12. Кормен Т., Лейзерсон Ч. Алгоритмы: построение и анализ. – М: Издательский дом «Вильямс», 2005. – 1296 с.
13. Кузюрин Н.Н., Фомин С.А. Эффективные алгоритмы и сложность вичислений, 2008. –326 с.
14. Куц А.К. Математическая логика и теория алгоритмов: учебное пособие. – Омск: Диалог-Сибирь, 2003. – 108 с.
15. Ляшко И.И., Макаров В.Л., Скоробагатько А.А. Методы вичислений. – К.: Вища школа, 1977, -408 с.
16. Мальцев А.М. Алгоритмы и рекурсивне функции. – М: Наука, 1986. – 368 с.
17. Макконнелл Дж. Основы современных алгоритмов –М.: Техносфера, 2004. – 368 с.
18. Марчук Г.И. Методы вычислительной математики. – М.: Наука, 1977. – 456 с.
19. Мозговой М.В. Классика программирования: языки, автоматы, компиляторы: практический поход. – Спб.: Наука и техника, 2006. – 320 с.
20. Самарский А.А. Введение в численные методы. – М.: Лань, 2009. – 288 с.
21. Самарский А.А., Николаев Е.С. Методы решения сеточных уравнений. – М.: Наука, 1978. – 592 с.
22. Сигорский В.П. Математический аппарат иженера. –К.: «Техніка», 1975, -783 с.
23. Словарь по кибернетике / [ред. академик В.М. Глушков]. – К.: Главная редакция украинской советской энциклопедии, 1979. – 624 с.
24. Фельдман Л.П., Петренко А.І., Дмитрієва О.А. Чисельні методи в інформатиці.- К.: Видавнича група BHV ,2006, - 480 с.
25. Хемминг Р.В. Численные методы. – М.: Наука, 1972. – 400 с.
26. Broder A., Charikar M. etc. Min-wise independent permutations // Journal of Computer and System Sciences, June 2000. – 60, Issue 3. – P. 630-659.
27. Chaitin J.G. Algorithmic information theory. – Yorktown Heights: IBM, 2003. – 234 с.

Алфавітний покажчик