

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КИЇВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Кафедра програмних систем і технологій

НІКОЛАЄНКО А. Ю.

КОМП'ЮТЕРНЕ МОДЕЛЮВАННЯ ПРОЦЕСІВ

Методичні вказівки до лабораторних занять
для здобувачів освітнього рівня «Бакалавр» зі спеціальності
121 – «Інженерія програмного забезпечення»

Київ – 2025

Рецензенти:

канд. техн. наук, доцент **Артамонов Є. Б.** – доцент кафедри інтелектуальних кібернетичних систем управління Державного некомерційного підприємства «Державний університет «Київський авіаційний інститут»

канд. техн. наук, доцент **Коваленко Є. Ю.** – доцент кафедри електронних пристроїв та систем Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського»

Рекомендовано до публікації в електронному вигляді кафедрою програмних систем і технологій Київського національного університету імені Тараса Шевченка, протокол №13 від 17.02.2025 р.

Рекомендовано до публікації в електронному вигляді Вченою радою факультету інформаційних технологій Київського національного університету імені Тараса Шевченка, протокол №13 від 19.05.2025 р.

Комп'ютерне моделювання процесів [Електронний ресурс]: методичні вказівки до лабораторних занять. Кафедра програмних систем і технологій / **А. Ю. Ніколаєнко** – К.: КНУ імені Тараса Шевченка, 2025. – 87 с.

Методичні вказівки до лабораторних занять з дисципліни «Комп'ютерне моделювання процесів» призначені для здобувачів третього курсу освітнього рівня «Бакалавр» денної форми навчання зі спеціальності 121 – «Інженерія програмного забезпечення» за освітньою програмою «Інженерія програмного забезпечення» галузі знань 12 – «Інформаційні технології».

Публікується в авторській редакції.

Електронна версія цього видання опублікована на сайті кафедри програмних систем і технологій за адресою: <https://pst.knu.ua/library> (дата публікації «27» травня 2025 р.)

© Ніколаєнко А. Ю., 2025 р.

ЗМІСТ

ВСТУП.....	5
Лабораторна робота №1 Основні поняття моделювання.....	8
Теоретичні відомості.....	8
Завдання на лабораторну роботу	10
Теми для дослідження.....	11
Цікаві задачі, які можна змодельовати:	13
Зміст звіту	14
Питання для самоконтролю	15
Лабораторна робота №2 Принципи побудови комп'ютерних моделей	16
Теоретичні відомості.....	16
Завдання на лабораторну роботу	23
Зміст звіту	23
Питання для самоконтролю	24
Лабораторна робота №3 Імітаційні моделі.....	25
Теоретичні відомості.....	25
Приклад розроблення імітаційної моделі	33
Завдання на лабораторну роботу	40
Варіанти завдань.....	40
Зміст звіту	45
Питання для самоконтролю	46
Лабораторна робота №4 Системи масового обслуговування.....	47
Теоретичні відомості.....	47
Приклад розроблення аналітичної моделі системи масового обслуговування	48
Завдання на лабораторну роботу	52
Варіанти завдань.....	53
Зміст звіту	58
Питання для самоконтролю	59
Лабораторна робота №5 Алгоритми класифікації.....	60

Теоретичні відомості.....	60
Приклад розроблення класифікатора.....	62
Завдання на лабораторну роботу	65
Варіанти завдань.....	67
Зміст звіту	68
Питання для самоконтролю	69
Лабораторна робота №6 Кластерний аналіз	70
Теоретичні відомості.....	70
Приклад кластерного аналізу даних.....	72
Завдання на лабораторну роботу	74
Варіанти завдань.....	75
Зміст звіту	76
Питання для самоконтролю	77
Лабораторна робота №7 Оптимізаційне моделювання	78
Теоретичні відомості.....	78
Завдання на лабораторну роботу	83
Варіанти завдань.....	83
Зміст звіту	85
Питання для самоконтролю	86
ЛІТЕРАТУРНІ ДЖЕРЕЛА	87

ВСТУП

Навчальна дисципліна «Комп'ютерне моделювання процесів» спрямована на вивчення основних понять та засобів моделювання, принципів побудови комп'ютерних моделей процесів, методів планування та проведення обчислювальних експериментів. Набуваються практичні вміння щодо математичної формалізації моделей процесів та їх реалізації в програмному коді.

Мета дисципліни – надати здобувачам освітнього рівня «Бакалавр» теоретичні підходи та відпрацювати практичні навички щодо моделювання процесів реального світу задля створення конкурентноздатного програмного забезпечення, знаходження оптимальних рішень та прогнозування їх наслідків в галузі інженерії програмного забезпечення.

Завдання дисципліни полягає у формуванні у здобувачів освіти здатності обґрунтовано вибирати засоби моделювання для того, щоб прогнозувати розвиток процесів реального світу, оцінювати та забезпечувати точність моделювання, перевіряти адекватність та коригувати досліджувані моделі, планувати та проводити обчислювальні експерименти, виконувати якісний та кількісний аналіз результатів моделювання, знаходити оптимальні шляхи вирішення практичних завдань і передбачувати їх наслідки в галузі інженерії програмного забезпечення.

Здобувач освіти має знати основні поняття та принципи програмування, теоретичні основи алгебраїчних, матричних і векторних обчислень; чисельні методи розв'язування лінійних і нелінійних рівнянь; методи інтерполяції, апроксимації та екстраполяції; основ оптимізації; методи чисельного диференціювання, рішення звичайних диференційних рівнянь і чисельного інтегрування. Здобувач освіти має вміти підбирати обчислювальні методи відповідно до особливостей математичної формалізації задачі, будувати алгоритми та програмувати розрахункові задачі.

В результаті вивчення дисципліни здобувач освіти має опанувати основні поняття і засоби моделювання, етапи і принципи побудови комп'ютерних

моделей, основні аспекти імітаційного моделювання, теоретичні основи методу статистичного моделювання, основні поняття та параметри систем масового обслуговування, основні методи оптимізації, основні алгоритми класифікації та кластеризації.

Перша лабораторна робота ознайомлює з основними поняттями моделювання: метою, об'єктом, предметом, завданнями та методами дослідження; гіпотезами, спрощеннями й обмеженнями моделі. У ході виконання цієї роботи здобувачі освіти опанують практичні навички розроблення концепції дослідження та побудови простої математичної моделі.

Друга лабораторна робота безпосередньо продовжує першу і присвячена створенню комп'ютерної моделі на основі розробленої математичної моделі, проведенню обчислювального експерименту, візуалізації та аналізу отриманих результатів.

У третій лабораторній роботі розглянуто основні етапи побудови імітаційної моделі, ідеї та прийоми статистичного моделювання, принципи дослідження випадкових процесів, а також методи збирання та аналізу статистичних даних.

Завдання четвертої лабораторної роботи спрямоване на дослідження системи масового обслуговування шляхом аналітичного та імітаційного моделювання. Особлива увага приділяється порівняльному аналізу результатів, отриманих за допомогою цих двох методів моделювання.

П'ята лабораторна робота ознайомлює з комплексним процесом розроблення моделей класифікації: від підготовки даних до впровадження алгоритмів класифікації, оцінювання продуктивності класифікатора та аналізу матриці помилок. У ході виконання цього завдання здобувачі освіти опанують практичні навички створення та вдосконалення класифікаторів з використанням бібліотеки Scikit-Learn, зосереджуючись на методах підвищення їхньої точності.

Шоста лабораторна робота присвячена вивченню методів кластерного аналізу з використанням бібліотеки Scikit-Learn. У рамках роботи здобувачі

освіти опановують алгоритм k-means та методологію визначення оптимальної кількості кластерів за допомогою різних метрик якості кластеризації.

У сьомій лабораторній роботі розглянуто чисельне знаходження локального мінімуму функції однієї змінної та функції двох змінних, що сприяє формуванню у здобувачів освіти навичок розв'язання оптимізаційних задач. Особлива увага приділяється побудові та аналізу карт ліній рівня функцій двох змінних, що надає візуальну інтерпретацію процесу оптимізації.

В результаті виконання лабораторних робіт здобувачі освіти мають набути практичні навички збирання та аналізування інформації про досліджуваний процес; визначення мети, об'єкта і предмета дослідження; розроблення концепції і формулювання завдань дослідження; математичного описання моделі; побудови блок-схеми алгоритму роботи моделі; створення комп'ютерної програми для симуляції поведінки досліджуваного процесу; перевірки коректності моделі та пошуку можливих шляхів виправлення незадовільних результатів.

Після проходження дисципліни здобувач освітнього рівня «Бакалавр» повинен вміти створювати комп'ютерні моделі з використанням відповідного програмного забезпечення, проводити комп'ютерні експерименти, збирати та аналізувати отримані результати, оцінювати адекватність моделі і робити висновки.

Лабораторна робота №1

Основні поняття моделювання

Мета – ознайомитися з основними поняттями моделювання, навчитися розробляти концепцію дослідження та будувати прості математичні моделі.

Теоретичні відомості

Моделлю називається спеціально створений об'єкт, який має необхідний для вивчення ступінь схожості вихідному об'єкту, адекватний цілям дослідження.

Моделі потрібні для того, щоб:

- зрозуміти сутність досліджуваного об'єкта: яка його структура, основні властивості, закони розвитку та взаємодії з навколишнім світом;
- навчитися керувати об'єктом чи процесом та визначити найкращі способи управління за заданих умов;
- прогнозувати прямі та опосередковані наслідки реалізації заданих методів і форм на об'єкт;
- розв'язувати прикладні завдання.

Мета дослідження – це очікуваний кінцевий результат. Вона визначає стратегію і тактику дослідження, його загальну спрямованість і логіку. Для одного й того самого об'єкта-оригіналу можна побудувати різні моделі залежно від мети дослідження.

Гіпотеза – припущення, яке висувають для пояснення досліджуваних явищ, перевіряють і за результатами перевірки підтверджують чи спростовують.

Залежно від мети вибирають об'єкт і предмет дослідження.

Об'єкт дослідження – це вибраний для вивчення процес або явище, що породжує проблемну ситуацію.

Предмет дослідження міститься в межах об'єкта. Це найбільш значущі властивості об'єкта, характеристики, функції чи взаємозв'язки, на які

безпосередньо має бути спрямовано дослідження (із зазначенням певних обмежень).

Об'єкт і предмет співвідносяться між собою як загальне і часткове. Предмет – це один із складових елементів об'єкта, саме той напрямок, який буде досліджуватися. Наприклад, тема «Модель розвитку ІТ-проєкту». Об'єктом дослідження є процес розвитку ІТ-проєкту. Предметом дослідження є моделі та методи прогнозування розвитку ІТ-проєкту з урахуванням ... (тут можна навести ключові характеристики ІТ-проєкту).

Визначення предмета дослідження практично є конкретизацією наукової проблеми, на основі якої формують **завдання дослідження**. Зазвичай послідовність самостійних завдань дають у вигляді переліку: *проаналізувати, розробити, узагальнити, виявити, обґрунтувати, довести, показати, описати, встановити, з'ясувати, визначити тощо*. Кожне з цих завдань стосується конкретного аспекту теми і підпорядковане меті.

Відповідно до мети і завдань дослідження визначають потребу в інформації, обґрунтовують набір показників, які характеризують об'єкт, здійснюють накопичення та узагальнення фактів, вибирають і ретельно аналізують джерела інформації. Тут під терміном інформація варто розуміти сукупність відомостей про об'єкт та предмет дослідження.

Важливим кроком у будь-якому дослідженні є правильний вибір методів і методик. **Метод** – це спосіб дослідження, інструмент для досягнення мети. **Методика** – правила використання конкретних методів.

Основні програмно-методологічні питання наукового дослідження можна сформулювати так:

- Мета – Для чого проводять дослідження?
- Об'єкт – Який процес досліджується?
- Предмет – Що саме у цьому процесі досліджують?
- Метод – Яким способом досліджують?
- Методика – Як використовують методи дослідження?

Гіпотеза та методи дослідження взаємопов'язані в науковому процесі. Гіпотеза визначає, які методи доцільно застосувати для її перевірки. Методи мають забезпечувати отримання даних, які можуть підтвердити або спростувати висунуте припущення. З іншого боку, результати застосування вибраних методів можуть змусити дослідника переглянути початкову гіпотезу. Правильний вибір методів, які відповідають характеру гіпотези, забезпечує достовірність результатів дослідження.

Завдання на лабораторну роботу

1. Вибрати тему для дослідження.
2. Провести аналіз предметної області.
3. Визначити мету, об'єкт та предмет дослідження згідно з вибраною темою.
Підказка: Уважно прочитайте спочатку теоретичні відомості до цієї роботи.
4. Висунути гіпотезу дослідження.
5. Сформулювати завдання дослідження. Зверніть увагу, що це не копія цього завдання на лабораторну роботу і це не технічне завдання на програму.
Підказка: першим пунктом завдань буде проаналізувати «щось», а останнім пунктом – розробити математичну модель «чогось».
6. Математично описати модель: навести всі потрібні формули, розшифрувати позначення використаних величин і вказати їх одиниці вимірювання.
Формули можна оформити будь-яким зручним способом, навіть просто вставити фото формул, написаних від руки у зошиті, або фрагменти з сайту.
 - Модель має включати нелінійні залежності між параметрами, що забезпечить можливість виявлення нетривіальних закономірностей у подальшому дослідженні.
 - Математична модель повинна містити функціональну залежність щонайменше від двох-трьох змінних параметрів, що надасть можливість

у наступній лабораторній роботі провести аналіз впливу кожного з них на досліджуваний процес.

7. Описати параметри моделі (вхідні величини, вихідні характеристики, діапазони змінних).
8. Обґрунтувати прийняті спрощення та описати обмеження моделі (*хоча б по 2 спрощення і по 2 обмеження*).
9. Сформулювати напрям подальшого дослідження: *визначити цільову змінну (що саме досліджуватиметься) та змінні параметри (які значення будуть змінюватися для аналізу їхнього впливу на цільову змінну)*.

Примітка: Пам'ятайте, що в першій лабораторній роботі дослідження завершується на етапі побудови математичної моделі. Комп'ютерна реалізація та експериментальне дослідження будуть виконані у другій лабораторній роботі. Тому особливу увагу приділіть вибору нелінійних залежностей і багатофакторних впливів у розроблюваній моделі, що забезпечить змістовність майбутнього дослідження.

Теми для дослідження

Найбільш простими темами для дослідження є моделі руху. Моделі руху належать до динамічних моделей, оскільки враховують залежність стану системи від часу. Стан моделі визначається точкою в координатному просторі. Точки, які відповідають різним моментам часу існування моделі, утворюють певну траєкторію. Для побудови динамічної моделі необхідно ввести систему координат та описати траєкторію руху тіла. Але моделювання самого руху – це не кінцевий етап роботи з моделлю: потрібно вміти від інформації, поданої знаковими конструкціями та графічними об'єктами (траєкторіями), перейти до інтерпретації моделі стосовно реальних об'єктів та зрозуміти її фізичний зміст.

Наприклад, під час дослідження польоту тіла, кинутого під кутом до горизонту, можна поставити множину питань і за допомогою моделі дослідити залежність вибраних змінних від початкових умов. Необхідно визначити початкові умови і вибрати досліджувані параметри. Наприклад: яка тривалість

польоту тіла залежно від початкової швидкості, кута кидання, маси тощо. Або яка дальність польоту? Або чому дорівнює максимальна висота підйому тіла за заданих початкових умов? Можна також дослідити момент часу, коли тіло перетне задану висотну позначку за конкретних початкових умов.

Приклади тем для дослідження

- Застосування законів збереження енергії та імпульсу в механічних явищах;
- Рух під дією кількох сил;
- Рух тіла, кинутого горизонтально та під кутом до горизонту;
- Рух тіла по похилій площині;
- Нитковий маятник і пружинний маятник;
- Пружні і непружні зіткнення;
- Гідродинамічні процеси в рідинах різної в'язкості;
- Поширення хвиль у неоднорідних середовищах;
- Оптичні явища в неоднорідних середовищах;
- Модель популяційної динаміки «хижак-жертва»;
- Поширення інфекційного захворювання в обмеженій популяції;
- Динаміка популяції комах залежно від кількості їжі та погодних умов;
- Ерозія ґрунту залежно від кількості опадів, крутизни схилу та типу рослинності;
- Кінетика хімічних реакцій із каталізаторами;
- Ціноутворення товару залежно від попиту та пропозиції;
- Динаміка заощаджень залежно від відсоткової ставки, інфляції та доходу;
- Поширення інформації в соціальній мережі залежно від кількості початкових джерел, активності користувачів та релевантності контенту;
- Ефективність алгоритму сортування залежно від розміру вхідних даних та їхньої початкової впорядкованості.

Цікаві задачі, які можна змоделювати:

1. Хлопець з дівчиною пішли у кафе випити кави з молоком. Коли їм принесли замовлення, то вони згадали, що треба помити руки. Тож вони одночасно пішли і через 5 хвилин одночасно повернулись. Питання: В кого кава буде холодніша, якщо дівчина додала молоко у каву перед тим, як йти мити руки, а хлопець додав молоко вже після повернення? Дослідити, чи впливає на відповідь температура молока і температура середовища.

2. Підводний човен знаходиться на глибині h від поверхні моря і рухається з деякою постійною горизонтальною швидкістю v . В момент часу $t = 0$ капітан човна дає наказ піднятися на поверхню ($h = 0$). Скласти математичну модель руху підводного човна.

3. Парашутист здійснює зтяжний стрибок. Залежно від заданих маси (від 50 до 100 кг), зросту (від 1,5 до 2 м), напівобхвату грудної клітини (від 0,3 до 0,55 м) і радіусу парашуту (від 2,2 до 3,4 м), визначити, починаючи з якого часу, після початку польоту швидкість парашутиста (з парашутом і без парашута) стає постійною. Дослідити залежність швидкості падіння і висоти польоту від часу.

4. Свинцева кулька масою m падає в маслі, зустрічаючи силу опору, пропорційну швидкості і силу гідростатичного виштовхування (силу Архімеда). Щільність свинцю $\rho_c = 11,3 \cdot 10^3 \text{ кг/м}^3$, щільність олії $\rho_m = 0,92 \cdot 10^3 \text{ кг/м}^3$, в'язкість олії $\mu_m = 0,95 \text{ кг/(м}\cdot\text{с)}$. Початкова швидкість v_0 спрямована по вертикалі. Дослідити залежність швидкості, прискорення та переміщення кульки від часу. Варіюючи v_0 та масу кульки встановити за допомогою комп'ютерної моделі характерні особливості руху кульки.

5. Куля, яка зроблена з кварцу, з радіусом r падає в невідомій речовині, зустрічаючи силу опору, пропорційну швидкості і силу гідростатичного виштовхування (силу Архімеда). Початкова швидкість 0 м/с, початкова висота h . Щільність кварцу $\rho_c = 2,21 \cdot 10^3 \text{ кг/м}^3$, щільність речовини ρ_m , в'язкість речовини μ_m . Дослідити залежність швидкості і часу падіння від щільності і в'язкості речовини.

6. Тіло кидають під кутом α до горизонту з початковою швидкістю від v_0 з початкової висоти h_0 . Дослідити залежність від початкових умов (швидкості і висоти) часу польоту тіла, дальності польоту тіла та його максимальної висоти підняття.

7. З човна масою M зробили постріл з гармати під кутом α до горизонту в напрямі руху човна. Маса снаряду m , початкова швидкість снаряду v_0 . Дослідити зміну швидкості човна від зміни кута пострілу та від зміни маси човна.

8. Літак виконує петлю Нестерова («мертву петлю») радіусом R і рухається по ній зі швидкістю v . З якою вагою пілот масою $m = 70$ кг тиснутиме на сидіння літака у нижній точці петлі? Дослідити залежність цієї ваги від радіуса петлі та швидкості літака.

9. Літак рухається горизонтально з надзвуковою швидкістю на висоті h над землею. Спостерігач чує звук через час τ секунд після того, як літак пролетів над ним. Дослідити залежність швидкості літака від висоти h та часу τ . Швидкість звуку взяти 330 м/с.

10. Снаряд масою m , який летів горизонтально із швидкістю v , попадає у візок з піском масою M і застрягає в піску. Візок до попадання снаряду рухався зі швидкістю V в напрямку руху снаряда. Дослідити зміну швидкості візка залежно від маси і швидкості снаряда.

Зміст звіту

1. Титульний аркуш;
2. Умова вибраного завдання;
3. Мета, об'єкт і предмет дослідження;
4. Гіпотеза дослідження;
5. Завдання дослідження (орієнтовно 4-5 пунктів – «що зробити?»);
6. Математичний опис розробленої моделі;
7. Опис параметрів моделі;
8. Спрощення та обмеження моделі;
9. Напрямок подальшого дослідження.

Питання для самоконтролю

1. Що таке модель і яке її основне призначення?
2. Для чого потрібні моделі?
3. Поясніть різницю між об'єктом і предметом дослідження.
4. Що таке мета дослідження і як вона впливає на побудову моделі?
5. Чому важливо чітко визначати предмет дослідження на початковому етапі моделювання?
6. Що таке гіпотеза і яка її роль у процесі моделювання?
7. Що таке метод дослідження і чому важливий правильний вибір методів?
8. Чим методика відрізняється від метода?
9. Чому для одного й того ж об'єкта можуть розроблятися різні моделі?
10. Чи може модель бути корисною, навіть якщо не всі гіпотези підтверджуються?

Лабораторна робота №2

Принципи побудови комп'ютерних моделей

Мета – набути практичних навичок створення комп'ютерної моделі, проведення обчислювального експерименту та аналізу отриманих результатів.

Теоретичні відомості

Основні принципи побудови комп'ютерних моделей:

1. Цілеспрямованість – модель має будуватися з урахуванням мети дослідження процесу чи явища.
2. Інформаційна достатність – необхідно мати оптимальний баланс знань про об'єкт-оригінал – достатній для побудови моделі, яка відображає його істотні властивості, але неповний, щоб зберегти доцільність самого моделювання.
3. Спрощення – модель має бути спрощеним відображенням оригіналу з відкиданням несуттєвих для конкретного дослідження властивостей, зберігаючи при цьому ключові характеристики об'єкта.
4. Деталізація – необхідно забезпечити оптимальний баланс між точністю результатів моделювання та складністю моделі, враховуючи обчислювальні ресурси та часові обмеження.
5. Агрегування – організація моделі у вигляді ієрархічної структури з відносно автономними блоками та чітко визначеними інтерфейсами між ними, що відображає етапи та режими функціонування досліджуваної системи.
6. Параметризація – побудова моделі у вигляді системи з відомою структурою, але невідомими параметрами, які потребують ідентифікації на основі експериментальних даних.
7. Адекватність моделі – перевірка відповідності моделі об'єкту-оригіналу за визначеною дослідником множиною властивостей і характеристик у межах допустимої похибки.
8. Документування – систематичне та повне документування всіх вхідних даних, досліджуваних характеристик об'єкта, використаних методів, припущень та

обмежень, а також отриманих результатів моделювання для забезпечення відтворюваності та подальшого аналізу.

Процес розроблення моделей та їх дослідження на комп'ютері можна розділити на кілька основних етапів (рис. 2.1).



Рисунок 2.1 – Основні етапи моделювання

Візуалізація результатів моделювання полегшує сприйняття отриманих даних, їх інтерпретацію та аналіз. Для репрезентації результатів використовують різноманітні засоби графічного представлення: графіки, діаграми, гістограми, схеми, таблиці тощо.

PrettyTable – це спеціалізована бібліотека Python для форматowanego виведення табличних даних у текстовому форматі. Дана бібліотека забезпечує

представлення числових та текстових результатів моделювання у вигляді структурованих, читабельних таблиць з можливістю виведення в консоль або збереження в текстовий файл (рис. 2.2).

```
import numpy as np
from prettytable import PrettyTable

n = np.arange(1, 9, 1)
X = 10 + 2 * n
Y = 3 * X + np.random.normal(5, 10, X.size)

table = PrettyTable()

table.add_column("#", n)
table.add_column("X", X)
table.add_column("Y", Y)
table.float_format = '.3'
print(table)
```

```
>>> +---+---+---+
| # | X | Y |
+---+---+---+
| 1 | 12 | 37.206 |
| 2 | 14 | 40.468 |
| 3 | 16 | 52.426 |
| 4 | 18 | 46.507 |
| 5 | 20 | 64.479 |
| 6 | 22 | 53.457 |
| 7 | 24 | 74.457 |
| 8 | 26 | 86.666 |
+---+---+---+
```

Рисунок 2.2 – Приклад використання бібліотеки PrettyTable для виведення табличних даних

У Python існує широкий спектр бібліотек для візуалізації результатів досліджень, серед яких найбільш значущими є Matplotlib, Seaborn, Plotly, Bokeh, Altair та Pygal. Це лише невеликий перелік найпопулярніших інструментів візуалізації даних. Вибір конкретної бібліотеки визначається специфікою дослідницьких завдань, особливостями даних та вимогами до кінцевого представлення результатів.

Бібліотека Matplotlib надає можливості для створення різноманітних типів візуалізацій: графіків, діаграм, скатерограм, теплових карт, 3D-проекцій тощо. Seaborn, будучи надбудовою над Matplotlib, розширює функціональність базової бібліотеки, пропонуючи високорівневий інтерфейс для створення статистично обґрунтованих та естетично довершених візуалізацій з мінімальними зусиллями завдяки оптимізованим налаштуванням за замовчуванням.

Хоча сучасні бібліотеки візуалізації надають потужні інструменти, створення ефективних візуалізацій не є таким простим, як може здаватися на перший погляд. Основні складнощі виникають не у використанні самих бібліотек, а на етапі вибору найбільш підходящого типу графіка та його оформлення для конкретних даних та дослідницьких завдань.

Важливо розуміти, що складніший графік не завжди означає кращу візуалізацію. Наприклад, якщо досліджується залежність між трьома змінними, не обов'язково будувати 3D-графік. Часто більш ефективними можуть бути альтернативні способи представлення: теплові карти, графіки з використанням кольору для позначення третьої змінної, або набір простіших 2D-графіків. Збільшення кількості вимірів у візуалізації часто ускладнює сприйняття інформації замість того, щоб її прояснити.

Приклад некоректної візуалізації експериментальних даних

У процесі дослідження закону Стокса студент отримав дані на моделі та спочатку подав їх у вигляді тривимірного графіка (рис. 2.3).

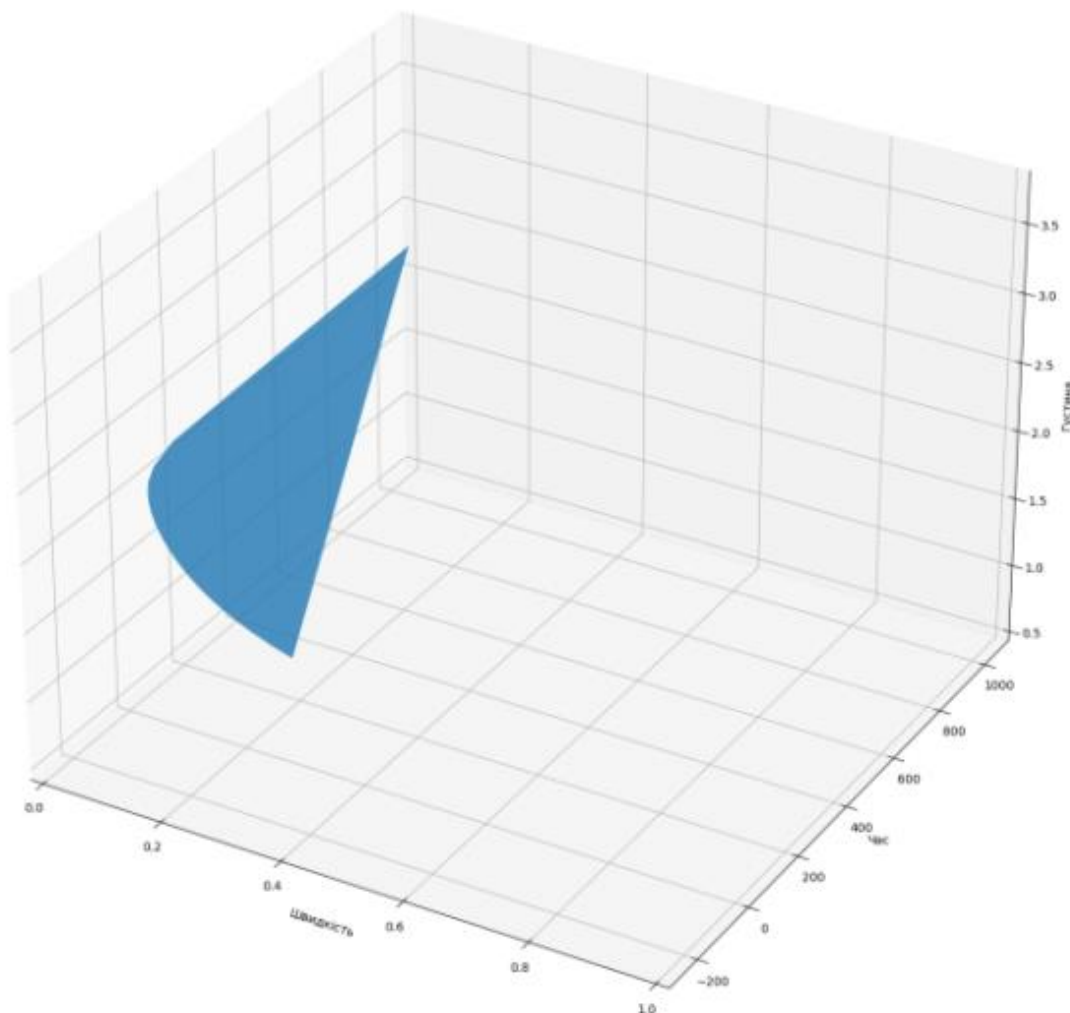


Рисунок 2.3 – Приклад некоректно побудованого 3D-графіка результатів дослідження закону Стокса

Незадоволений першим результатом, він змінив кут огляду графіка (рис. 2.4), вибравши таке розташування координатних осей, яке наближає візуалізацію до двовимірного представлення.

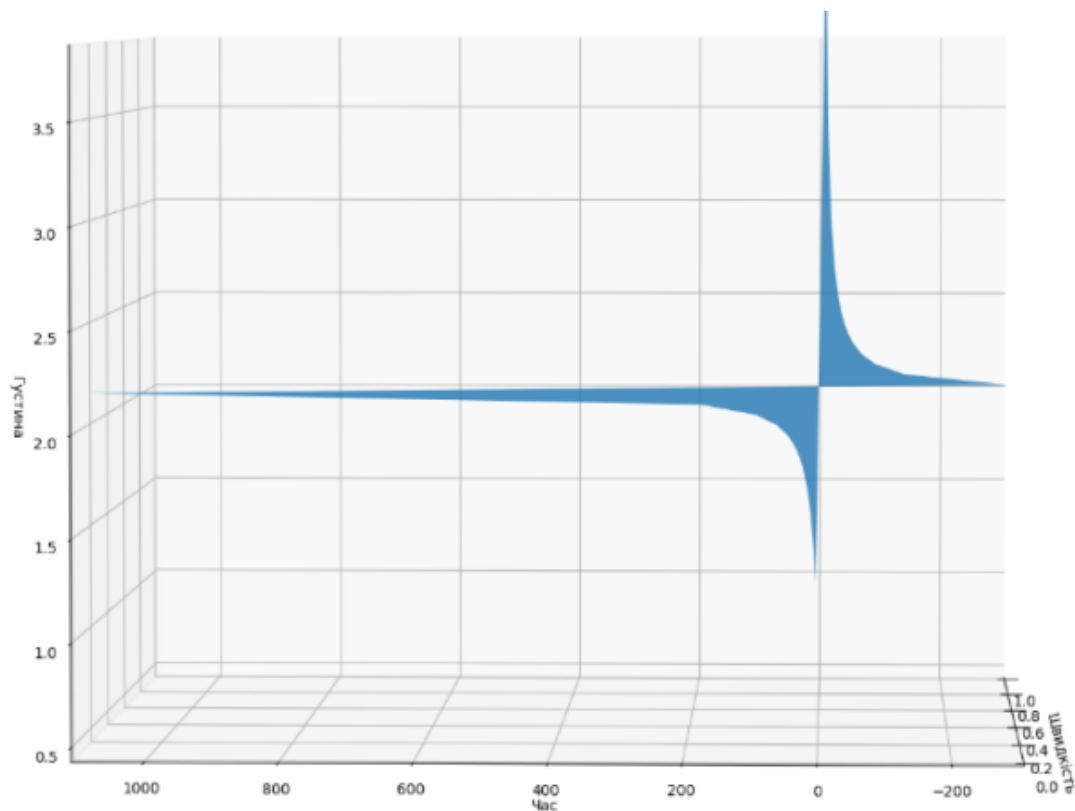


Рисунок 2.4 – Приклад некоректно побудованого 3D-графіка (вигляд з іншого ракурсу)

Отриманий графік є не тільки некоректним, а й може призвести до хибної інтерпретації результатів дослідження. У поставленій задачі густина речовини не змінюється з часом, тому відображення її як динамічної величини в часовій перспективі є фізично необґрунтованим. Правильний підхід до візуалізації таких даних полягає у відображенні часової динаміки швидкості падіння тіла як функції від густини середовища. Доцільно для кожного значення густини побудувати окремий двовимірний графік зміни швидкості тіла в часі. Така серія графіків точніше відобразить фізичну сутність закону Стокса та забезпечить коректну інтерпретацію результатів експерименту.

Представлення функцій багатьох змінних за допомогою 2D-графіків

2D-графіки зазвичай забезпечують кращу наочність та спрощують аналіз даних. Традиційно вважається, що 2D-графік призначений для відображення функції однієї змінної. Виникає питання: яким чином ефективно представити функцію двох, трьох чи більшої кількості змінних?

Розв'язання цієї задачі можливе через фіксацію значень усіх змінних, крім однієї, на певних сталих рівнях. У такому випадку отримуємо одну криву, яка відображає залежність результату від цієї єдиної змінної. Якщо ж послідовно змінювати значення однієї з попередньо зафіксованих змінних з визначеним кроком і будувати відповідні криві, то отримаємо сімейство функціональних залежностей. На рис. 2.5 показано тривимірну поверхню, «нарізану» площинами з різними значеннями Y , що створює сімейство кривих.

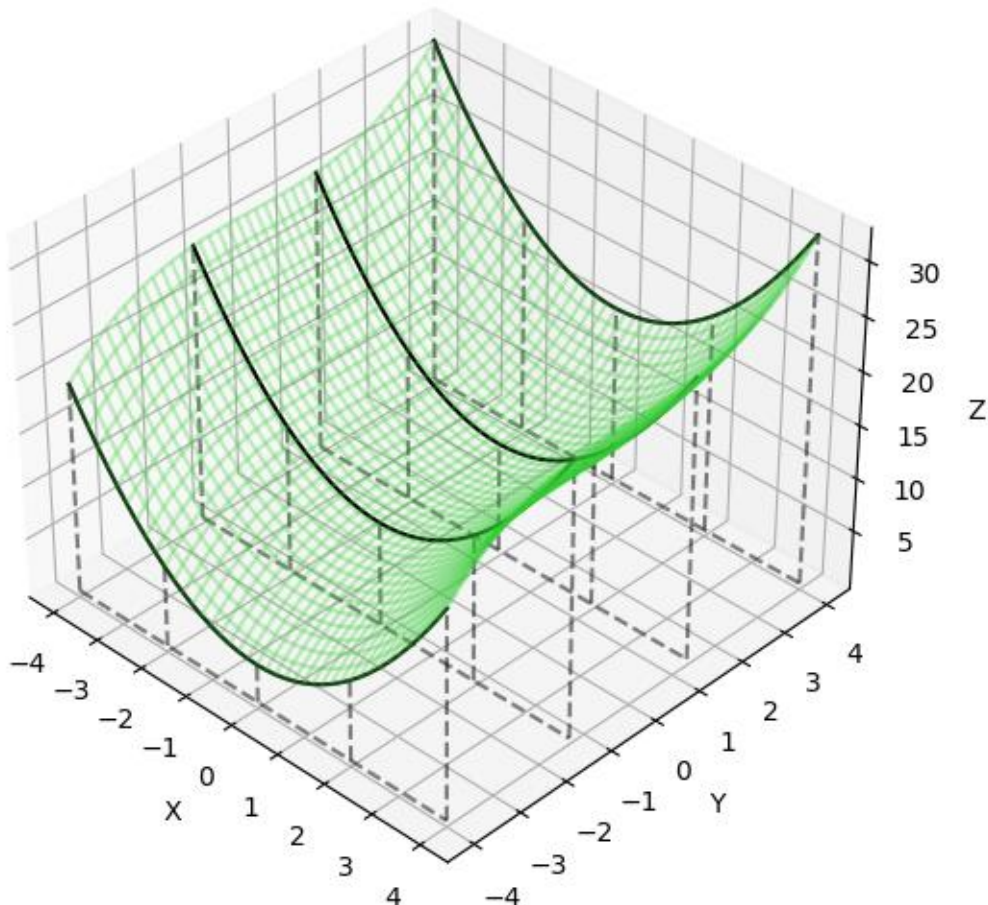


Рисунок 2.5 – Сімейство кривих у тривимірному просторі (зміна значення Y з деяким кроком).

Надалі ці криві можна проєктувати на двовимірну площину, отримуючи сімейство ліній на одному графіку. Такий підхід дає змогу ефективно візуалізувати результати дослідження функції багатьох змінних за допомогою двовимірних представлень. На рис. 2.6 показано результати моделювання траєкторії руху мінометного снаряда у вигляді сімейства кривих на 2D-графіку. Студент продемонстрував залежність максимальної висоти і дальності польоту снаряда від кута пострілу.

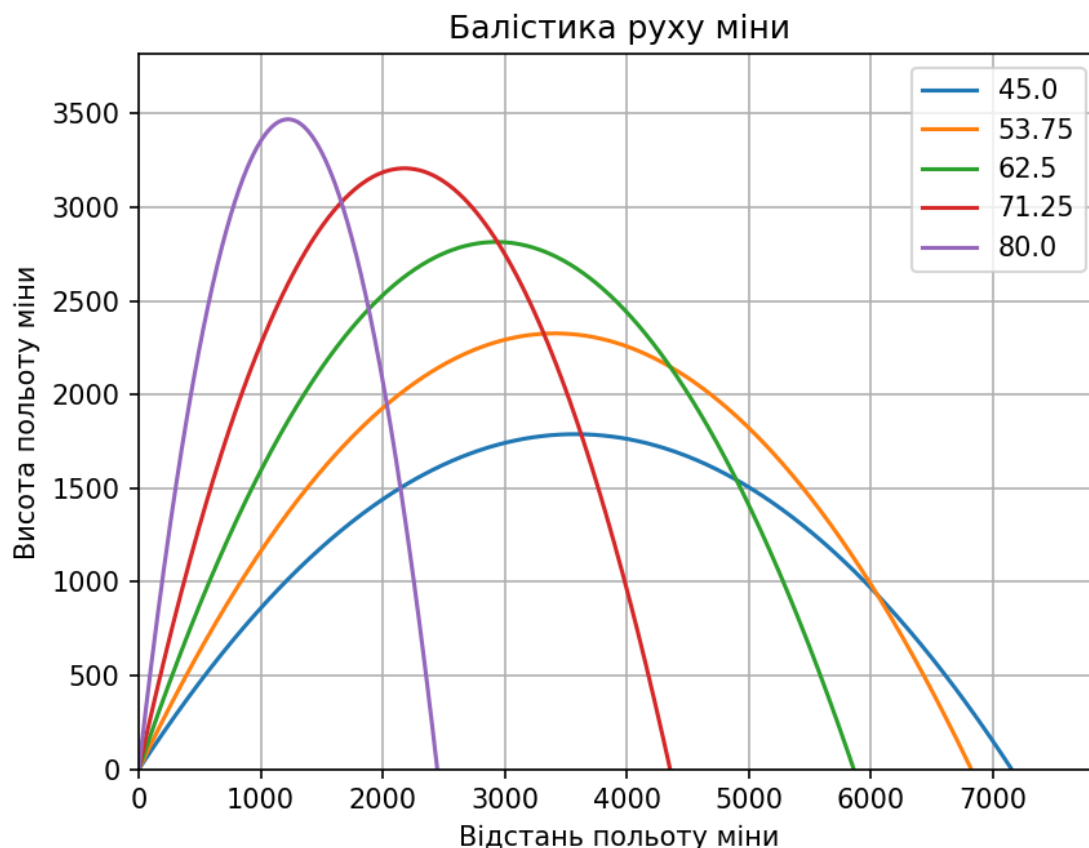


Рисунок 2.6 – Сімейство кривих у двовимірному просторі (є недоліки в оформленні – нема одиниць вимірювання і не оформлена легенда).

Побудований графік є інформативним і легким для аналізу. Однак слід зазначити недоліки в його оформленні: відсутні одиниці вимірювання висоти і відстані (імовірно, метри), а також недостатньо чітко оформлена легенда, в якій не вказано, що параметром сімейства кривих є саме кут вильоту снаряда.

Вибір способу візуалізації результатів моделювання визначається специфікою дослідницького завдання та характером отриманих даних. Хоча в окремих випадках тривимірні графіки можуть бути оптимальним рішенням, для

більшості практичних задач 2D-графіки забезпечують кращу інтерпретацію результатів та ефективність сприйняття.

Завдання на лабораторну роботу

1. Побудувати алгоритм роботи математичної моделі, розробленої в лабораторній роботі №1. Алгоритм зобразити у вигляді блок-схеми з використанням стандартних позначень.
2. Розробити програмний код комп'ютерної моделі.
3. Провести комп'ютерні експерименти, візуалізувати та проаналізувати отримані результати. Подати результати дослідження у різних формах:
 - не менше 2х різних графіків з сімейством кривих (3-5 кривих на кожному), які відображають залежність вихідних показників моделі від вхідних.
 - діаграма для подання результатів у альтернативній формі,
 - таблиця з підсумковими результатами;Забезпечити всі графічні матеріали необхідними позначеннями: назви осей, легенда, заголовок, одиниці вимірювання.
4. Перевірити адекватність моделі. Провести верифікацію моделі шляхом порівняння з теоретичними розрахунками. За можливості виконати валідацію моделі за допомогою порівняння з наявними даними (якщо доступні);
5. Зробити висновки, які відображають критичний аналіз результатів виконання роботи.

Зміст звіту

- 1-9. *Продовжити звіт лабораторної роботи №1.*
10. Блок-схема алгоритму реалізації комп'ютерної моделі;
11. Код програми (може бути оформлений як додаток або прикріплений як окремий текстовий файл чи архів, якщо файлів багато);

12. Результати моделювання;
13. Аналіз адекватності моделі;
14. Висновки.

Питання для самоконтролю

1. Як визначається потреба в інформації для моделювання?
2. Чому модель повинна бути простішою за об'єкт дослідження?
3. Яким чином обмеження моделі впливають на інтерпретацію отриманих результатів?
4. Як надмірне спрощення моделі може вплинути на точність результатів дослідження?
5. Чи завжди найскладніша модель є найкращим рішенням?
6. Як можна перевірити адекватність розробленої моделі?
7. У яких випадках доцільно використовувати кілька взаємодоповнюючих моделей для дослідження одного об'єкта?
8. Чому важливо документувати всі вхідні дані та результати моделювання?
9. Яка головна проблема при використанні 3D-графіків для візуалізації даних?
10. Які проблеми можуть виникнути при неправильному виборі способу візуалізації результатів моделювання?

Лабораторна робота №3

Імітаційні моделі

Мета – ознайомитися з етапами побудови імітаційної моделі, прийомами статистичного моделювання та методами дослідження випадкових процесів.

Теоретичні відомості

Імітаційні моделі – це моделі прогінного типу, які мають вхід і вихід. Якщо подати на вхід моделі певні значення параметрів, можна отримати відповідний їм результат.

Імітаційне моделювання базується на математичному методі статистичних випробувань, який передбачає моделювання і дослідження на комп'ютері процесу функціонування системи з урахуванням випадкових процесів або випадкових впливів. Для нових значень параметрів чи взаємозв'язків імітаційну програму потрібно запустити знову, тобто «прогнати». **Експеримент** на імітаційній моделі проводиться для того, щоб отримати інформацію про функціонування досліджуваної системи.

Динаміка реалізується за допомогою механізму перебігу **модельного часу**. Для цього вводять глобальну змінну, яка забезпечує синхронізацію всіх подій у системі. Існують два основних способи зміни модельного часу: покроковий – з фіксованими інтервалами його зміни, і подієвий, за якого величина кроку визначається змінним інтервалом до настання наступної події.

У процесі моделювання відбуваються зміни модельного часу, який найчастіше приймається дискретним. Час змінюється після того, як закінчена імітація чергової групи подій, які відносяться до наступного моменту часу. Імітація закінчується, коли поточний час перевищить заданий відрізок часу або коли буде досягнуто певних граничних умов системи (наприклад, вичерпано обмежені ресурси чи накопичено необхідний обсяг вихідних даних).

Імітація супроводжується **збиранням статистики** змінних системи. Метод імітаційного моделювання полягає у багаторазовому розрахунку (прогоні)

процесів функціонування системи (рис. 3.1). Після серії прогонів здійснюють оброблення та аналіз накопичених статистичних даних. За потреби параметри моделі коригуються для досягнення бажаних результатів.

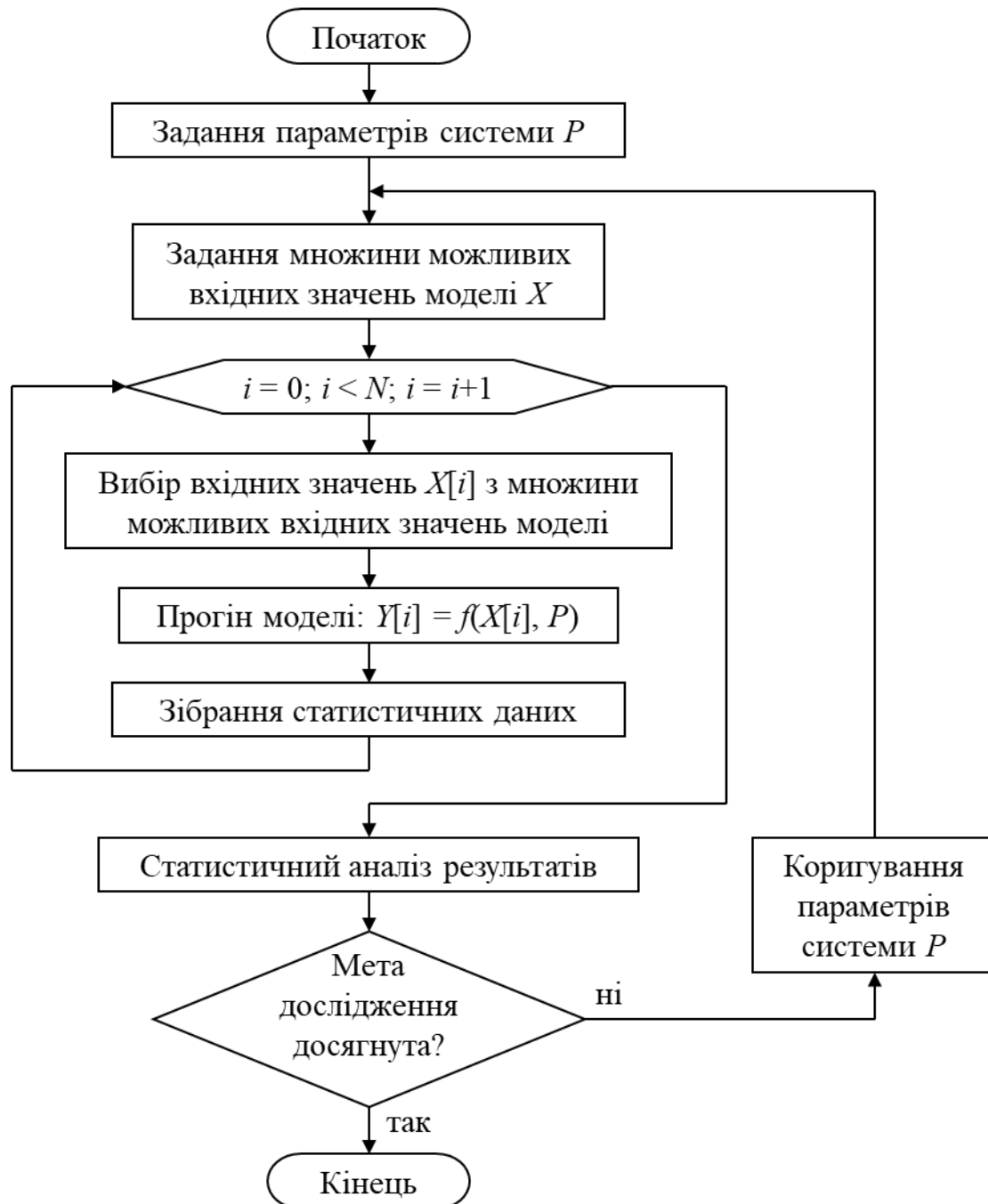


Рисунок 3.1 – Блок-схема типового процесу імітаційного моделювання

На рисунку прийняті такі позначення: P – параметри системи, X – вхідні змінні, Y – вихідні змінні, f – функція відгуку системи, i – лічильник прогонів, N – загальна кількість прогонів.

Для побудови імітаційних моделей зазвичай застосовують спеціальні мови імітаційного моделювання, але можна використовувати і мови програмування загального призначення. Як правило, кожен елемент моделі у програмному коді описується певною процедурою (підпрограмою). Тоді модель можна подати як алгоритм, що складається з упорядкованих звернень до цих процедур, які відображають інформацію щодо поведінки системи-оригіналу.

Для створення імітаційної моделі на Python доцільно використовувати бібліотеки `pandas`, `numpy` і `matplotlib`.

Бібліотека Pandas

Pandas – це бібліотека для обробки та аналізу даних на Python. Вона значно підвищує ефективність роботи аналітиків: операції, які раніше потребували десятків рядків коду, тепер можна записати одним рядком. Для використання бібліотеки необхідно спочатку встановити її, а потім імпортувати в проект:

```
import pandas as pd
```

У `pandas` існує два основні типи структур даних: `Series` та `DataFrame`.

Series – це одновимірна структура даних, яка нагадує вектор чи список, але з додатковими функціональними можливостями.

DataFrame – це двовимірна структура даних, яка складається зі стовпців та рядків. Стовпці мають назви, а рядки позначаються індексами. Важливо розуміти, що кожен стовпець в об'єкті `DataFrame` є об'єктом `Series`, елементи якого формують значення рядків.

Загальний формат класу **Series**:

```
class pandas.Series(data=None, index=None, dtype=None, name=None, copy=False, fastpath=False)
```

де `data` – масив (словник, список, серія) даних; `index` – одновимірний масив, значення якого повинні бути хешованими і мати таку ж довжину, що і дані; `dtype` – необов'язковий параметр, який визначає тип даних для вихідної серії (може бути рядковим, `numpy.dtype` або `ExtensionDtype`); `name` – необов'язковий

параметр рядкового типу, який визначає ім'я серії; copy – логічний тип (за замовчуванням false), який вказує, чи потрібно копіювати вхідні дані.

Приклад створення об'єкта Series та виведення на екран:

```
import numpy as np
import pandas as pd
S1=pd.Series(data=np.arange(10, 70, 10),
             index=['x1','x2','x3','x4','x5','x6'])
print(S1)
```

>>

x1	10
x2	20
x3	30
x4	40
x5	50
x6	60

 dtype: int32

Загальний вигляд класу [DataFrame](#):

```
class pandas.DataFrame(data=None, index=None, columns=None, dtype=None,
copy=None)
```

де data – масив даних (словник, список) або інший DataFrame; index – індекс або одновимірний масив для позначення рядків; columns – індекс або одновимірний масив для позначення стовпців; dtype – тип даних елементів DataFrame; copy – логічний параметр, який визначає, чи потрібно копіювати вхідні дані.

Об'єкт DataFrame має два індекси: для рядків та для стовпців. Якщо індекс для рядків або стовпців не заданий, то pandas автоматично встановлює цілочисельний індекс RangeIndex від 0 до $n-1$, де n – кількість стовпців або рядків в таблиці відповідно. Для доступу до рядків за текстовою міткою використовується [.loc](#), для доступу за числовим індексом (починаючи з 0) – [.iloc](#). У DataFrame можливе створення, видалення та перейменування стовпців (детальніше за [посиланням](#)) і рядків (детальніше за [посиланням](#)).

Нижче наведено приклад створення об'єкта DataFrame, заповнення його випадковими цілими числами з рівномірним розподілом, використання методу `.sum()` для підсумовування значень у кожному стовпці/рядку таблиці, додавання підсумкового стовпця (gsl) і використання `.loc` для додавання підсумкового рядка в таблицю:

```

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

month_abbr=['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
            'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']
count_month=len(month_abbr)
test_name=['x1', 'x2', 'x3', 'x4', 'x5', 'x6']
count_test=len(test_name)
test=np.random.randint(10, 30, size=(count_month, count_test))

S_df=pd.DataFrame(data=test, index=month_abbr, columns=test_name)
S_df.loc['Total']=S_df.sum()      #Total sum per column
S_df['Total']=S_df.sum(axis=1)    #Total sum per row
print(S_df)

```

Результат виведення об'єкта DataFrame на екран:

	x1	x2	x3	x4	x5	x6	Total
Jan	16	24	11	11	20	11	93
Feb	23	26	12	25	14	24	124
Mar	24	16	13	27	25	26	131
Apr	26	20	19	22	21	10	118
May	27	17	11	20	12	27	114
Jun	10	28	15	28	19	26	126
Jul	26	21	16	11	25	26	125
Aug	17	27	14	20	24	15	117
Sep	20	14	14	18	25	27	118
Oct	16	19	23	11	10	29	108
Nov	25	15	24	20	20	23	127
Dec	16	16	14	25	14	16	101
Total	246	243	186	238	229	260	1402

Основні статистичні параметри кожного стовпця DataFrame можна переглянути за допомогою методу `.describe()`. Застосуємо цей метод до всіх значень таблиці, крім підсумкового рядка і стовпця.

```
print(S_df.iloc[:-1,:-1].describe())
```

	x1	x2	x3	x4	x5	x6
count	12.000000	12.000000	12.000000	12.000000	12.000000	12.000000
mean	20.500000	20.250000	15.500000	19.833333	19.083333	21.666667
std	5.435573	4.938255	4.337993	6.132378	5.384461	6.746492
min	10.000000	14.000000	11.000000	11.000000	10.000000	10.000000
25%	16.000000	16.000000	12.750000	16.250000	14.000000	15.750000
50%	21.500000	19.500000	14.000000	20.000000	20.000000	25.000000
75%	25.250000	24.500000	16.750000	25.000000	24.250000	26.250000
max	27.000000	28.000000	24.000000	28.000000	25.000000	29.000000

Pandas у поєднанні з matplotlib надає можливість будувати графіки. Достатньо імпортувати модуль `matplotlib.pyplot` та викликати метод `plot()` на об'єкті DataFrame. Нижче наведено приклад виведення кількох графіків на одне полотно (layout):

```

fig = plt.figure()
ax1 = fig.add_subplot(2, 2, (1,2))
plt.title('DataFrame except sums')
plt.xlabel('month')
plt.ylabel('test')

ax2 = fig.add_subplot(2, 2, 3)
plt.title('Last column of DataFrame')
plt.xlabel('month')
plt.ylabel('Total sum per row')

ax3 = fig.add_subplot(224)
plt.title('Last row of DataFrame')
plt.xlabel('test_name')
plt.ylabel('Total sum per column')

S_df.iloc[:,-1].plot(kind="bar", ax=ax1, figsize=(8, 6))
S_df.iloc[:,-1:].plot(kind="bar", ax=ax2, legend=None)
S_df.iloc[-1:,-1].T.plot(kind="bar", ax=ax3, legend=None)
ax1.legend(bbox_to_anchor=(1.01, 1), loc="upper left", borderaxespad=0)
plt.tight_layout(pad=1)
plt.show()

```

Індексація у `add_subplot(m, n, p)` визначає положення підграфіка у сітці розміром $m \times n$. Параметр *p* – порядковий номер підграфіка, який нумерується зліва направо і зверху вниз, починаючи з 1. Зверніть увагу, що `add_subplot(mnp)` є альтернативною (скороченою) формою запису `add_subplot(m, n, p)`. Параметр *p* може бути не тільки одним числом, а й кортежем або злитим діапазоном чисел (наприклад, `add_subplot(2, 2, (1, 2))`). Це дає змогу об'єднувати кілька комірок сітки в один підграфік, який займає відповідний простір у фігурі.

У наведеному прикладі побудовано три стовпчасті діаграми (рис. 3.2):

- 1) діаграма всієї таблиці, крім підсумкового рядка і стовпця (ax1, верхній subplot),
- 2) діаграма лише підсумкового стовпця (ax2, нижній лівий subplot),
- 3) діаграма лише підсумкового рядка (ax3, нижній правий subplot).

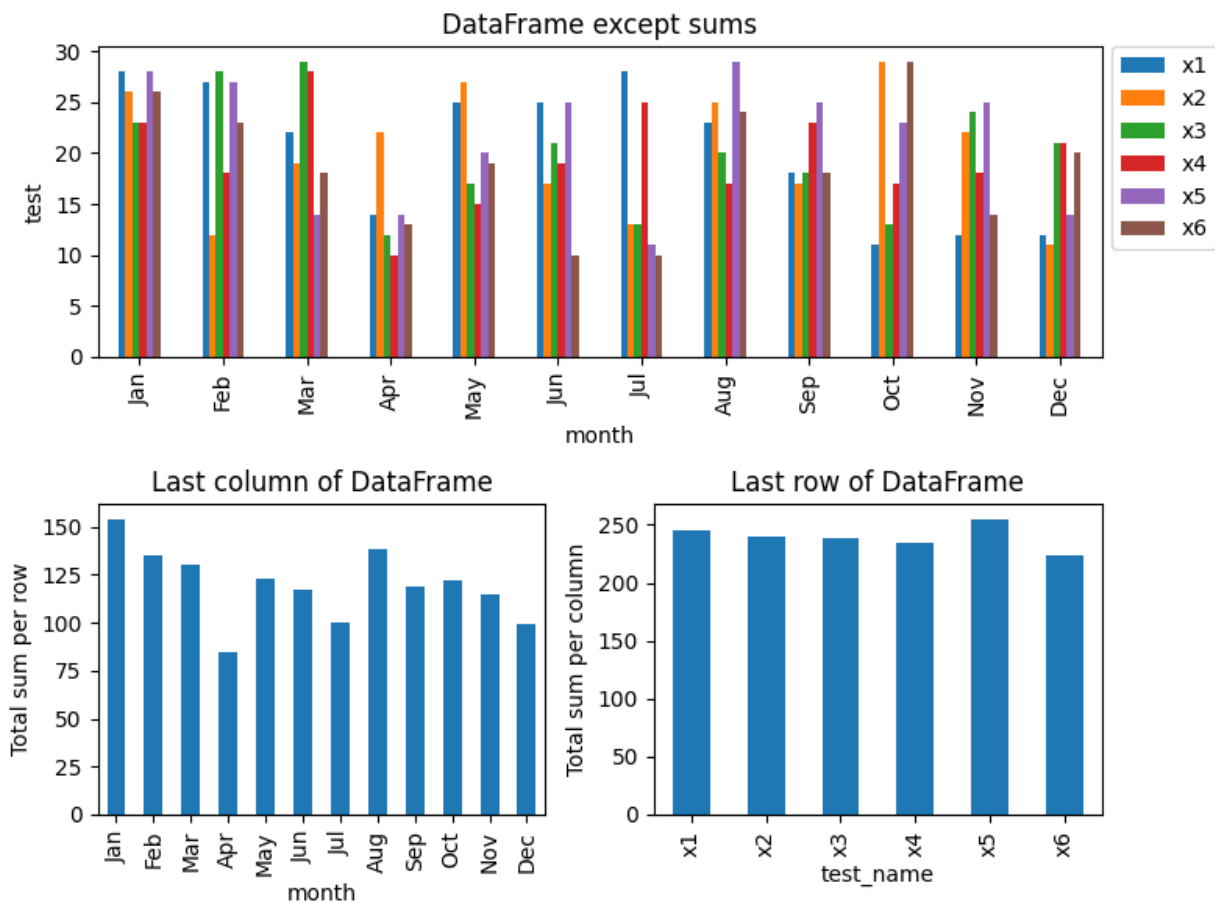


Рисунок 3.2 – Приклад візуалізації даних об'єкта DataFrame

Кожен графік має назву і підписи по осях. Легенда першого графіка винесена за межі області вліво.

Закони розподілу

Імітаційне моделювання включає ідеї та методи статистичного моделювання на комп'ютері, дослідження стохастичних систем і випадкових процесів.

Основні закони розподілу було розглянуто на лекції 4. Однією з форм закону розподілу як дискретних, так і неперервних випадкових величин є функція розподілу $F(x)$. Значення функції розподілу належать відріzkу $[0; 1]$: $0 \leq F(x) \leq 1$.

Законом розподілу будь-якої дискретної величини називається співвідношення, що визначає залежність між значеннями випадкової величини

та ймовірностями, з якими ці значення набуваються. Закон розподілу можна задати у вигляді таблиці, аналітично (у вигляді формули) та графічно.

Однією з форм закону розподілу, що існує тільки для неперервних випадкових величин, є щільність розподілу $f(x)$, яка дорівнює першій похідній функції розподілу $F(x)$. Щільність розподілу $f(x)$ ще називають диференціальною функцією розподілу. Важлива властивість щільності розподілу полягає в тому, що інтеграл від неї в нескінченних границях дорівнює одиниці.

Для генерації псевдовипадкових чисел у Python можна використовувати модуль Random бібліотеки Numpy. Цей модуль містить численні методи для створення випадкових чисел з різними законами розподілу.

Рівномірний розподіл: [`numpy.random.rand\(\)`](#) для генерації чисел (або масиву чисел) з плаваючою точкою в діапазоні $[0, 1)$ та [`numpy.random.randint\(\)`](#) для генерації цілих чисел у заданому діапазоні.

Нормальний розподіл – [`numpy.random.randn\(\)`](#) для генерації чисел з плаваючою точкою з нормальним розподілом.

Розподіл Пуассона – [`numpy.random.poisson\(\)`](#) – це дискретний розподіл ймовірностей, який моделює частоту певних подій протягом фіксованого інтервалу часу на основі середньої частоти виникнення цих подій. Застосовується для моделювання кількості випадків, коли подія відбувається протягом фіксованого періоду часу (наприклад, кількість відвідувань певного закладу/сайту за добу) або в певній області простору (наприклад, кількість дефектів тканини на квадратний метр)

Трикутний розподіл – [`numpy.random.triangular\(\)`](#) – це неперервний розподіл ймовірностей з нижньою межею зліва, піком у моді та верхньою межею праворуч. Ці параметри безпосередньо визначають форму щільності закону розподілу.

Гамма-розподіл – [`numpy.random.gamma\(\)`](#) визначається двома невід’ємними параметрами: `shape` – число ступенів свободи та `scale` – коефіцієнт масштабу. За цілих значень ступенів свободи гамма-розподіл називається розподілом **Ерланга**.

Детальнішу інформацію про функції отримання випадкових чисел з використанням різних розподілів можна знайти в офіційній документації NumPy: <https://numpy.org/doc/stable/reference/random/legacy.html#distributions>

Приклад розроблення імітаційної моделі

Під час проектування системи потрібно визначити загальне середовище, в якому відбуваються події, елементи системи, на які ці події впливають, і всі події, які тільки можуть статися в системі. Наприклад, загальним середовищем може бути банк, магазин, фірма, автозаправна станція, завод тощо. Загальним середовищем може бути і частина якоїсь більшої системи, як-от каса магазину чи технологічна лінія заводу.

Для прикладу виберемо загальним середовищем автозаправну станцію (АЗС). Машини на АЗС заправляють паливом. Насправді послуги АЗС можуть бути ширшими, але у прикладі розглянемо лише одну з її функцій – заправлення машин паливом. Визначимо елементи системи, які для цього потрібні: машини (клієнти АЗС), резервуар для зберігання пального (будемо вважати, що він один, об'єм 10 000 літрів), паливо (для спрощення тільки один вид), паливороздавальні колонки (для спрощення тільки одна колонка), місце для черги автомобілів (не обмежено), графік роботи (цілодобово). Вважаємо, що АЗС максимально автоматизована – водії самі обслуговують свої автомобілі та сплачують карткою за паливо, тож не враховуємо касирів і працівників, які наливають паливо в бак.

Для побудови моделі необхідно визначити характеристики елементів системи. На основі зібраних даних (у цьому прикладі вигаданих) про функціонування цієї АЗС протягом тривалого часу встановлено:

- Кількість машин, які приїжджають на АЗС, є випадковою величиною, розподіленою за законом Пуассона: з 07:00 до 22:00 – математичне сподівання 7 автомобілів на годину, з 22:00 до 07:00 – математичне сподівання 2 автомобілі на годину.

`numpy.random.poisson(Lam, size)`

- Кількість пального, яке замовляє одна машина – випадкова величина з трикутним розподілом: нижня межа – 1 літр, верхня межа – 70 літрів, мода – 30 літрів.

`numpy.random.triangular(left, mode, right, size)`

- Тривалість заправлення одного автомобіля не залежить від об'єму замовленого пального і є випадковою величиною з логонормальним законом розподілу: $\mu = 2$, $\sigma = 0,1$.

`numpy.random.lognormal(mu, sigma, size)`

Для побудови імітаційної моделі виберемо крок моделювання 1 година.

Модель використовуватиметься для аналізу роботи АЗС впродовж 1 доби.

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import datetime

step=1
count_step=24
```

Створимо клас Car:

```
class Car(object):
    def __init__(self):
        self.fuel=0
        self.duration=0
    def service(self):
        self.fuel=round(np.random.triangular(1, 30, 70), 2)
        self.duration=round(np.random.lognormal(2, 0.1), 2)
#-----
```

Інформація про автомобіль стає відомою, коли він обслуговується колонкою. Створимо клас Statistics з DataFrame для ведення статистики (збереження інформації про обслужений автомобіль):

```
class Statistics(object):
    def __init__(self):
        self.df=pd.DataFrame(data=[[0, 0]], columns=['fuel', 'duration'])
    def change_info(self, fuel, duration):
        new_info=pd.DataFrame(data=[[fuel, duration]], columns=['fuel', 'duration'])
        self.df=pd.concat([self.df, new_info], axis=0, ignore_index=True)
#-----
```

Створимо клас загального середовища – Gas_station:

```
class Gas_station(object):
    def __init__(self, current_time):
        self.current_time = current_time
        self.column = Car()
        self.fuel_tank = 10000;
        self.queue = 0
        self.count_car = 0
        self.statistics=Statistics()
```

Визначимо такі методи класу Gas_station:

- Метод виведення загальної інформації про АЗС: залишок часу обслуговування поточної машини, залишок пального в резервуарі, кількість машин у черзі.

```
def display_general_info(self):
    print('--- ', self.current_time.strftime('%d.%m %H:%M'), ' ---')
    print('column: %.2f' % self.column.duration)
    print('fuel_tank: %.2f' % self.fuel_tank)
    print('queue:', self.queue)
```

- Метод збільшення модельного часу на один крок (на 1 годину).

```
def increase_time(self):
    self.current_time+=datetime.timedelta(hours=step)
```

- Метод визначення, визначення кількості машин, які приїхали до АЗС за поточний крок.

```
def car_arrived(self):
    if self.current_time.hour>=7 and self.current_time.hour<=22:
        Lam=7
    else:
        Lam=2
    self.count_car=np.random.poisson(Lam)
    self.queue+=self.count_car
```

- Метод виведення цієї кількості машин на екран.

```
def display_intermediate_info(self):
    print('\n*****')
    print(self.count_car, 'car(s) arrived in the past hour')
```

- Ключовий метод обслуговування машин за годину.

```

def car_service(self):
    time=60*step
    while self.queue>0 and time > 0:
        if self.column.duration == 0:
            self.column.service()
            self.statistics.change_info(self.column.fuel, self.column.duration)
            self.queue-=1
            self.fuel_tank-=self.column.fuel
            self.column.fuel=0
            time-=self.column.duration
        if time>=0:
            self.column.duration=0
        else:
            self.column.duration=-time
            time=0
#-----

```

На цьому етапі вважаємо імітаційну модель створеною.

Створимо функцію для **прогону** моделі, яка виконує цикл моделювання з визначеними параметрами:

```

def simulation(n_step):
    i=0
    while i < n_step:
        GS1.increase_time()
        GS1.car_arrived()
        GS1.car_service()
        GS1.display_intermediate_info()
        GS1.display_general_info()
        i+=1
#-----

```

Створення об'єкту класу Gas_station, запуск моделі та виведення результатів:

```

GS1 = Gas_station(datetime.datetime(1, 9, 1, 7))
GS1.display_general_info()

simulation(count_step)
print('-----')
print('Log of cars')
print(GS1.statistics.df)
print('\Fuel sold')
print(GS1.statistics.df.sum().fuel)
print('Fuel control')
print(10000-GS1.statistics.df.sum().fuel)

plt.hist(GS1.statistics.df.iloc[1:,0],
         bins = 30, density = True)
plt.title('Fuel ordered by cars')
plt.xlabel('Fuel, L')
plt.show()

plt.hist(GS1.statistics.df.iloc[1:,1],
         bins = 20, density = True)
plt.title('Duration of service')
plt.xlabel('Time, min')
plt.show()

```

Результати запуску моделі:

Дата і час старту: 01 вересня, 7:00. На початку біля колонки і в черзі машин немає, резервуар повний:

```
--- 01.09 07:00 ---  
column: 0.00  
fuel_tank: 10000.00  
queue: 0
```

Перша година роботи (7:00-8:00): За цей період було заправлено 4 автомобілі. Станом на 8:00 біля колонки і в черзі машин немає:

```
4 car(s) arrived in the past hour  
--- 01.09 08:00 ---  
column: 0.00  
fuel_tank: 9882.28  
queue: 0
```

Оскільки загальна кількість кроків моделювання становить 24 (повна доба), доцільно розглянути лише найбільш репрезентативні часові інтервали:

```
7 car(s) arrived in the past hour  
--- 01.09 19:00 ---  
column: 0.00  
fuel_tank: 7484.04  
queue: 0
```

```
*****  
11 car(s) arrived in the past hour  
--- 01.09 20:00 ---  
column: 3.26  
fuel_tank: 7267.43  
queue: 3
```

```
6 car(s) arrived in the past hour  
--- 01.09 21:00 ---  
column: 1.85  
fuel_tank: 7012.85  
queue: 1
```

```
*****  
3 car(s) arrived in the past hour  
--- 01.09 22:00 ---  
column: 0.00  
fuel_tank: 6854.14  
queue: 0
```

В інтервалі з 19:00 до 20:00 зафіксовано прибуття 11 транспортних засобів, тоді як пропускна здатність колонки уможливила обслуговування лише 8 з них. Таким чином, 3 автомобілі станом на 20:00 залишилися в черзі.

З 20:00 до 21:00 прибуло ще 6 транспортних засобів. Колонка обслужила ті 3 машини, які перебували в черзі з попереднього періоду, та ще 5 з новоприбулих транспортних засобів. Як результат, на кінець інтервалу (21:00) у черзі залишився 1 автомобіль. А вже о 22:00 черга порожня.

Підсумки роботи АЗС за 24 години роботи – до 7:00 наступного дня(02 вересня): було заправлено 123 автівки, продано 4052,63 літрів пального:

```

Log of cars
      fuel  duration
0      0.00      0.00
1     23.99      8.01
2     38.67      7.65
3     29.57      6.99
4     25.49      8.37
..      ...      ...
119    19.48      7.78
120    18.98      6.54
121    43.09      6.79
122    26.96      7.64
123    49.92      7.24

[124 rows x 2 columns]
\Fuel sold
4052.63
Fuel control
5947.37
>>> GS1.statistics.df.sum().fuel
4052.63

```

Основні статистичні параметри:

```

>>> GS1.statistics.df.iloc[1:,:].describe()
      fuel  duration
count  123.000000  123.000000
mean    32.948211    7.472927
std     13.784013    0.755655
min      5.690000    5.990000
25%     23.280000    6.975000
50%     32.750000    7.510000
75%     42.585000    7.990000
max     67.080000    9.660000

```

Середнє значення кількості пального у одному замовленні та середня тривалість обслуговування одного автомобіля відповідають заданим на початку параметрам моделі.

На рис. 3.3 (а) показано гістограму розподілу обсягів замовленого пального для автомобілів, які обслуговувались на АЗС протягом доби. Розподіл має виражену асиметричну форму з модою близько 30 літрів, що відповідає параметрам заданого трикутного розподілу. На рис. 3.3 (б) показано гістограму розподілу часу обслуговування автомобілів на АЗС у хвилинах. Форма розподілу наближена до логнормального з центром близько 7,5-8,0 хвилин, що узгоджується з параметрами, заданими в моделі ($\mu = 2$, $\sigma = 0,1$).

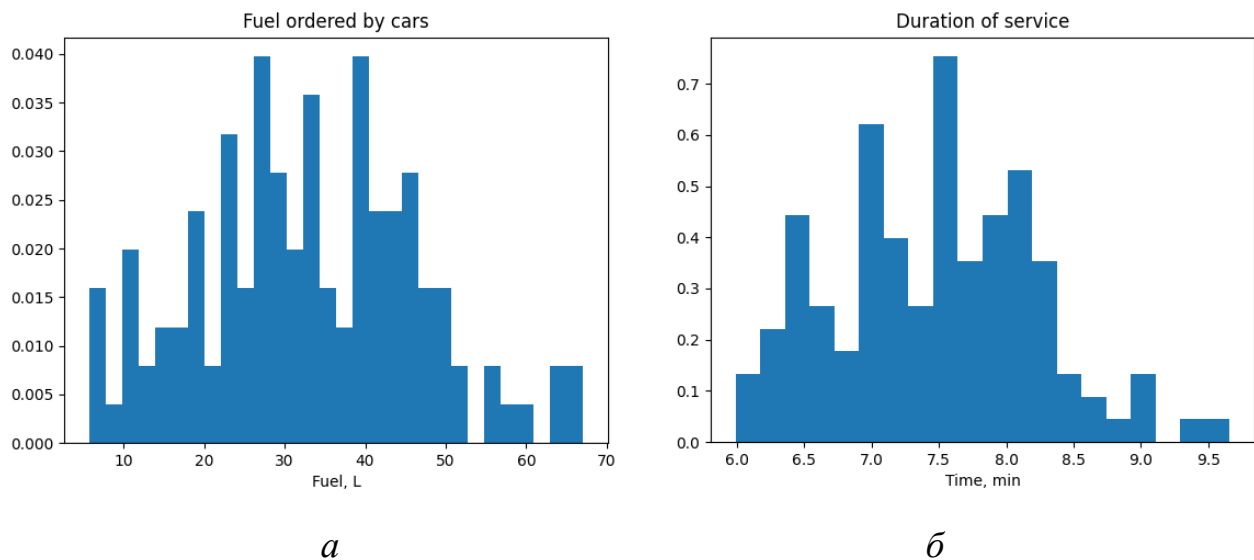


Рисунок 3.3 – Гістограми розподілу параметрів моделі АЗС: обсягів замовленого пального (*a*) та тривалості обслуговування клієнтів (*б*)

Важливо зазначити, що імітаційна модель АЗС базується на використанні генераторів псевдовипадкових чисел для моделювання стохастичних процесів, таких як кількість автомобілів, які прибувають, обсяг замовленого пального та тривалість обслуговування. Внаслідок цього кожний окремий запуск моделі даватиме дещо відмінні результати, що є природним відображенням випадкової природи модельованих процесів.

Для отримання статистично значущих висновків щодо функціонування АЗС необхідно виконати багаторазові прогони моделі з однаковими початковими параметрами. Рекомендується провести щонайменше **100 незалежних експериментів** (прогонів), що надасть можливість зібрати репрезентативні статистичні дані.

Після збору результатів множинних прогонів доцільно застосувати методи математичної статистики для їх аналізу, включаючи розрахунок середніх значень, стандартних відхилень, побудову гістограм розподілу ключових показників та виявлення кореляцій між різними параметрами системи.

Такий комплексний підхід до аналізу результатів моделювання забезпечить високу достовірність висновків та дасть змогу прийняти обґрунтовані рішення щодо вдосконалення функціонування досліджуваної

системи. Наприклад, можна оцінити потенційну доцільність збільшення кількості пальнороздавальних колонок для зменшення часу очікування клієнтів у періоди підвищеного попиту.

Завдання на лабораторну роботу

1. Вибрати завдання для дослідження.
2. Визначити мету, об'єкт та предмет дослідження згідно з вибраним завданням.
3. Висунути гіпотезу дослідження.
4. Сформулювати завдання дослідження. *Примітка: Це не копія завдання на лабораторну роботу і не технічне завдання на програму, а конкретні дослідницькі завдання, які потрібно виконати для досягнення мети.*
5. Описати параметри імітаційної моделі (вхідні величини, вихідні характеристики, вибрані закони розподілу випадкових величин та їхні параметри).
6. Обґрунтувати прийняті спрощення та описати обмеження моделі.
7. Побудувати алгоритм роботи імітаційної моделі чи її ключових функцій.
8. Розробити програмний код імітаційної моделі.
9. Перевірити адекватність моделі (*перевірити, чи не суперечать результати моделювання логіці процесу в разі зміни величин найважливіших параметрів*).
10. Провести комп'ютерні експерименти (прогони моделі), зібрати та проаналізувати результати (усереднені за 100 прогонів моделі).
11. Зробити висновки щодо функціонування досліджуваної системи.

Варіанти завдань

1. Моделювання роботи однієї каси магазину. Магазин працює цілодобово. Швидкість обслуговування лінійно залежить від кількості товару у покупця (параметри лінійної залежності вибрати самостійно). Кількість товару є

випадковою величиною з біноміальним розподілом. Крок зміни модельного часу – 20 хвилин. Тривалість моделювання – 1 доба. Кількість покупців є випадковою величиною, розподіленою за законом Пуассона, з такими параметрами:

- З 08:00 до 10:00 – математичне сподівання 7 покупців за 20 хвилин.
- З 10:00 до 17:00 – математичне сподівання 4 покупці за 20 хвилин.
- З 17:00 до 20:00 – математичне сподівання 10 покупців за 20 хвилин.
- З 20:00 до 23:00 – математичне сподівання 7 покупців за 20 хвилин.
- З 23:00 до 08:00 – математичне сподівання 1 покупець за 20 хвилин.

Чергу до каси вважати необмеженої довжини. Якщо черга впродовж моделювання виявиться зavelика – скоригувати швидкість обслуговування чи математичні сподівання для надходження покупців.

2. Моделювання роботи одного перекладача. Вартість та тривалість виконання одного замовлення на переклад лінійно залежить від кількості сторінок у документі. Швидкість перекладу в середньому становить 2 сторінки на годину. Вартість перекладу 1 сторінки – 10 умовних одиниць. Перекладач працює по 8 годин на день, 5 днів на тиждень. На основі тривалих спостережень було виявлено, що кількість сторінок у документах – випадкова величина, розподілена за законом Пуассона з математичним сподіванням 7 сторінок в одному документі. Кількість документів, які надходять за робочий день, є випадковою величиною з рівномірним законом розподілу. Параметри розподілу залежать від дня тижня: понеділок – [1, 4], вівторок – [2, 3], середа – [3, 5], четвер – [1, 3], п'ятниця – [1, 2]. Крок зміни модельного часу – один робочий день. Тривалість моделювання – 10 тижнів. Ймовірність поганого самопочуття перекладача – 0,05 (у цей день він не працює, хоча документи надходять і стають у чергу замовлень). Нові замовлення приймаються до тих пір, поки черга не перевищить 40 сторінок. Якщо черга переповнена, то замовлення не приймається. Не можна відхиляти сторінки, лише замовлення повністю. За потреби скоригувати кількість документів, які надходять за робочий день, та математичне сподівання кількості сторінок в одному документі.

3. Моделювання роботи однієї складальної дільниці цеху підприємства. На складальну дільницю через випадкові інтервали часу, розподілені

експоненціально з середнім значенням 10 хв, надходить партія з 4 деталей. Половина деталей перед складанням має пройти попередню обробку тривалістю 5 хв. Процес складання однієї деталі займає 8 хв. Після складання виріб надходить на регулювання, яке триває в середньому 8 хв (час виконання – випадкова величина з логонормальним розподілом). Під час регулювання з ймовірністю 0,01 може виявитися, що виріб бракований, з ймовірністю 0,02, що одна з деталей бракована і потребує заміни (час заміни – 3 хв). Крок зміни модельного часу вибрати самостійно. Тривалість моделювання – 1 зміна (8 годин). За потреби скоригувати тривалість виробничих процесів.

4. Моделювання роботи піцерії з однією піччю для виготовлення піци. Швидкість виготовлення стандартної піци з меню – 10 хвилин, індивідуальне замовлення – 15 хвилин. Вартість стандартної піци – від 10 до 15 умовних одиниць, вартість індивідуального замовлення – від 15 до 20 умовних одиниць, (рівномірний закон розподілу). Крок зміни модельного часу – 5 хвилин. Тривалість моделювання – 1 робочий день. Ймовірність того, що протягом 5 хвилин надійде замовлення залежить від часу дня: з 09:00 до 11:00 – 0,3; з 11:00 до 15:00 – 0,5; з 15:00 до 20:00 – 0,9; з 20:00 до 22:00 – 0,7. Піцерія працює з 09:00 до 23:00, але замовлення приймаються до 22:00. «Індивідуальність» замовлення – випадкова величина з розподілом Бернуллі (дивитися біноміальний розподіл для однократного випробовування). В черзі може знаходитися не більше 5 замовлень, інакше замовлення не приймається. За потреби скоригувати швидкість виготовлення піци та ймовірності надходження замовлення.

5. Моделювання роботи одного стоматолога. Графік прийому у стоматологічній клініці: з понеділка по п'ятницю з 09:00 до 18:00, субота з 10:00 до 16:00, обідня перерва з 13:00 до 14:00. Самостійно розробити перелік послуг, які надає стоматолог (4-5 послуг). Для кожної послуги задати вартість і тривалість виконання. Визначити ймовірності замовлення цих послуг (див. лекцію 4 – моделювання повної групи несумісних подій). Кількість клієнтів протягом робочого дня є випадковою величиною (самостійно вибрати закон розподілу та обґрунтувати вибір). Якщо клієнт не встигає потрапити до

стоматолога, то вважаємо, що клініка втрачає цього клієнта, оскільки він йде до конкурентів. Крок зміни модельного часу – одна година. Тривалість моделювання – 1 тиждень (моделювати лише робочі години). Ймовірність відсутності світла чи води у клініці – 0,05, тривалість відключення 3-4 години – на цей час обслуговування призупиняється.

6. Моделювання роботи одного поверху підземного паркінгу торговельно-розважального центру (ТРЦ). На поверсі передбачено 100 місць для паркування транспортних засобів. Якщо клієнт не знаходить місця для паркування, то вважати, що ТРЦ втрачає цього клієнта, оскільки він їде до сусіднього ТРЦ. Паркінг працює з 07:00 до 23:00, а ТРЦ відкритий для клієнтів з 08:00 до 22:00. Крок зміни модельного часу – одна година. Тривалість моделювання – 1 тиждень. Кількість клієнтів вважати випадковою величиною, але середня кількість залежить від часу доби та дня тижня (самостійно вибрати закон розподілу та його параметри, обґрунтувати вибір). Тривалість перебування автомобіля на паркінгу є випадковою величиною з експоненційним розподілом із середнім значенням 2 години.

7. Моделювання роботи кав'ярні з однією станцією приготування напоїв. Кав'ярня працює з 07:00 до 22:00. Максимальна довжина черги – 7 клієнтів. Якщо клієнт бачить, що черга перевищує цю кількість у момент його прибуття, то він йде до конкурентів. Крок зміни модельного часу – 10 хвилин. Тривалість моделювання – 1 день. Кількість клієнтів є випадковою величиною з розподілом Пуассона. Математичне сподівання залежить від часу дня (самостійно вибрати параметри).

Асортимент, вартість, тривалість приготування та ймовірність замовлення напою:

- еспreso: 1 у.о., 1 хв, ймовірність замовлення – 0,25;
- американо: 1,5 у.о., 1,5 хв, ймовірність замовлення – 0,4;
- капучино: 2 у.о., 2 хв, ймовірність замовлення – 0,15;
- лате: 2,5 у.о., 2,5 хв, ймовірність замовлення – 0,15;
- гарячий шоколад: 3,5 у.о., 3 хв, ймовірність замовлення – 0,05;

Один клієнт може замовити декілька напоїв, наприклад, два еспресо або одне еспресо і одне американо. Для моделювання кількості напоїв у замовленні клієнта використати геометричний розподіл (параметр вибрати самостійно). Вид кожного з напоїв у замовленні вибирається відповідно до заданих ймовірностей замовлення.

8. Моделювання роботи закладу швидкого харчування з однією точкою видачі. Замовлення клієнт може зробити з 08:00 до 22:00. Крок зміни модельного часу – 10 хвилин. Тривалість моделювання – 1 день. Кількість клієнтів є випадковою величиною, розподіленою за законом Пуассона (самостійно вибрати параметри). Можливі варіанти поведінки клієнтів:

- Подія A – замовлення лише чогось одного з перших страв (швидкість обслуговування 1 хвилина);
- Подія B – замовлення лише чогось одного з других страв (швидкість обслуговування 1,5 хвилини);
- Подія AB – замовлення одночасно чогось одного з перших і чогось одного з других страв (швидкість обслуговування 2 хвилини);
- Подія $\overline{AB}$ – нічого не замовляє та йде через те, що не сподобався асортимент (втрачені клієнти).

Виявлені такі ймовірності настання цих залежних і сумісних подій: $P(A) = 0,45$; $P(B) = 0,85$; $P(AB) = 0,35$ (див. лекцію 4 – моделювання сумісних подій).

Чергу до точки видачі замовлення вважати необмеженою.

9. Моделювання роботи сервісного центру з ремонту техніки з одним майстром. Сервісний центр працює з 09:00 до 18:00, 5 днів на тиждень. Кількість клієнтів є випадковою величиною, розподіленою за законом Пуассона, з такими параметрами:

- З 09:00 до 13:00 – математичне сподівання 3 клієнти за годину;
- З 13:00 до 17:00 – математичне сподівання 4 клієнти за годину;
- З 17:00 до 18:00 – математичне сподівання 5 клієнтів за годину.

Одне замовлення складається лише з однієї одиниці техніки. Вартість і час ремонту техніки залежить від типу техніки та складності поломки:

- Смартфони: ймовірність 0,5, 10 у.о., час ремонту 30 хв;
- Планшети: ймовірність 0,2, 20 у.о., час ремонту 45 хв;
- Ноутбуки: ймовірність 0,2, 30 у.о., час ремонту 60 хв;

- Інша техніка: ймовірність 0,1, 40 у.о., час ремонту 90 хв.

Чергу до сервісного центру вважати обмеженою 12 клієнтами. Якщо черга повна, клієнт іде до іншого сервісного центру. Крок зміни модельного часу – 1 година. Тривалість моделювання – 1 тиждень. За потреби скоригувати тривалість ремонту та ймовірності надходження замовлення.

10. Моделювання роботи пункту видачі інтернет-замовлень з одним працівником. Пункт видачі працює з 10:00 до 20:00 щодня. Крок зміни модельного часу – 15 хвилин. Тривалість моделювання – 1 день. Час обслуговування клієнта залежить від розміру замовлення: по 3 хвилини на огляд кожного товару у замовленні та оплата всього замовлення – 1 хв. Кількість товарів у замовленні є випадковою величиною з геометричним розподілом з ймовірністю успіху $p = 0,3$.

Кількість клієнтів є випадковою величиною, розподіленою за законом Пуассона, з такими параметрами:

- З 10:00 до 14:00 – математичне сподівання 4 клієнти за 15 хвилин;
- З 14:00 до 17:00 – математичне сподівання 6 клієнтів за 15 хвилин;
- З 17:00 до 20:00 – математичне сподівання 8 клієнтів за 15 хвилин.

Ймовірність того, що клієнт відмовиться принаймні від одного з товарів у замовленні – 0,1, що збільшує загальний час обслуговування ще на 4 хв. Чергу до пункту видачі вважати обмеженою 15 клієнтами. Якщо черга повна, клієнт приходить іншого дня (вважати втраченим клієнтом). За потреби скоригувати інтенсивність надходження клієнтів.

Зміст звіту

1. Титульний аркуш;
2. Умова вибраного завдання;
3. Мета, об'єкт і предмет дослідження;
4. Гіпотеза дослідження;
5. Завдання дослідження (орієнтовно 4-5 пунктів – «що зробити?»);
6. Опис параметрів імітаційної моделі;

7. Спрощення та обмеження моделі;
8. Блок-схема алгоритму роботи імітаційної моделі;
9. Код програми (може бути оформлений як додаток або прикріплений як окремий текстовий файл чи архів, якщо файлів багато);
10. Аналіз адекватності моделі;
11. Результати моделювання:
 - за один прогон моделі – діаграми, гістограми та фрагмент логу,
 - середні показники за 100 прогонів моделі – таблиці, гістограми та діаграми;
12. Висновки.

Питання для самоконтролю

1. Що таке імітаційні моделі та яке їх призначення?
2. Чим відрізняється імітаційне моделювання від аналітичного?
3. На якому математичному методі базується імітаційне моделювання?
4. Що означає термін «прогін» в контексті імітаційного моделювання?
5. Що таке модельний час в імітаційних моделях?
6. У чому полягає різниця між покроковим та подієвим способом зміни модельного часу?
7. Які основні компоненти входять до структури імітаційної моделі?
8. Як відбувається збирання та обробка статистичних даних під час імітаційного моделювання?
9. Коли використовуються неперервні закони розподілу, а коли дискретні? Наведіть приклади.
10. Поясніть потребу у коригуванні параметрів імітаційної моделі.

Лабораторна робота №4

Системи масового обслуговування

Мета – ознайомитися з методами дослідження систем масового обслуговування шляхом аналітичного та імітаційного моделювання.

Теоретичні відомості

На лекціях 6-8 було розглянуто найпростіші види систем масового обслуговування, для яких отримані аналітичні вирази розрахунку характеристик системи. Такі аналітичні моделі досить зручні у використанні, оскільки не потребують значних витрат обчислювальних ресурсів, дають змогу кількісно оцінити характер впливу вхідних даних на вихідні параметри і виявити загальні закономірності в поведінці системи.

Проте аналітичне дослідження вдається реалізувати тільки для порівняно нескладних систем. Для складних систем аналітичні моделі, якщо й вдається побудувати, то тільки за певних припущень, які спрощують систему, що ставить під сумнів адекватність такої моделі.

Перевагою імітаційного моделювання систем масового обслуговування є можливість відтворення процесів будь-якої складності з урахуванням їхніх індивідуальних особливостей, таких як: нестаціонарність потоків заявок, складні дисципліни обслуговування, різні закони розподілу часу обслуговування та інтервалів між надходженнями заявок тощо. За допомогою імітаційної моделі можна проводити численні експерименти з системою, змінюючи її параметри та структуру.

Імітаційне моделювання супроводжується збиранням статистичних даних, таких як: кількість заявок, які були обслужені та не обслужені (втрачені) системою, сумарний час зайнятого стану для кожного з пристроїв, середні довжини черг тощо. Аналіз зібраних результатів дає змогу знайти оптимальні проектні рішення перед реалізацією реальної системи.

Проте слід зазначити, що побудова імітаційної моделі вимагає більших трудовитрат порівняно з аналітичним моделюванням, а результати імітації мають статистичну природу і потребують додаткової обробки та інтерпретації.

Незважаючи на ці обмеження, саме **імітаційне** моделювання вважають основним підходом до аналізу систем масового обслуговування на етапі проектування, тоді як **аналітичне** дослідження використовують для попередньої оцінки різних варіантів систем.

Приклад розроблення аналітичної моделі системи масового обслуговування

Побудуємо аналітичну модель станції технічного обслуговування автомобілів з трьома бригадами механіків. До автосервісу в середньому під'їжджає 7 машин на годину, тобто інтенсивність вхідного потоку $\lambda = 7$ вимог/год. Один клієнт обслуговується однією бригадою механіків. Середній час обслуговування одного клієнта $t_{обсл} = 45 \text{ хв} = 0,75 \text{ год}$. У черзі може розміститися не більше 4 автомобілів: $m = 4$. Якщо місця у черзі не вистачає, то клієнтам відмовляється в обслуговуванні.

Визначимо вид такої системи масового обслуговування – багатоканальна з обмеженою чергою. На рис. 4.1 показано схему такої системи.

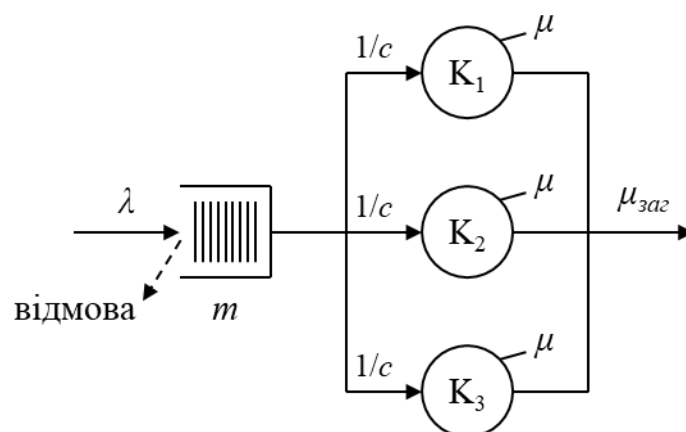


Рисунок 4.1 – Схема станції технічного обслуговування автомобілів з трьома бригадами механіків з обмеженою чергою

Число вимог у черзі в цьому випадку не може перевищувати $m = 4$, а така система матиме не більше $c+m+1 = 3+4+1 = 8$ станів (рис. 4.2).

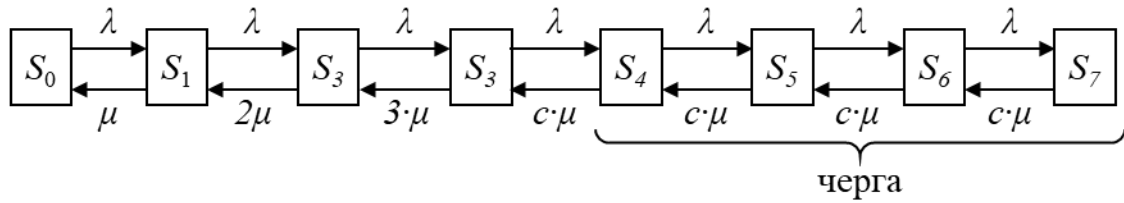


Рисунок 4.2 – Граф станів станції технічного обслуговування автомобілів з трьома бригадами механіків з обмеженою чергою

Інтенсивність обслуговування однією бригадою механіків: $\mu = \frac{1}{t_{обсл}} = \frac{4}{3}$ вимог/год. Навантаження на одну бригаду: $r = \frac{\lambda}{\mu} = 7 \cdot \frac{3}{4} = 5,25$. Кількість каналів обслуговування $c = 3$. Загальна інтенсивність обслуговування всієї системи: $\mu_{зар} = c \cdot \mu = 3 \cdot \frac{4}{3} = 4$ вимоги/год. Тобто, три бригади механіків, які працюють паралельно, здатні обслуговувати в середньому 4 автомобілі за годину. Приведена завантаженість системи $\rho = \frac{\lambda}{\mu_{зар}} = \frac{7}{4} = 1,75$.

Знайдемо ймовірність відсутності клієнтів на території автосервісу.

Оскільки $\rho \neq 1$, то використаємо формулу: $q_0 = \left(\left(\sum_{i=0}^{c-1} \frac{r^i}{i!} \right) + \frac{r^c}{c!} \cdot \frac{1-\rho^{m+1}}{1-\rho} \right)^{-1}$.

$$q_0 = \left(1 + 5,25 + \frac{5,25^2}{2!} + \frac{5,25^3}{3!} \cdot \frac{(1 - 1,75^5)}{(1 - 1,75)} \right)^{-1} = \frac{1}{(20,031250 + 495,627045)} \approx 0,001939.$$

Знайдемо граничні ймовірності:

$$q_1 = 5,25 \cdot 0,001939 \approx 0,010181,$$

$$q_2 = \frac{5,25^2}{2!} \cdot 0,001939 \approx 0,026726,$$

$$q_3 = \frac{5,25^3}{3!} \cdot 0,001939 \approx 0,046770,$$

$$q_4 = \frac{5,25^3}{3!} \cdot 1,75 \cdot 0,001939 \approx 0,081847,$$

$$q_5 = \frac{5,25^3}{3!} \cdot 1,75^2 \cdot 0,001939 \approx 0,143232,$$

$$q_6 = \frac{5,25^3}{3!} \cdot 1,75^3 \cdot 0,001939 \approx 0,250656,$$

$$q_7 = \frac{5,25^3}{3!} \cdot 1,75^4 \cdot 0,001939 \approx 0,438649,$$

Ймовірність того, що вимога потрапить у чергу: $p_q = \sum_{k=0}^{m-1} q_{c+k}$.

Оскільки $\rho \neq 1$, то використаємо формулу: $p_q = \frac{r^c}{c!} \cdot \frac{1-\rho^m}{1-\rho} \cdot q_0$.

$$p_q = \frac{5,25^3 \cdot (1 - 1,75^4)}{3! \cdot (1 - 1,75)} \cdot 0,001939 \approx 0,522505.$$

Ймовірність відмови: $p_{\text{відмови}} = q_{c+m} = 0,438649$.

Відносна пропускна здатність: $Q = 1 - p_{\text{відмови}} = 0,561351$.

Абсолютна пропускна здатність:

$$A = \lambda \cdot Q = 7 \cdot 0,561351 \approx 3,929 \text{ вимоги/год.}$$

Середня кількість зайнятих каналів: $\bar{k}_{\text{зайнятих}} = r \cdot Q \approx 2,947$.

Середня кількість вимог у черзі для випадку $\rho \neq 1$ розраховується за формулою:

$$L_q = \frac{r^c}{c!} \cdot \rho \cdot \frac{1 - \rho^m - m \cdot \rho^m (1 - \rho)}{(1 - \rho)^2} \cdot q_0$$

$$L_q = \frac{5,25^3}{3!} \cdot 1,75 \cdot \frac{1 - 1,75^4 - 4 \cdot 1,75^4 \cdot (1 - 1,75)}{(1 - 1,75)^2} \cdot 0,001939 \approx 2,875.$$

Середня кількість вимог у системі: $L_s = L_q + \bar{k}_{\text{зайнятих}} \approx 5,822$.

Середній час перебування вимог у черзі: $W_q = \frac{2,875}{7} \approx 0,411 \text{ год.}$

Середній час перебування вимог у системі: $W_s = \frac{5,822}{7} \approx 0,832 \text{ год.}$

Нехай технічне обслуговування одного автомобіля приносить у середньому дохід у розмірі 35 умовних одиниць (у.о.), а утримання однієї ремонтної бригади коштує 20 у.о./годину. Тоді виручку можна визначити за формулою:

$$D = 35 \cdot A - 20 \cdot c \approx 77,5 \text{ у.о./год.}$$

З отриманих результатів видно, що автосервіс перевантажений. Розрахуємо характеристики системи за більшої кількості каналів обслуговування (рис. 4.3).

	c = 3	c = 4	c = 5	c = 6	c = 7
ro	1.750	1.312	1.050	0.875	0.750
q_0	0.002	0.003	0.004	0.004	0.005
p_vidmovy	0.439	0.278	0.156	0.077	0.034
p_q	0.523	0.591	0.552	0.433	0.291
Q	0.561	0.722	0.844	0.923	0.966
A	3.929	5.051	5.910	6.463	6.764
k_zaniat	2.947	3.788	4.433	4.848	5.073
L_q	2.875	2.196	1.485	0.884	0.470
L_s	5.822	5.985	5.917	5.732	5.543
W_q	0.411	0.314	0.212	0.126	0.067
W_s	0.832	0.855	0.845	0.819	0.792
D	77.531	96.786	106.852	106.217	96.739

Рисунок 4.3 – Характеристики автосервісу залежно від кількості ремонтних бригад

Зі збільшенням кількості ремонтних бригад збільшилася абсолютна пропускна здатність автосервісу (рис. 4.4).

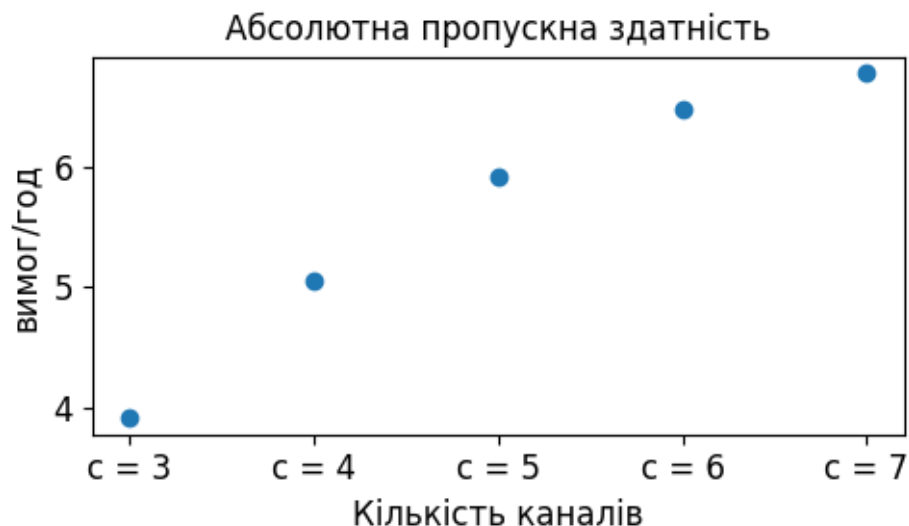


Рисунок 4.4 – Характер залежності абсолютної пропускної здатності автосервісу від кількості ремонтних бригад

Найбільш стрімке зростання показника спостерігається за збільшення кількості ремонтних бригад до 6. Подальше ж збільшення кількості каналів обслуговування дає лише незначне збільшення показника.

Побудуємо діаграму залежності виручки автосервісу від кількості ремонтних бригад (рис. 4.5).

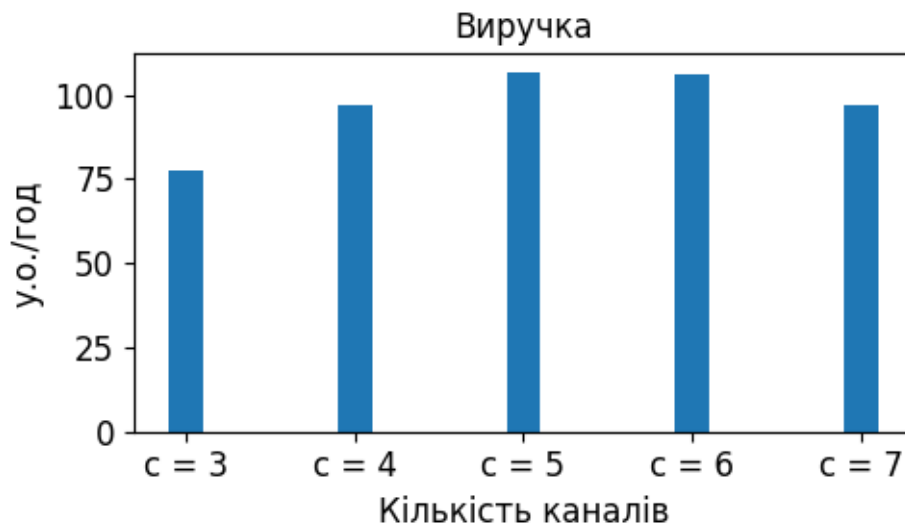


Рисунок 4.5 – Виручка за годину роботи автосервісу залежно від кількості ремонтних бригад

Як видно з діаграми, найбільшу виручку автосервіс за заданих умов отримає, якщо найме 5 ремонтних бригад. Якщо ремонтних бригад найняти не 5, а 6, то виручка майже не знизиться, але зменшиться ймовірність відмови, ймовірність черги та середній час очікування клієнта у черзі.

Завдання на лабораторну роботу

1. Продовжити тему дослідження попередньої лабораторної роботи, розширюючи імітаційну модель одного процесу на випадок паралельного виконання кількох таких процесів у багатоканальній системі масового обслуговування.
2. Визначити мету, об'єкт та предмет дослідження.
3. Висунути гіпотезу дослідження.

4. Сформулювати завдання дослідження. *Примітка: Це не копія завдання на лабораторну роботу і не технічне завдання на програму, а конкретні дослідницькі завдання, які потрібно виконати для досягнення мети.*
5. Описати параметри окремо аналітичної моделі, окремо – імітаційної моделі (вхідні величини, вихідні характеристики, вибрані закони розподілу випадкових величин та їхні параметри).
6. Побудувати схему багатоканальної системи масового обслуговування та її граф станів (див. 8 лекцію).
7. Розробити комп'ютерні програми – окремо для **імітаційної** моделі, що реалізує метод статистичних випробувань (метод Монте-Карло), та окремо для **аналітичної** моделі, яка базується на теорії масового обслуговування (див. 8 лекцію).
8. Перевірити адекватність моделей (*перевірити, чи не суперечать результати моделювання логіці процесу в разі зміни величин найважливіших параметрів*).
9. Провести комп'ютерні експерименти з різними параметрами системи, зібрати та проаналізувати результати. Порівняти результати імітаційного моделювання (середні за 100 прогонів моделі) з результатами аналітичного моделювання.
10. Зробити висновки щодо функціонування досліджуваної системи, обґрунтувати доцільність прийнятих рішень та надати практичні рекомендації щодо організації роботи багатоканальної системи масового обслуговування.

Варіанти завдань

1. Моделювання роботи магазину, який працює цілодобово. У магазині є 5 касових апаратів, з яких 1 працює постійно, а інші вмикають за потреби. Чергу вважаємо необмеженою та єдиною до всіх кас. Власника магазину цікавить, як розподілити графік роботи касирів – по скільки касирів треба виводити на

чергування в кожний період часу: з 08:00 до 10:00, з 10:00 до 17:00, з 17:00 до 23:00, з 23:00 до 08:00. Час очікування на обслуговування не має перевищувати 10 хвилин.

Швидкість обслуговування лінійно залежить від кількості товару у покупця (параметри лінійної залежності вибрати самостійно). Кількість товару – випадкова величина з біноміальним розподілом. Крок зміни модельного часу – 20 хв. Тривалість моделювання – 1 день. Кількість покупців є випадкова величина, розподілена за законом Пуассона (параметри вибрати самостійно, але так, щоб в пікові години працювало усі 5 кас).

2. Моделювання роботи агентства з експрес-перекладу документів. В агентстві працює декілька перекладачів. Черга до них одна. Нові замовлення приймаються до тих пір, поки черга не перевищить 40 сторінок. Якщо черга переповнена, то замовлення не приймається. Кількість документів, які надходять за робочий день, вважати випадковою величиною в проміжку $[a, b]$ з рівномірним законом розподілу. Залежить від дня тижня: понеділок – $[5, 12]$, вівторок – $[6, 9]$, середа – $[9, 15]$, четвер – $[3, 9]$, п'ятниця – $[3, 7]$. За потреби скоригувати кількість документів, які надходять за робочий день, та математичне сподівання кількості сторінок в одному документі. Власника агентства цікавить, скільки перекладачів йому потрібно найняти, щоб мінімізувати кількість відмов у прийнятті замовлень, але при цьому ймовірність простою системи була менша за 0,15.

Вартість та тривалість виконання одного замовлення на переклад лінійно залежить від кількості сторінок у документі. Кількість сторінок у документах – випадкова величина, розподілена за законом Пуассона з математичним сподіванням 7 сторінок в одному документі. Швидкість перекладу в середньому становить 2 сторінки на годину. Вартість перекладу 1 сторінки – 10 умовних одиниць. Крок зміни модельного часу – один робочий день. Тривалість моделювання – 10 тижнів. Перекладачі працюють по 8 годин на день, 5 днів на тиждень. Ймовірність поганого самопочуття перекладача – 0,05 (у цей день він не працює, хоча документи надходять і стають у чергу замовлень).

3. Моделювання роботи цеху підприємства, який складається чотирьох складальних дільниць. У цех підприємства через випадкові інтервали часу, розподілені експоненціально з середнім значенням X хвилин, надходить партія з 4 деталей. Кожна дільниця може обробляти одну партію деталей одночасно.

Якщо всі складальні дільниці зайняті, то партія деталей стає у чергу. Черга одна, загальна для всіх дільниць, необмежена. Визначити X , за якого черга за зміну не буде перевищувати 20 партій.

Половина деталей перед складанням має пройти попередню обробку тривалістю 5 хв. Процес складання однієї деталі займає 8 хв. Після складання виріб надходить на регулювання, яке триває в середньому 8 хв (час виконання – випадкова величина з логонормальним розподілом). Під час регулювання з ймовірністю 0,01 може виявитися, що виріб бракований, з ймовірністю 0,02, що одна з деталей бракована і потребує заміни (час заміни – 3 хв). Крок зміни модельного часу вибрати самостійно. Тривалість моделювання – 1 зміна (8 годин). За потреби скоригувати тривалість виробничих процесів.

4. Моделювання роботи піцерії з 10 печами для виготовлення піци. Черга до них одна. В черзі може знаходитися не більше 10 замовлень, інакше замовлення не приймається. Утримання 1 печі ввімкненою коштує закладу 20 у.о. за годину. Власника піцерії цікавить, скільки печей потрібно вмикати в різні періоди часу, щоб отримати максимальний прибуток. Ймовірність того, що протягом 1 хвилини надійде замовлення залежить від часу дня: з 09:00 до 11:00 – 0,1; з 11:00 до 15:00 – 0,3; з 15:00 до 20:00 – 0,5; з 20:00 до 22:00 – 0,4.

Швидкість виготовлення стандартної піци з меню – 10 хвилин, індивідуальне замовлення – 15 хвилин. Вартість стандартної піци – від 10 до 15 умовних одиниць, вартість індивідуального замовлення – від 15 до 20 умовних одиниць (рівномірний закон розподілу). Крок зміни модельного часу – 5 хвилин. Тривалість моделювання – 1 робочий день. Піцерія працює з 09:00 до 23:00, але замовлення приймаються до 22:00. «Індивідуальність» замовлення – випадкова величина з розподілом Бернуллі. За потреби скоригувати швидкість виготовлення піци та ймовірності надходження замовлення.

5. Моделювання роботи стоматологічної клініки, в якій працює декілька стоматологів. До клініки в середньому звертається 12 осіб на годину. Черга до стоматологів одна. У черзі може розміститися не більше 10 клієнтів. Якщо місця у черзі не вистачає, то клієнтам відмовляється в обслуговуванні. Середній час обслуговування одного клієнта становить 30 хвилин. Клініка платить 100 умовних одиниць на день кожному стоматологу. Власника стоматологічної

клініки цікавить, скільки йому потрібно найняти стоматологів, щоб отримати максимальний прибуток.

Графік прийому у стоматологічній клініці: з понеділка по п'ятницю з 09:00 до 18:00, субота з 10:00 до 16:00, обідня перерва з 13:00 до 14:00. Самостійно розробити перелік послуг, які надає стоматолог (4-5 послуг). Для кожної послуги задати вартість і тривалість виконання. Визначити ймовірності замовлення цих послуг. Кількість клієнтів протягом робочого дня є випадковою величиною (самостійно вибрати закон розподілу та обґрунтувати вибір). Якщо клієнт не встигає потрапити до стоматолога, то вважаємо, що клініка втрачає цього клієнта, оскільки він йде до конкурентів. Крок зміни модельного часу – одна година. Тривалість моделювання – 1 тиждень (моделювати лише робочі години). Ймовірність відсутності світла чи води у клініці – 0,05, тривалість відключення 3-4 години – на цей час обслуговування призупиняється.

6. Моделювання роботи підземного паркінгу. Через зростання популярності деякої місцевості бізнесмен хоче побудувати платний паркінг з декількома поверхами. Площа, яка відведена під паркінг, дасть змогу на одному поверсі розмістити 100 транспортних засобів. Утримання одного поверху буде коштувати у майбутньому бізнесмену 5 000 у.о. на місяць. Година паркування одного автомобіля буде приносити бізнесмену 10 у.о. Йому потрібно визначити, скільки поверхів буде приносити максимальний прибуток.

Якщо клієнт не знаходить місця для паркування, то паркінг втрачає цього клієнта. Паркінг працюватиме з 07:00 до 23:00. Крок зміни модельного часу – одна година. Тривалість моделювання – 1 тиждень. Кількість клієнтів вважати випадковою величиною, але середня кількість залежить від часу доби та дня тижня (самостійно вибрати закон розподілу та його параметри, обґрунтувати вибір). Тривалість перебування автомобіля на паркінгу є випадковою величиною з експоненційним розподілом із середнім значенням 2 години.

7. Моделювання роботи кав'ярні з декількома станціями приготування напоїв. В середньому на обслуговування однієї станції, коли та використовується, заклад витрачає 20 у.о. на годину (сюди входить і заробітна плата персоналу). Кав'ярня має 5 станцій для приготування напоїв. Черга до кав'ярні одна. Максимальна довжина черги – 10 клієнтів. Якщо клієнт бачить, що черга перевищує цю кількість у момент його прибуття, то він йде до

конкурентів. Власника кав'ярні цікавить, скільки станцій потрібно використовувати в різні періоди дня, щоб отримувати якомога більший прибуток за якомога меншої втрати клієнтів.

Кав'ярня працює з 07:00 до 22:00. Крок зміни модельного часу – 10 хвилин. Тривалість моделювання – 1 день. Кількість клієнтів є випадковою величиною з розподілом Пуассона. Математичне сподівання кількості клієнтів залежить від часу дня (самостійно вибрати параметри). Асортимент закладу, вартість, тривалість приготування та ймовірність замовлення напою як і в попередній лабораторній роботі. Один клієнт може замовити декілька напоїв, наприклад, два еспreso або одне еспreso і одне американо. Для моделювання кількості напоїв у замовленні клієнта використати геометричний розподіл (параметр вибрати самостійно). Вид кожного з напоїв у замовленні вибирається відповідно до заданих ймовірностей замовлення.

8. Моделювання роботи закладу швидкого харчування з декількома точками видачі. Черга необмежена, загальна для всіх точок видачі. Власника закладу цікавить, скільки точок видачі потрібно використовувати в різні періоди дня, щоб отримувати якомога більший прибуток, але водночас щоб середній час перебування клієнта в системі не перевищував 15 хвилин. Утримання однієї точки видачі коштує 150 у.о. на день.

Замовлення клієнт може зробити з 08:00 до 22:00. Крок зміни модельного часу – 10 хвилин. Тривалість моделювання – 1 день. Кількість клієнтів є випадковою величиною, розподіленою за законом Пуассона (самостійно вибрати параметри). Можливі варіанти поведінки клієнтів як і в попередній лабораторній роботі.

9. Моделювання роботи сервісного центру з ремонту техніки з декількома майстрами. Черга до майстрів одна, загальна, обмежена 15 клієнтами. Власника сервісного центру цікавить, скільки майстрів потрібно найняти, щоб максимізувати прибуток. Зарплата одного майстра – 100 у.о. на день.

Сервісний центр працює з 09:00 до 18:00, 5 днів на тиждень. Крок зміни модельного часу – 1 година. Тривалість моделювання – 1 тиждень. Кількість клієнтів є випадковою величиною, розподіленою за законом Пуассона (самостійно вибрати параметри). Одне замовлення складається лише з однієї одиниці техніки. Вартість і час ремонту техніки як і в

попередній лабораторній роботі. За потреби скоригувати тривалість ремонту та ймовірності надходження замовлення.

10. Моделювання роботи пункту видачі інтернет-замовлень з декількома працівниками. Черга до працівників одна, загальна, обмежена 15 клієнтами. Якщо черга повна, клієнт приходить іншого дня (вважати втраченим клієнтом). Власника пункту видачі цікавить, скільки працівників потрібно найняти, щоб зменшити кількість втрачених клієнтів, але водночас ймовірність простою системи була менша за 0,15.

Пункт видачі працює з 10:00 до 20:00 щодня. Крок зміни модельного часу – 15 хвилин. Тривалість моделювання – 1 день. Час обслуговування клієнта залежить від розміру замовлення: по 3 хвилини на огляд кожного товару у замовленні та оплата всього замовлення – 1 хв. Кількість товарів у замовленні є випадковою величиною з геометричним розподілом з ймовірністю успіху $p = 0,3$. Кількість клієнтів є випадковою величиною, розподіленою за законом Пуассона з параметрами, які залежать від часу доби (самостійно вибрати параметри для різних періодів дня). Ймовірність того, що клієнт відмовиться принаймні від одного з товарів у замовленні – 0,1, що збільшує загальний час обслуговування ще на 4 хв. За потреби скоригувати інтенсивність надходження клієнтів.

Зміст звіту

1. Титульний аркуш;
2. Умова вибраного завдання;
3. Мета, об'єкт і предмет дослідження;
4. Гіпотеза дослідження;
5. Завдання дослідження (орієнтовно 4-5 пунктів – «що зробити?»);
6. Опис параметрів імітаційної моделі;
7. Опис параметрів аналітичної моделі;
8. Схема багатоканальної системи масового обслуговування;
9. Граф станів багатоканальної системи масового обслуговування;
10. Код програми **імітаційної** моделі, яка реалізує метод статистичних випробувань (метод Монте-Карло);

11. Код програми **аналітичної** моделі, яка базується на теорії масового обслуговування;
12. Аналіз адекватності моделі;
13. Результати моделювання:
 - для **аналітичної** моделі – таблиця та діаграми;
 - для **імітаційної** моделі – середні показники за 100 прогонів моделі – таблиці, гістограми та діаграми;
14. Аналіз отриманих результатів. Порівняння результатів двох моделей.
15. Висновки щодо функціонування досліджуваної системи та рекомендації щодо організації роботи багатоканальної системи масового обслуговування.

Питання для самоконтролю

1. Що таке система масового обслуговування?
2. Назвіть можливі складові систем масового обслуговування.
3. Які переваги має аналітичне моделювання систем масового обслуговування?
4. Чому аналітичне дослідження можливе лише для нескладних систем?
5. Які основні переваги імітаційного моделювання порівняно з аналітичним?
6. Які недоліки має імітаційне моделювання систем масового обслуговування?
7. Назвіть особливості систем, які можуть бути враховані при імітаційному моделюванні.
8. Назвіть основні види систем масового обслуговування.
9. Які статистичні дані збираються під час імітаційного моделювання систем масового обслуговування?
10. У чому різниця між часом очікування та часом обслуговування? Як ці показники впливають на оцінку ефективності системи?
11. Як зміна параметрів і структури моделі допомагає знайти оптимальні проектні рішення?

Лабораторна робота №5

Алгоритми класифікації

Мета – ознайомитися з алгоритмами класифікації, набути практичних навичок попереднього підготування набору даних та опанувати створення класифікаторів за допомогою бібліотеки Scikit-Learn.

Теоретичні відомості

Класифікація – це процес розподілу множини об'єктів на попередньо задані групи (класи) за певними ознаками.

Клас – це підмножина об'єктів, у якій вони схожі один на одного за певними ознаками.

Ознаки класифікації – це властивості чи характеристики об'єктів, вибрані для визначення схожості або відмінності між цими об'єктами.

Оцінка ефективності класифікатора – це процес визначення того, наскільки правильно модель або алгоритм класифікації визначає класи для набору тестових даних порівняно зі справжніми мітками класів.

Основні етапи підготування даних до класифікації:

1. Збір даних;
2. Очищення даних;
3. Відбір даних;
4. Аналіз і відбір лише значимих для моделі ознак, видалення зайвих або корелюючих ознак;
5. Кодування категоріальних ознак;
6. Масштабування числових ознак;
7. Розділення даних на навчальну (тренувальну) та тестову частини.

Основні методи класифікації:

- K-Nearest Neighbors – метод k-найближчих сусідів;
- Decision Tree Classifier – класифікатор дерева рішень;
- Support Vector Machines – метод опорних векторів;

- Logistic Regression – логістична регресія;
- Random Forests – класифікатор випадковий ліс;
- Naive Bayes – наївний байєсівський метод;
- Linear Discriminant Analysis – лінійний дискримінантний аналіз;

Бібліотека Scikit-Learn

Бібліотека Scikit-Learn – це потужна бібліотека Python для машинного навчання, яка надає інструменти для підготовки даних, навчання класифікаторів і оцінювання їхньої ефективності (див. лекцію 9 і 10).

ClassifierMixin у бібліотеці Scikit-Learn є базовим класом для класифікаторів. Основні методи та поля ClassifierMixin:

- `fit(X_train, y)` – метод для навчання моделі на тренувальному (навчальному) наборі даних `X_train` та відповідних мітках класів `y`;
- `predict(X_new)` – метод для передбачення класів для нових даних `X_new` після того, як модель була навчена;
- `score(X_test, y)` – метод для оцінки якості моделі на тестових даних `X_test` та відповідних мітках класів `y`, повертає середню похибку на даній тестовій вибірці;
- `classes_` – поле, яке містить унікальні класи, з якими працює модель.

Основні класи та методи в модулі `sklearn.preprocessing`:

- `StandardScaler` стандартизує дані, забезпечуючи їхнє середнє значення 0 та стандартне відхилення 1;
- `MinMaxScaler` масштабує дані до заданого діапазону, зазвичай `[0, 1]`;
- `RobustScaler` масштабує дані, використовуючи медіану та інтерквартильний розмах;
- `Normalizer` нормалізує дані;
- `MaxAbsScaler` масштабує дані відповідно до їх максимального абсолютного значення;

- Binarizer перетворює дані у двоїчні значення (0 або 1) відповідно до порогового значення.
- LabelEncoder присвоює числові ідентифікатори категоріям в порядку їх зустрічі.
- OneHotEncoder перетворює категоріальні ознаки в бінарний вектор (One-Hot Encoding).

Приклад розроблення класифікатора

Розглянемо створення класифікатора з використанням бібліотеки Scikit-Learn на прикладі даних з файлу **Iris.csv**. Набір даних Iris містить 150 записів про квіти ірису. Кожна квітка класифікована як один із трьох видів: *Setosa*, *Versicolor* або *Virginica*. У наборі є чотири стовпці з вимірами квітів у сантиметрах: Sepal.length, Sepal.width, Petal.length, Petal.width. Більш детально цей приклад розглянуто на лекції 10.

Відкриємо цей набір програмно і створимо датасет:

```
import pandas as pd
import matplotlib.pyplot as plt
from pandas.plotting import scatter_matrix
from sklearn.neighbors import KNeighborsClassifier

path = "C:\Python\iris.csv"
dataset = pd.read_csv(path)
```

Для виведення матриці графіків розсіювання скористаємося методом pandas.plotting.scatter_matrix():

```
color_wheel = {"Setosa": "red",
               "Versicolor": "blue",
               "Virginica": "green"}
colors = dataset["variety"].map(lambda x: color_wheel.get(x))
scatter_matrix(dataset, color=colors)
plt.show()
```

Матриця графіків розсіювання дає змогу попарно проаналізувати взаємозв'язки між усіма ознаками набору даних, фактично розгортаючи багатовимірний простір у серію двовимірних графіків (рис. 5.1).

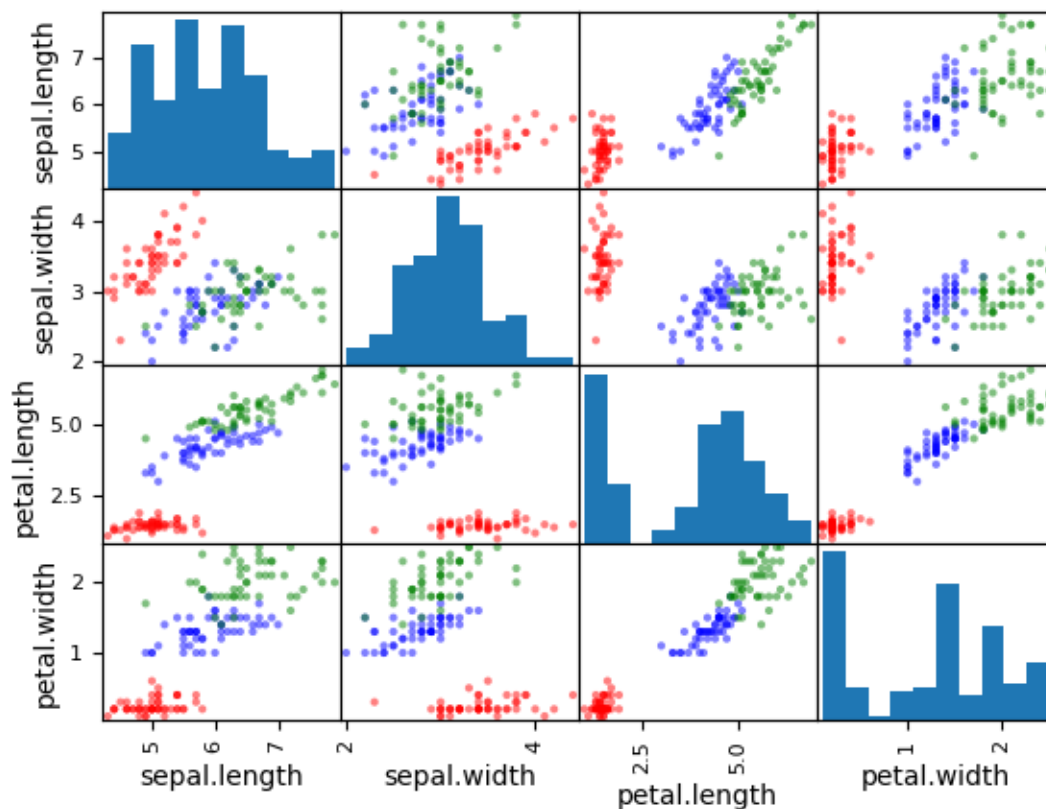


Рисунок 5.1 – Матриця графіків розсіювання

Кожен графік у матриці (крім діагональних) є своєрідним «зрізом» багатовимірнього простору даних, що надає можливість детально дослідити взаємодію між двома конкретними параметрами. Діагональні графіки фактично є гістограмами, які показують розподіл значень для кожної ознаки.

Метод `describe()` в бібліотеці `Pandas` використовується для генерації описової статистики для числових даних у `DataFrame`:

```
print("Statistical information:")
print(dataset.describe(), "\n")
```

```
Statistical information:
      sepal.length  sepal.width  petal.length  petal.width
count    150.000000    150.000000    150.000000    150.000000
mean         5.843333         3.057333         3.758000         1.199333
std          0.828066         0.435866         1.765298         0.762238
min          4.300000         2.000000         1.000000         0.100000
25%          5.100000         2.800000         1.600000         0.300000
50%          5.800000         3.000000         4.350000         1.300000
75%          6.400000         3.300000         5.100000         1.800000
max          7.900000         4.400000         6.900000         2.500000
```

За допомогою групування – методу `groupby()` – можна порахувати, скільки об'єктів міститься у кожному класі:

```
| print("Class distribution by ", dataset.groupby('variety').size(), "\n")
|
| Class distribution by  variety
| Setosa              50
| Versicolor         50
| Virginica          50
| dtype: int64
```

Розділимо датасет на ознаки (чотири розміри квітки) та класи (стовпець «Variety»):

```
| iris_size = dataset.iloc[:, :-1].values
| iris_class = dataset.iloc[:, 4].values
```

Розділимо за допомогою модуля train_test_split набір даних на дві частини – навчальну (80%) і тестову (20%).

```
| from sklearn.model_selection import train_test_split
| Size_train, Size_test, Class_train, Class_test = \
| train_test_split(iris_size, iris_class, test_size=0.2)
```

Метод fit використовується для навчання моделі (адаптації під навчальні дані):

```
| classifier = KNeighborsClassifier(n_neighbors = 3, metric='euclidean')
| classifier.fit(Size_train, Class_train)
```

Застосуємо метод predict() створеного класифікатора для передбачення класів тестових даних:

```
| test_predict=classifier.predict(Size_test)
```

Метод score використовується для обчислення частки правильних передбачень серед тестових даних:

```
| score_knn = classifier.score(Size_test, Class_test)
| print("Correct predictions proportion = ", score_knn, "\n")
|
| Correct predictions proportion =  0.9666666666666667
```

Функція classification_report() у sklearn використовується для відображення текстових звітів за основними показниками класифікації:

```
| print(classification_report(Class_test, test_predict), "\n")
|
|              precision    recall  f1-score   support
|
|   Setosa               1.00      1.00      1.00         9
|  Versicolor           1.00      0.93      0.96        14
|   Virginica           0.88      1.00      0.93         7
|
|   accuracy                   0.97        30
```

Зі звіту видно, що класифікатор точно визначив об'єкти класу *Setosa*, а от точність визначення об'єктів класу *Versicolor* трошки нижча – є якась плутанина з класом *Virginica*.

Побудуємо матрицю плутанини (*confusion_matrix*):

```
from sklearn.metrics import classification_report, confusion_matrix, ConfusionMatrixDisplay
cm=confusion_matrix(Class_test, test_predict)
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=classifier.classes_)
disp.plot()
plt.show()
```

На рис. 5.2 показано результат застосування методу *ConfusionMatrixDisplay* для візуалізації *confusion_matrix*.

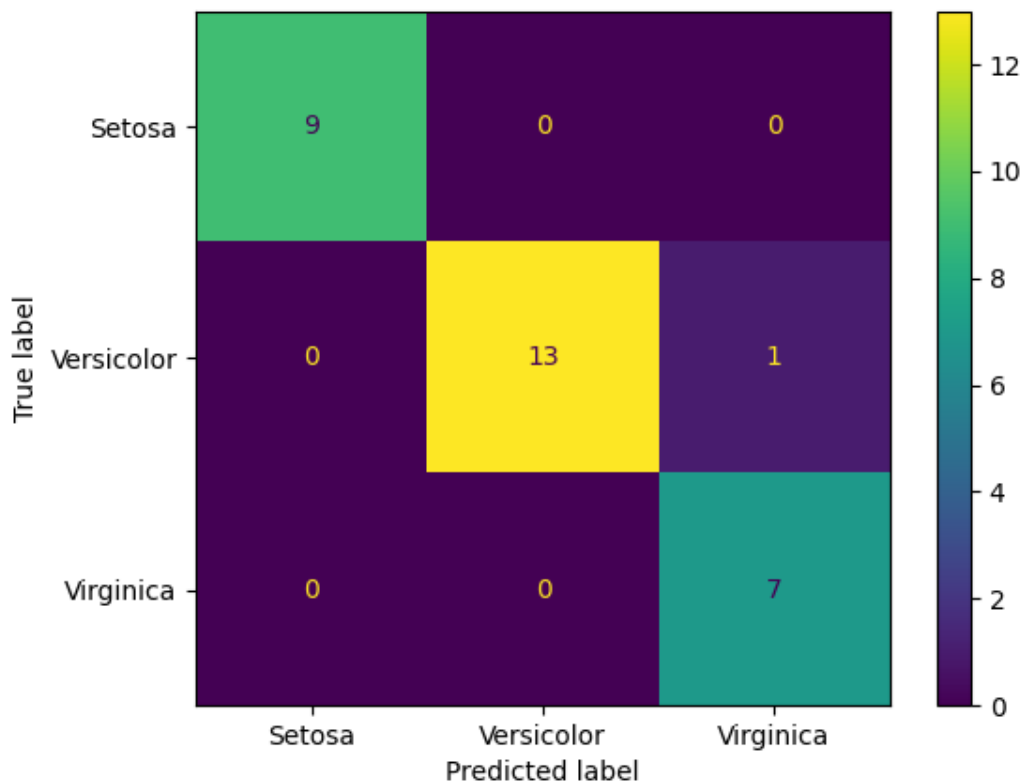


Рисунок 5.2 – Приклад візуалізації матриці плутанини

З матриці плутанини видно, що класифікатор правильно визначив усі 9 об'єктів класу *Setosa* та усі 7 об'єктів класу *Virginica*, а от з 14 об'єктів класу *Versicolor* один об'єкт помилково відніс до класу *Virginica*.

Завдання на лабораторну роботу

1. Проаналізувати та підготувати набір даних до класифікації (за варіантом). Вивести статистичну інформацію числових даних з *DataFrame* (метод

describe бібліотеки Pandas). Дослідити розподіл даних: скільки об'єктів (записів) у вибірці, скільки всього ознак у наборі і що це за ознаки, який зі стовпців містить мітки класів, співвідношення класів. Порахувати кількість записів, які містять пропущені значення та видалити їх. Порахувати кількість дублікатів та видалити їх. Виконати кодування категоріальних ознак за наявності. Порахувати кількість записів, які залишились у вибірці після очищення та відбору даних для класифікації. Порахувати кількість екземплярів у кожному класі.

2. Візуалізувати матрицю графіків розсіювання. Проаналізувати зв'язок між ознаками.
3. Розділити дані з файлу на тренувальний (навчальний) та тестовий набори. Можна 80% віднести до тренувального (навчального) набору, а 20% до тестового.
4. Розробити класифікатор із застосуванням Scikit-Learn (вибрати один з розглянутих основних методів класифікації). Навчити класифікатор на тренувальному наборі даних. Застосувати класифікатор до тестового набору даних.
5. Оцінити точність класифікатора (побудувати звіт щодо продуктивності класифікатора з використанням метрик: точність, повнота, F1-міра).
6. Побудувати, візуалізувати та проаналізувати матрицю плутанини.
7. Виконати масштабування числових ознак та проаналізувати, чи вплинуло це на продуктивність класифікатора.
8. Підібрати метод класифікації та параметри класифікатора, за якого найбільша частка правильних передбачень серед тестових даних.
9. Проаналізувати, що і як саме впливає на точність класифікатора, який навчено саме на цьому наборі даних і зробити висновки.
10. Написати власну функцію, яка розраховує евклідову відстань між двома об'єктами. Вибрати по три об'єкти з кожного класу тестового набору. Для кожного з цих об'єктів визначити найближчого сусіда з навчального набору використовуючи власну функцію розрахунку евклідової відстані.

Порівняти, чи до одного й того ж класу відносяться вибрані об'єкти і їх найближчі сусіди. На основі результатів порівняння зробити висновок про ефективність методу k-найближчих сусідів для класифікації цього набору даних.

Варіанти завдань

Набори даних, необхідні для виконання лабораторної роботи, здобувачі освіти отримують у викладача.

1. **Abalone.csv** – набір даних з 4177 записами. Кожен запис містить вісім характеристик абалона: довжина, ширина, висота, вага, об'єм, товщина зовнішньої оболонки, товщина внутрішньої оболонки та кількість кілець. Класифікація може бути здійснена за статевою приналежністю: чоловіча (M), жіноча (F) або інтерсексуальна (I).

2. **Cancer patient.csv** – це набір даних з 1000 записами*, класифікованими як один із трьох класів (Level).

**Зверніть увагу, що після видалення зайвих ознак і дублікатів лишилося набагато менше записів.*

3. **Car Evaluation.csv** – набір даних з 1728 записами, класифікованими як один із чотирьох класів (Acceptability): дуже хороший, хороший, прийнятний або неприйнятний. Кожен запис містить шість характеристик автомобіля. *Зверніть увагу, що в двох стовпцях з майже числовими даними є кілька значень «Interval[{5, Infinity}]» – просто замініть ці значення на 5 та використовуйте ці ознаки як числові.*

4. **Diamonds.csv** – це набір даних з 53940 записами*, які містять характеристики діамантів, такі як вага (у каратах), колір, чистота, ціна тощо. Кожен запис класифікований за якістю зрізу (cut): задовільний, хороший, дуже добрий, преміальний, ідеальний.

**Можна використовувати зменшену вибірку – файл Diamonds_4000.csv.*

5. **Glass.csv** – це набір даних з 214 записами, які містять характеристики скла. Кожен запис класифікований як один із шести класів (Type).
6. **Pizza.csv** – це набір даних з 300 записами, класифікованими як один із 10 класів (brand).
7. **Seeds.csv** – це набір даних з 210 записами, які містять характеристики насіння трьох видів.
8. **Vehicle.csv** – це набір даних з 846 записами, класифікованими як один із трьох класів (class): car, van, bus.
9. **Vowel.csv** – це набір даних з 990 записами, які містять характеристики голосних звуків. Кожен запис класифікований як один із 11 класів: hid, hId, hed, hEd, had, hAd, hYd, hod, hOd, hud, hUd.
10. **Wine.csv** – набір даних з 178 записами, класифікованими як один із трьох класів (Wine). Кожен запис містить 13 різних характеристик вина.
11. **Yeast.csv** – набір даних з 1484 записами, які містять характеристики дріжджів. Кожен запис класифікований як один із 10 класів (name).

Зміст звіту

1. Титульний аркуш;
2. Результати аналізу та попереднього оброблення набору даних за варіантом;
3. Код програми (може бути оформлений як додаток або прикріплений як окремий текстовий файл);
4. Матриця графіків розсіювання;
5. Звіт щодо продуктивності розробленого класифікатора та аналіз отриманих метрик;
6. Візуалізація та аналіз матриці плутанини;
7. Аналіз впливу масштабування ознак на продуктивність класифікатора;
8. Оптимізація методу та параметрів класифікації;

9. Аналіз факторів впливу на точність класифікації;
10. Реалізація методу найближчого сусіда та його аналіз;
11. Висновки.

Питання для самоконтролю

1. Що таке класифікація? Назвіть основні методи класифікації.
2. Які основні етапи підготування даних для побудови класифікатора?
3. Як правильно оброблювати пропущені значення?
4. Що показує матриця графіків розсіювання і як вона допомагає в аналізі даних?
5. Які проблеми можуть виникати при незбалансованих наборах даних?
6. Для чого розділяють дані на тренувальний та тестовий набори?
7. Назвіть метрики оцінки якості класифікатора.
8. За якою з метрик більш правильно аналізувати ефективність роботи класифікатора?
9. Що таке матриця помилок і яку інформацію вона надає?
10. Як впливає масштабування ознак на роботу різних алгоритмів класифікації?
11. Для чого потрібно виконувати кодування категоріальних ознак?
12. Які параметри можна налаштовувати для підвищення точності класифікатора?
13. Що таке евклідова відстань і як вона використовується в методі найближчого сусіда?
14. Які переваги та недоліки методу найближчого сусіда порівняно з іншими методами класифікації?

Лабораторна робота №6

Кластерний аналіз

Мета – ознайомитися з методами кластерного аналізу, методологією визначення оптимальної кількості кластерів та метриками якості кластеризації. Набути практичних навичок застосування алгоритму k-means та опанувати створення моделей кластеризації за допомогою бібліотеки Scikit-Learn.

Теоретичні відомості

Кластеризація (або кластерний аналіз) – це процес розподілу множини об'єктів на однорідні підмножини (групи) за схожістю певних ознак об'єктів.

Головна **відмінність кластеризації від класифікації** у тому, що перелік груп (класів, кластерів) не заданий, а визначається за алгоритмом.

Алгоритм K-Means групує дані таким чином, щоб розділити набір даних на **n** компактних кластерів з приблизно однаковою дисперсією та розміром. Компактність полягає у тому, що критерій, відомий як інерція або сума квадратів у межах кластера (WCSS), має бути мінімальний за усіма кластерами. Мінімізація інерції призводить до створення компактних кластерів, оскільки об'єкти у кожному кластері мають бути якомога ближчими до його центроїду.

Алгоритм KMeans реалізовано в `sklearn.cluster.KMeans`. Основні параметри:

n_clusters – кількість кластерів, які необхідно сформувати, а також кількість центроїдів для створення, за замовчуванням = 8.

init – спосіб ініціалізації центроїдів (початкові значення), за замовчуванням = 'k-means++'. Це може бути функція ('k-means++', 'random') або масив заданих k центроїдів для ініціалізації (початкові центри).

n_init – кількість разів виконання алгоритму k-mean з різними випадковими центроїдами. Оскільки на результати алгоритму впливає початкове значення, необхідно запустити алгоритм кілька разів, щоб вибрати найкращий результат кластеризації. За замовчуванням `n_init = 'auto'`. У цьому разі кількість

запусків залежить від значення `init`: `n_init = 10`, якщо використовується `init = 'random'`, або `n_init = 1`, якщо використовується `init='k-means++'` або `init` є масивом заданих `k` центроїдів.

max_iter – максимальна кількість ітерацій алгоритму `k-mean` за один пробіг, за замовчуванням `= 300`.

Після успішного завершення кластеризації `k-means` клас `KMeans` матиме такі важливі атрибути для отримання результатів:

labels_ – надає прогнозовані мітки класів (кластерів) для кожної точки даних.

cluster_centers_ – надає розташування центроїдів кожного кластера.

inertia_ – загальна внутрішньо-кластерна сума квадратів для всіх кластерів (сума квадратів відстаней кожного зразка до найближчого центроїда).

n_iter_ – кількість ітерацій, які виконує алгоритм `k-means`, щоб отримати мінімальну суму квадратів всередині кластера.

Для визначення оптимальної кількості кластерів можна використовувати такі метрики:

- Індекс інерції (WCSS). Точка на графіку, де відбувається різке зниження градієнта (метод ліктя), вказує на оптимальну кількість кластерів для даного набору даних.
- Індекс Данна (Dunn). Чим вище значення індексу Данна, тим краще кластеризація.
- Індекс силуету (Silhouette). Значення силуетного індексу знаходяться в діапазоні від -1 до 1, де значення, близьке до 1, означає добре кластеризовані дані.
- Індекс Девіса-Булдіна (Davies-Bouldin). Чим нижче значення індексу Девіса-Булдіна, тим краще кластеризація.
- Індекс Калінського-Харабаша (Calinski-Harabasz). Чим вище значення цього індексу, тим краще кластеризація.

Приклад кластерного аналізу даних

Розглянемо кластерний аналіз з використанням бібліотеки Scikit-Learn на прикладі даних з файлу **Iris.csv**. Відкриємо цей набір програмно і створимо датасет:

```
import matplotlib.pyplot as plt
import pandas as pd
from collections import Counter
from sklearn import metrics
from sklearn.cluster import KMeans
from sklearn.preprocessing import MinMaxScaler
from tabulate import tabulate

path = "C:\Python\iris.csv"
dataset = pd.read_csv(path)
```

Припустимо, що кількість видів ірисів у наборі даних невідома, і необхідно розподілити дані на п'ять кластерів. Оскільки мітки класів у цьому випадку не використовуються, відокремимо останній стовпець набору даних, де вони містяться::

```
iris_size = dataset.iloc[:, :-1].values
iris_class = dataset.iloc[:, 4].values
```

Застосуємо до даних алгоритм KMeans:

```
clusterer = KMeans(n_clusters=5)
clusterer.fit(iris_size)
labels = clusterer.labels_
```

Розділимо іриси на кластери за характеристиками:

```
predictions = clusterer.predict(iris_size)
dataset["cluster"] = predictions
print("Результати:")
print(dataset, "\n")
```

	sepal.length	sepal.width	petal.length	petal.width	variety	cluster
0	5.1	3.5	1.4	0.2	Setosa	1
1	4.9	3.0	1.4	0.2	Setosa	1
2	4.7	3.2	1.3	0.2	Setosa	1
3	4.6	3.1	1.5	0.2	Setosa	1
4	5.0	3.6	1.4	0.2	Setosa	1
...	...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Virginica	4
146	6.3	2.5	5.0	1.9	Virginica	3
147	6.5	3.0	5.2	2.0	Virginica	4
148	6.2	3.4	5.4	2.3	Virginica	4
149	5.9	3.0	5.1	1.8	Virginica	3

Визначимо кількість ірисів у кожному кластері:

```
count_cluster = Counter(labels)
print("Кількість у кластерах: ")
print(count_cluster, "\n")

Кількість у кластерах:
Counter({1: 50, 3: 39, 2: 25, 4: 24, 0: 12})
```

Згрупуємо дані за стовпцем «cluster» і порахуємо кількість входжень кожної мітки класів за стовпцем «variety»:

```
cluster_content = dataset.groupby(["cluster", "variety"]).size().unstack(fill_value=0)
cluster_content['Total'] = cluster_content.sum(axis=1)
cluster_content.loc['Total'] = cluster_content.sum()
print(tabulate(cluster_content, headers="keys", tablefmt="psql"))
```

cluster	Setosa	Versicolor	Virginica	Total
0	0	0	12	12
1	50	0	0	50
2	0	24	1	25
3	0	26	13	39
4	0	0	24	24
Total	50	50	50	150

Визначимо розташування центроїдів кожного кластера:

```
centroids = clusterer.cluster_centers_
print("Координати усіх центроїдів: ")
print(centroids, "\n")
```

Оскільки у датасеті є чотири характеристики об'єктів та вибрано $n_clusters = 5$, отримаємо 4 координати для 5 центроїдів:

```
Координати усіх центроїдів:
[[7.475      3.125      6.3      2.05      ]
 [5.006      3.428      1.462      0.246      ]
 [5.508      2.6       3.908      1.204      ]
 [6.20769231 2.85384615 4.74615385 1.56410256]
 [6.52916667 3.05833333 5.50833333 2.1625      ]]
```

Зобразимо результати кластеризації для однієї пари параметрів і застосуємо різні кольори для різних груп кластерів:

```
fig, ax = plt.subplots()
scatter1 = ax.scatter(iris_size[:, 0], iris_size[:, 1],
                      c=predictions, s=15, cmap="brg")
handles, labels = scatter1.legend_elements()
legend1 = ax.legend(handles, labels, loc="upper right")
ax.add_artist(legend1)
scatter2 = ax.scatter(centroids[:, 0], centroids[:, 1], marker="x",
                      c="purple", s=200, linewidths=3, label="centroids")
plt.legend(loc="lower right")
plt.xlabel(f"{dataset.columns[0]}")
plt.ylabel(f"{dataset.columns[1]}")
plt.show()
```

В результаті отримаємо графік розсіювання для однієї пари ознак – ширина і довжина пелюстки (рис. 6.1). Точки (об'єкти-ірис) розфарбовані за кластерами, а центроїди кластерів позначені хрестиками.

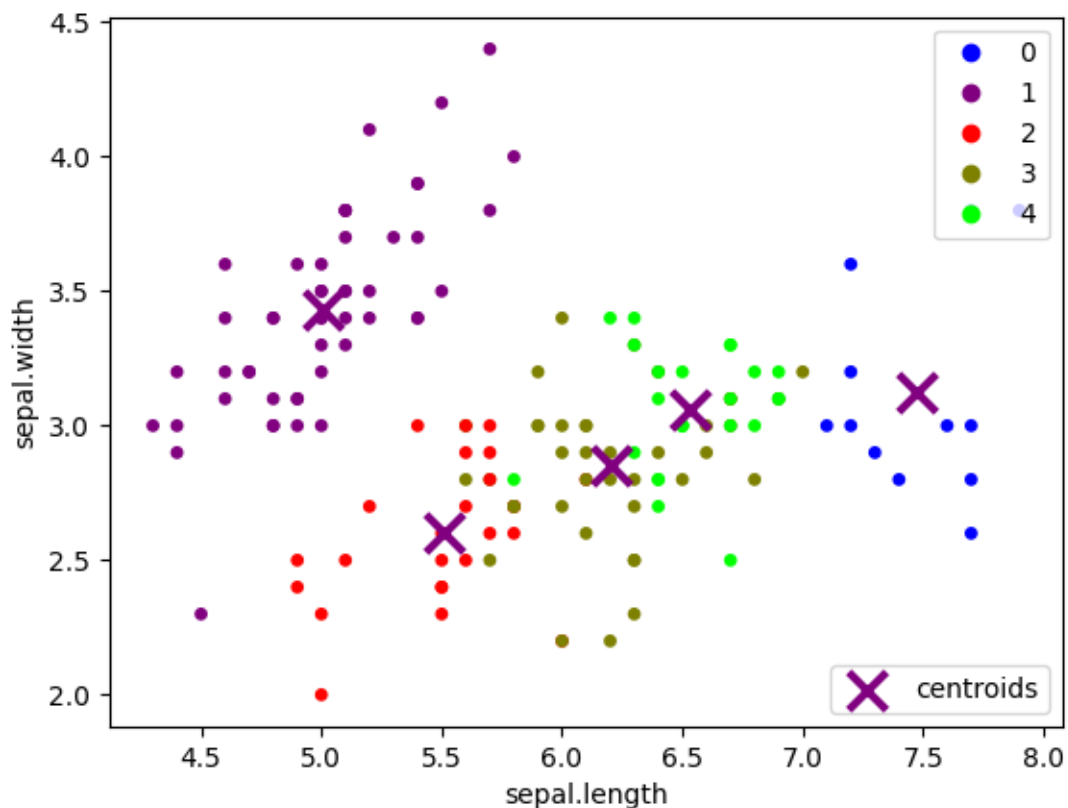


Рисунок 6.1 – Приклад графіка розсіювання для однієї пари ознак

Варто зазначити, що за кожного запуску програми кластери буде пронумеровано у просторі по різному, якщо параметр `random_state` не зафіксовано.

Більш детально цей приклад розглянуто на лекції 11.

Завдання на лабораторну роботу

1. Підготувати набір даних для кластеризації (за варіантом).
2. Провести кластеризацію даних за алгоритмом k-means із застосуванням бібліотеки Scikit-Learn. Кількість кластерів вибрати таку ж, як і кількість класів у попередній лабораторній роботі.
3. Побудувати графіки результатів кластеризації для декількох параметрів попарно (не менше 4-х), додати на графіки центроїди.

4. Порахувати кількість екземплярів у кожному кластері та проаналізувати, скільки об'єктів кожного з класів потрапило у кожний кластер. Вивести результати у вигляді таблиці.
5. Визначити оптимальну кількість кластерів за допомогою трьох-чотирьох метрик. Вивести таблицю та побудувати графіки залежності показників якості кластеризації (метрик) від кількості кластерів. Проаналізувати отримані результати.
6. Дослідити вплив різних методів масштабування ознак (StandardScaler, MinMaxScaler) на результат кластеризації та на оптимальну кількість кластерів. Побудувати графіки залежності показників якості кластеризації (для 3-4 метрик) від кількості кластерів для різних методів масштабування.
7. Зробити висновки щодо проведеної кластеризації, ефективності використаного алгоритму та впливу параметрів на якість кластеризації.

Варіанти завдань

Набори даних, необхідні для виконання лабораторної роботи, здобувачі освіти отримують у викладача.

1. **Abalone.csv** – набір даних з 4177 записами. Кожен запис містить вісім характеристик абалона: довжина, ширина, висота, вага, об'єм, товщина зовнішньої оболонки, товщина внутрішньої оболонки та кількість кілець. Стовець з даними щодо статевої приналежності абалонів у кластерному аналізі не використовувати.
2. **Cancer patient.csv** – це набір даних з 1000 записами. Стовець Level у кластерному аналізі не використовувати. *Зверніть увагу, що після видалення зайвих ознак і дублікатів лишилось набагато менше записів.*
3. **Car Evaluation.csv** – набір даних з 1728 записами. Кожен запис містить шість характеристик автомобіля. *Зверніть увагу, що в двох стовпцях з майже числовими даними є кілька значень «Interval[{5, Infinity}]» – просто*

замініть ці значення на 5 та використовуйте ці ознаки як числові. Стовець Acceptability у кластерному аналізі не використовувати.

4. **Diamonds.csv** – це набір даних з 53940 записами*, які містять характеристики діамантів, такі як вага (у каратах), колір, чистота, ціна тощо. Стовець cut у кластерному аналізі не використовувати.
*Можете використовувати зменшену вибірку – файл Diamonds_4000.csv.
5. **Glass.csv** – це набір даних з 214 записами, які містять характеристики скла. Стовець Type у кластерному аналізі не використовувати.
6. **Pizza.csv** – це набір даних з 300 записами. Стовець brand у кластерному аналізі не використовувати.
7. **Seeds.csv** – це набір даних з 210 записами. Стовець class у кластерному аналізі не використовувати.
8. **Vehicle.csv** – це набір даних з 846 записами. Стовець class у кластерному аналізі не використовувати.
9. **Vowel.csv** – це набір даних з 990 записами, які містять характеристики голосних звуків. Стовець Class у кластерному аналізі не використовувати.
10. **Wine.csv** – набір даних з 178 записами, які містять характеристики вин. Стовець Wine у кластерному аналізі не використовувати.
11. **Yeast.csv** – набір даних з 1484 записами, які містять характеристики дріжджів. Стовець name у кластерному аналізі не використовувати.

Зміст звіту

1. Титульний аркуш;
2. Короткий опис набору даних;
3. Код програми (може бути оформлений як додаток або прикріплений як окремий текстовий файл);

4. Графіки результатів кластеризації (не менше 4-х);
5. Таблиця розподілу об'єктів у кластерах;
6. Таблиця та графіки залежності показників якості кластеризації (для 3-4 метрик) від кількості кластерів;
7. Графіки залежності показників якості кластеризації (для 3-4 метрик) від кількості кластерів для різних методів масштабування ознак (StandardScaler, MinMaxScaler);
8. Аналіз впливу різних методів масштабування ознак на якість кластеризації та на оптимальну кількість кластерів.
9. Висновки.

Питання для самоконтролю

1. Що таке кластеризація та чим вона відрізняється від класифікації?
2. У чому полягає основний принцип роботи алгоритму k-means?
3. Який вплив має параметр k (кількість кластерів) на результати кластеризації?
4. Що таке центроїди кластерів і як обчислюються їх координати?
5. Які метрики використовуються для визначення оптимальної кількості кластерів?
6. Що таке внутрішньокластерна і міжкластерна відстані?
7. Що таке радіус кластера і як він визначається?
8. Що показує індекс силуету (Silhouette) та як інтерпретувати його значення?
9. Що таке метод ліктя та як його застосовувати?
10. Як впливає масштабування ознак на результати кластеризації?
11. У чому полягає особливість методу k-means++ і які переваги він має перед звичайною випадковою ініціалізацією центроїдів?

Лабораторна робота №7

Оптимізаційне моделювання

Мета — ознайомитися з чисельними методами пошуку локального мінімуму функцій однієї та двох змінних та візуальною інтерпретацією процесу оптимізації.

Теоретичні відомості

Оптимізаційне моделювання передбачає побудову математичної моделі досліджуваного процесу або системи з такими значеннями параметрів моделі, які відповідають мінімуму або максимуму певної цільової функції. Обидва ці поняття – максимум і мінімум – об'єднуються єдиним терміном «екстремум».

Цільова функція – це залежність критерію оптимальності від параметрів процесу або системи, які підлягають оптимізації із обов'язковим зазначенням напрямку екстремуму.

Критерій оптимальності – це деякий показник, на підставі якого проводиться кількісне оцінювання процесу або системи.

Задачею оптимізації конкретного процесу є пошук екстремуму його цільової функції. Розв'язати таку задачу означає знайти її оптимальний розв'язок чи встановити, що його немає. У задачах багатокритеріальної оптимізації використовуються кілька цільових функцій, кожна з яких відповідає певному критерію оптимальності.

Вхідні дані для оптимізації задач називають **ресурсами оптимізації**. Важливо обережно підходити до визначення ресурсів оптимізації. Що більше аргументів цільової функції, то складніша задача і важче досягти оптимальних значень.

Локальний екстремум функції – це точка, в якій функція досягає мінімального або максимального значення в деякому околі цієї точки.

Глобальний екстремум функції – це точка, у якій функція досягає абсолютного максимуму чи мінімуму в усій своїй області визначення.

Знаходження мінімумів засобами бібліотек Python

Для вирішення завдань оптимізації в Python зручно використовувати бібліотеку [scipy.optimize](#). В бібліотеці є методи для умовної та безумовної мінімізації скалярних функцій багатьох аргументів, глобальної оптимізації, апроксимації кривих, аналізу даних методом найменших квадратів, пошуку коренів нелінійних алгебраїчних рівнянь та систем лінійних алгебраїчних рівнянь.

Чисельне знаходження локалізованого мінімуму скалярної функції однієї змінної засобами бібліотеки `scipy.optimize` реалізовано в методі [minimize_scalar](#).
`minimize_scalar(fun, bracket=None, bounds=None, args=(), method='brent', tol=None, options=None)`

Опція `method` задає метод (Brent, Bounded, Golden) чисельного розрахунку мінімуму. За замовчуванням встановлено метод Брендта ('brent').

Розглянемо знаходження мінімуму функції:

$$f(x) = 16 \cdot x \cdot \exp\left(-\frac{1}{2} \cdot x^2 - 1\right) - 1$$

за допомогою `minimize_scalar` бібліотеки `scipy.optimize`:

```
x_min = minimize_scalar(f, bracket=[a, b]).x  
f_min = f(x_min)
```

де f – задана функція, x – вхідна змінна, $[a, b]$ – відрізок пошуку.

Зобразимо саму функцію і точку мінімуму на графіку (рис. 7.1) з використанням бібліотеки `matplotlib.pyplot`.

```
plt.plot(x, f(x), c='k', label=r'$f(x)$')  
plt.scatter(x_min, f_min, c='r', label=text_min)  
plt.title('$f(x) = 16*x*\exp(-\frac{1}{2}x^2-1)-1$')
```

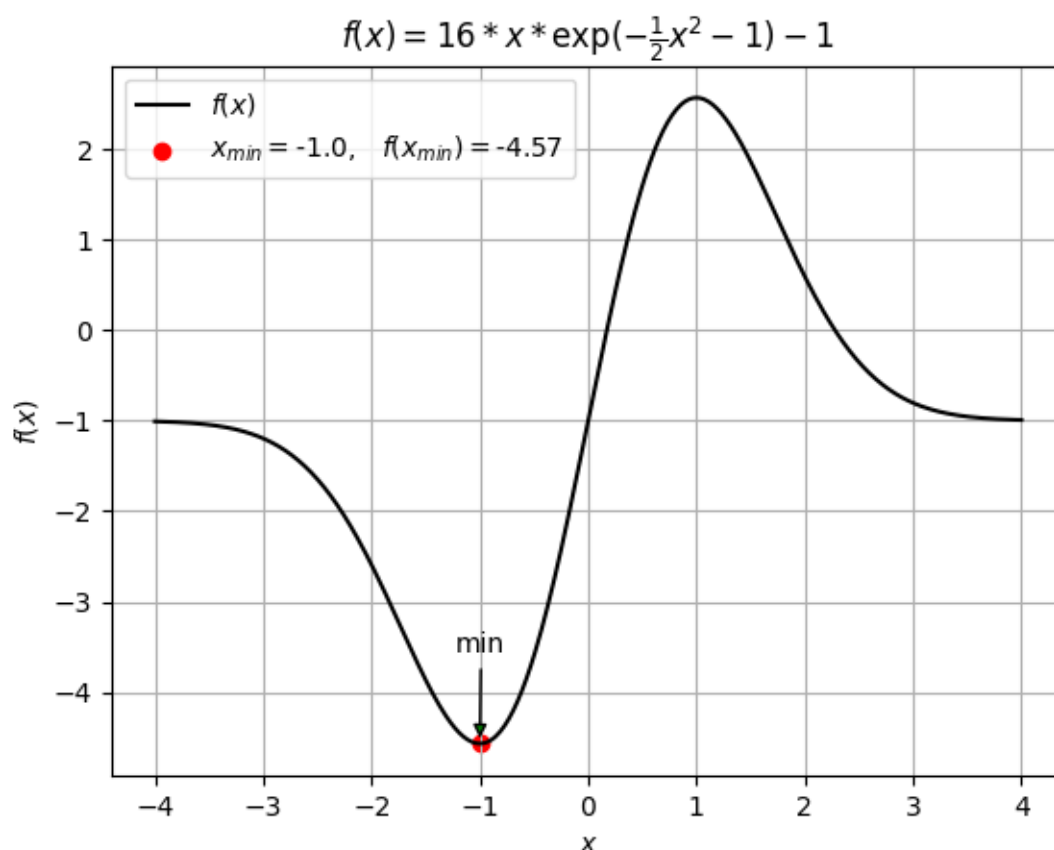


Рисунок 7.1 – Приклад знаходження мінімуму функції однієї змінної

Мінімізація скалярних функцій кількох змінних забезпечується методом

[minimize](#)

```
minimize(fun, x0, args=(), method=None, jac=None, hess=None, hessp=None,
        bounds=None, constraints=(), tol=None, callback=None, options=None)
```

Опцією method передбачено велику кількість різних методів розрахунку:

- метод симплексного спуску [Нелдера – Міда](#) ('Nelder-Mead'),
- метод сполучених напрямів Пауелла ('Powell'),
- метод [Бройдена-Флетчера-Гольдфарба-Шанно](#) ('BFGS'),
- метод сполучених [градієнтів Ньютона](#) ('Newton-CG') тощо.

Розглянемо знаходження мінімуму функції:

$$f(x, y) = 16 \cdot x \cdot \exp\left(-\frac{1}{2} \cdot x^2 - \frac{1}{5} \cdot y^2 - 1\right)$$

за допомогою minimize бібліотеки scipy.optimize:

```

X, Y = np.meshgrid(x, y)
Z = f([X, Y])
x_min = minimize(f, (0, 0), method='Nelder-Mead').x
Z_min = f(x_min)

```

Зверніть увагу, що `minimize` приймає у якості аргументів не діапазон пошуку, а *відправну точку*. У цьому прикладі вибрано відправну точку (0, 0).

Зобразимо саму функцію і точку мінімуму на графіку (рис. 7.2) з використанням бібліотеки `matplotlib.pyplot`:

```

ax.plot_surface(X, Y, Z, edgecolor='grey', lw=0.5, rstride=8,
                cstride=8, alpha=0.8, cmap='coolwarm')
ax.scatter(x_min[0], x_min[1], Z_min, s=50, c='k', marker='o')
p1=ax.contour(X, Y, Z, zdir='z', offset=-8, cmap='coolwarm')
p2=ax.contour(X, Y, Z, zdir='x', offset=-3, cmap='coolwarm')
p3=ax.contour(X, Y, Z, zdir='y', offset=6, cmap='coolwarm')
ax.set(xlim=(-3, 3), ylim=(-6, 6), zlim=(-8, 6),
       xlabel='X', ylabel='Y', zlabel='Z')
plt.title(text_func)
plt.show()

```

$$Z = 16 * X * \exp(-\frac{1}{2}X^2 - \frac{1}{5}Y^2 - 1)$$

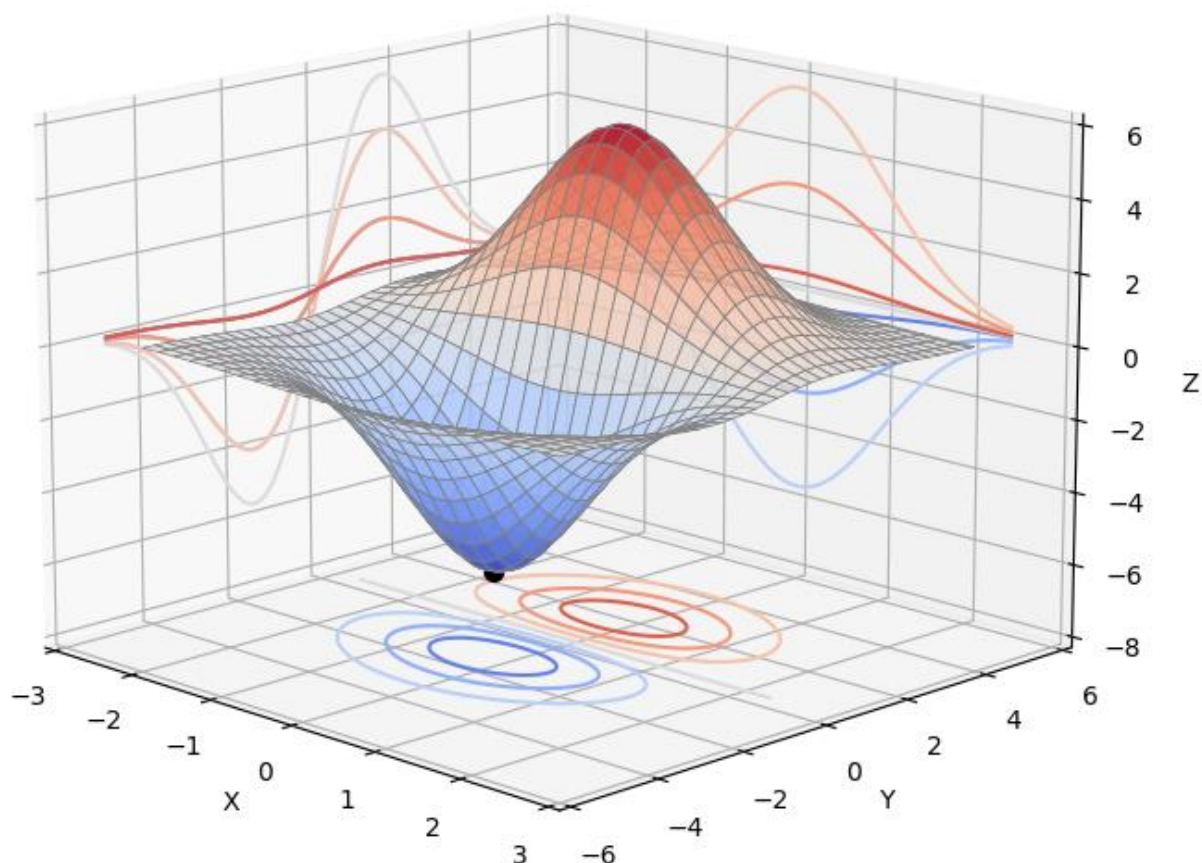


Рисунок 7.2 – Приклад знаходження мінімуму функції двох змінних

Фрагмент коду для побудови карти ліній рівня функції двох змінних для площини XOY:

```
ax2 = fig.add_subplot(132)
plt.contourf(X, Y, Z, cmap='coolwarm', levels=20)
plt.colorbar()
plt.scatter(x_min[0], x_min[1], s=40, c='k', marker='x',
            label = 'min', linewidths=2)
plt.xlabel('X')
plt.ylabel('Y')
plt.legend()
```

Зобразимо карти ліній рівня функції двох змінних (рис. 7.3) для трьох площин: YOZ, XOY, XOZ.

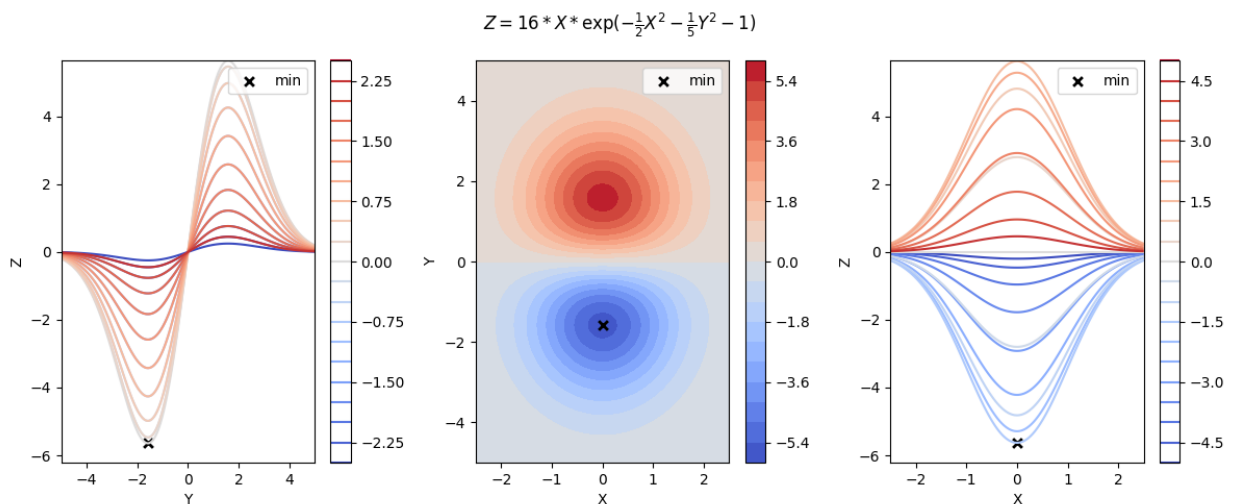


Рисунок 7.3 – Карти ліній рівня функції двох змінних

Лінія рівня – це множина всіх точок на площині, в яких функція набуває одного і того ж значення. Іншими словами, це крива, вздовж якої значення функції є константою.

Карти ліній рівня дають змогу візуалізувати тривимірну інформацію про функцію на двовимірній площині, що спрощує її аналіз. За допомогою карт ліній рівня можна виявити екстремуми функції, області зростання та спадання, а також оцінити загальну форму поверхні, заданої цією функцією.

Завдання на лабораторну роботу

1. Написати програму для чисельного знаходження локалізованого мінімуму скалярної функції однієї змінної (за варіантом). Підібрати область пошуку.
2. Написати програму для чисельного знаходження локалізованого мінімуму скалярної функції двох змінних. Підібрати область пошуку.
3. Побудувати двовимірний графік функції однієї змінної з відміченим мінімумом. Графік має містити підписи осей, заголовок і легенду.
4. Побудувати тривимірний графік функції двох змінних з відміченим мінімумом. Підібрати належну кольорову схему та ракурс огляду графіка, так щоб було видно форму поверхні та розташування екстремумів функції.
5. Побудувати три карти ліній рівня (для трьох площин: YOZ, XOY, XOZ) функції двох змінних з відміченим мінімумом на кожній з карт.
6. Розглянути і порівняти методи пошуку глобального мінімуму функції однієї змінної.
7. Відмітити на графіку всі локальні та глобальний мінімум різними маркерами, відмітити на графіку всі локальні та глобальний максимум різними маркерами.
8. Зробити висновки щодо ефективності використаних методів оптимізації.

Варіанти завдань

1. Функція однієї змінної:

$$f(x) = \left(1 + \frac{\cos(6x)}{2}\right) \cdot \left(\frac{\sin(x)}{1 + x^2}\right);$$

Функція двох змінних:

$$f(x, y) = 1 + x \cdot \sin(y^2);$$

2. Функція однієї змінної:

$$f(x) = \sin(3x) + \frac{x^2}{2};$$

Функція двох змінних:

$$f(x, y) = \sin^2(x) - \cos^2(y) + \sin(x \cdot y) + \frac{x^2 + y^2}{10};$$

3. Функція однієї змінної:

$$f(x) = \sin(5x) \cdot e^{-x^2} - \frac{1}{10}x^3 + \cos(2x);$$

Функція двох змінних:

$$f(x, y) = \sin(x) + 2 \sin(y);$$

4. Функція однієї змінної:

$$f(x) = x \cdot \sin\left(\frac{1}{x}\right);$$

Функція двох змінних:

$$f(x, y) = (x - 1)^2 + (y - 1)^2 + \sin(3x) \cdot \sin(3y);$$

5. Функція однієї змінної:

$$f(x) = \sin(3x) + \sin(5x) + x + 2;$$

Функція двох змінних:

$$f(x, y) = x^4 - 3x^3 - \frac{3}{1 + y^2} + 4x;$$

6. Функція однієї змінної:

$$f(x) = 1 + x \cdot \sin(x^2);$$

Функція двох змінних:

$$f(x, y) = x^2 + y^2 - \cos(18x) - \cos(18y);$$

7. Функція однієї змінної:

$$f(x) = (x - 2)^2 + \sin(5x);$$

Функція двох змінних:

$$f(x, y) = \sin \frac{(x + y)}{1 + x^2 + y^2};$$

8. Функція однієї змінної:

$$f(x) = 2x^2 \cdot \sin(x) + \cos(x);$$

Функція двох змінних:

$$f(x, y) = \sin(x) + \sin(y) + \frac{x^2 + y^2}{10};$$

9. Функція однієї змінної:

$$f(x) = x^2 - \cos(18x);$$

Функція двох змінних:

$$f(x, y) = 2 + \sin(2 \sin(y) + \cos(3y));$$

10. Функція однієї змінної:

$$f(x) = x^5 \cdot \sin(x) \cdot e^x;$$

Функція двох змінних:

$$f(x, y) = x^2 + y^2 + 10 \sin(x) \cdot \cos(y);$$

11. Функція однієї змінної:

$$f(x) = 3 \cdot (x - 2) \cdot \sin(x);$$

Функція двох змінних:

$$f(x, y) = \sin(y) \cdot \cos(y) + \frac{x^2 + y^2}{20};$$

Зміст звіту

1. Титульний аркуш;
2. Функції однієї та двох змінних за варіантом;
3. Код програми для знаходження мінімуму функції однієї змінної;
4. Графік функції однієї змінної з відміченим мінімумом;
5. Код програми для знаходження мінімуму функції двох змінних;
6. Тривимірний графік функції двох змінних з відміченим мінімумом;
7. Три графіки – карти ліній рівня функції двох змінних для площин YOZ, XOY, XOZ;
8. Код програми для знаходження всіх локальних та глобальних екстремумів функції однієї змінної;

9. Графік функції з відміченими різними маркерами глобальними та усіма локальними екстремумами функції однієї змінної;
10. Висновки.

Питання для самоконтролю

1. Що таке оптимізаційне моделювання та яка його мета?
2. Що таке цільова функція?
3. Що розуміють під терміном «критерій оптимальності»?
4. У чому полягає різниця між локальним та глобальним екстремумом функції?
5. Що таке область пошуку і як правильно її визначити?
6. Як можна використати функцію пошуку мінімуму для знаходження максимуму функції?
7. Які переваги та недоліки чисельних методів оптимізації порівняно з аналітичними?
8. Як впливає вибір початкової точки на результат пошуку екстремуму?
9. Як можна визначити, що знайдений екстремум є глобальним, а не локальним?
10. Що таке лінії рівня функції та для чого використовуються їх карти?

ЛІТЕРАТУРНІ ДЖЕРЕЛА

1. Снитюк В. Є. Прогнозування. Моделі. Методи. Алгоритми: навчальний посібник. – К: «Маклаут», 2008. – 364 с.
2. Вислоух С. П., Волошко О. В., Тимчик Г. С., Філіппова М. В. Комп'ютерне моделювання процесів та систем. Чисельні методи: підручник. – Київ : КПІ ім. Ігоря Сікорського, Вид-во «Політехніка», 2021. – 228 с.
3. Павленко П. М., Філоненко С. Ф., Чередніков О. М., Трейтяк В. В. Математичне моделювання систем і процесів: навчальний посібник – К: НАУ, 2017. – 392 с.
4. Жученко А. І., Ладієва Л. Р., Піргач М. С., Жураковський Я. Ю. Математичне моделювання процесів і систем: навчальний посібник. – К: КПІ ім. Ігоря Сікорського, 2021. – 351 с.
5. Лавров Є. А., Перхун Л. П., Шендрик В. В., Кузнецов Е. Г., Парфененко Ю. В., Сергієнко В. А. Математичні методи дослідження операцій: підручник. – Суми: Сумський державний університет, 2017. – 212 с.
6. Chollet F. Deep Learning with Python, Second Edition, 2021. – 504 p.
7. Анісімов А. В., Дорошенко А. Ю., Погорілий С. Д., Дорогий Я. Ю. Програмування числових методів мовою Python : підруч. за ред. А. В. Анісімова. – К.: Видавничо-поліграфічний центр "Київський університет", 2014. – 640 с.
8. Андруник В. А., Висоцька В. А., Пасічник В. В., Чирун Л. Б. Чисельні методи в комп'ютерних науках. Львів: Видавництво «Новий світ–2000», 2017. – 470 с.
9. Strzałko J., Grabski J., Stefański A., Perlikowski P., Kapitaniak T. “Dynamics of coin tossing is predictable.” Physics reports, 2008, Vol. 469, № 2, p. 59-92.
<https://doi.org/10.1016/j.physrep.2008.08.003>
10. SimPy: Simulating Real-World Processes With Python. URL :
<https://realpython.com/simpy-simulating-with-python/> (дата звернення: 20.11.2024).