4. Lazzaro D., Montefusco L. B. Radial basis functions for the multivariate interpolation of large scattered data sets. *Journal of Computational and Applied Mathematics.* 2002. № 140. P. 521–536. URL: https://core.ac.uk/download/pdf/82502771.pdf (дата звернення: 16.05.2024).
5. Runge's phenomenon. *Wolfram Demonstrations Project.* URL: https://demonstrations.wolfram.com/RungesPhenomenon/ (дата звернення: 16.05.2024).

УДК 004.421:519.61](043.2)

Левченко М. Р., *здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,*
Сеник І. О., *асистент кафедри інформаційних технологій*

ПОРІВНЯЛЬНИЙ АНАЛІЗ МЕТОДІВ РОЗВ'ЯЗАННЯ СИСТЕМИ ЛІНІЙНИХ РІВНЯНЬ. МЕТОД ГАУСА ТА СІМПСОНА

Донецький національний університет імені Василя Стуса, м. Вінниця

У цьому дослідженні проводиться порівняльний аналіз ефективності і точності різних методів розв'язання систем лінійних рівнянь, зокрема методу Гауса та методу Сімпсона. Результати дослідження дають змогу виявити переваги та недоліки кожного методу залежно від характеристик системи лінійних рівнянь, як-от розмір матриці, співвідношення між числом рівнянь та невідомих, а також структура матриці. Ця робота має на меті визначити найбільш підходящий метод для конкретних умов задачі, що є важливим кроком у покращенні ефективності обчислення та точності результатів у чисельних обчисленнях.

Метод Гауса – його суть полягає в тому, що шляхом елементарних перетворень СЛАР приводять до еквівалентної системи трикутного або трапецієвидного вигляду, з якої послідовно, починаючи з останніх (за номером) змінних, знаходять усі невідомі [1].

Метод Сімпсона – це чисельний метод інтегрування, який використовується для наближеного обчислення визначених інтегралів функцій. Основна ідея полягає в апроксимації площі під кривою за допомогою квадратичних апроксимаційних функцій, відомих як параболи [2].

Порівняємо основні переваги та недоліки двох методів.

Таблиця 1 – Переваги методів

Метод Гауса	**Метод Сімпсона**
Універсальність: може застосовуватися для розв'язання широкого спектра систем лінійних рівнянь з різними характеристиками	*Висока точність:* часто дає дуже точні результати, особливо під час обчислення великих інтегралів
Стійкість: має добру стійкість до помилок округлення, що робить його надійним для використання в чисельних обчисленнях	*Простота реалізації:* має просту інтуїтивну інтерпретацію, що робить його доступним для реалізації навіть без великого досвіду в чисельних обчисленнях
Ефективність: для деяких типів систем лінійних рівнянь може бути дуже ефективним і швидким	*Можливість адаптації:* може бути легко адаптований для обчислення невизначених інтегралів та інших специфічних випадків

Таблиця 2 – Недоліки методів

Метод Гауса	**Метод Сімпсона**
Помірна складність: у деяких випадках може мати помірну обчислювальну складність, особливо для великих систем рівнянь	*Обмеження на класи функцій:* може бути менш ефективним для деяких класів функцій, зокрема для функцій з великими похідними або розривами
Чутливість до особливостей матриці: у разі певних особливостей матриці, як-от великі коефіцієнти або велика розбіжність між масштабами елементів, може втратити ефективність та стійкість	*Обчислювальна складність:* для деяких випадків може вимагати більше обчислювальних ресурсів, особливо для дуже складних функцій або великих інтервалів

Розглянемо практичну роботу методів у розв’занні системи лінійних рівнянь у мові Python.

```
import numpy as np
import sys

n = int(input('Введіть кількість невідомих: '))

a = np.zeros((n, n + 1))

x = np.zeros(n)

print('Введіть коефіцієнти розширеної матриці:')
for i in range(n):
    for j in range(n + 1):
        a[i][j] = float(input('a[' + str(i) + '][' + str(j) + ']='))

for i in range(n):
    if a[i][i] == 0.0:
        sys.exit('Ділення на нуль!')

    for j in range(i + 1, n):
        ratio = a[j][i] / a[i][i]

        for k in range(n + 1):
            a[j][k] = a[j][k] - ratio * a[i][k]

x[n - 1] = a[n - 1][n] / a[n - 1][n - 1]

for i in range(n - 2, -1, -1):
    x[i] = a[i][n]

    for j in range(i + 1, n):
        x[i] = x[i] - a[i][j] * x[j]

    x[i] = x[i] / a[i][i]

print('\nРозв’язок: ')
for i in range(n):
    print('X%d = %0.2f' % (i, x[i]), end='\t')
```

Рис. 1. Лістинг програми методу Гауса

```
Введіть кількість невідомих: 2
Введіть коефіцієнти розширеної матриці:
a[0][0]=1
a[0][1]=4
a[0][2]=5
a[1][0]=2
a[1][1]=8
a[1][2]=5
Ділення на нуль!
```

Рис. 2. Результат програми методу Гауса. Ділення на нуль

```
Введіть кількість невідомих: 2
Введіть коефіцієнти розширеної матриці:
a[0][0]=1
a[0][1]=2
a[0][2]=3
a[1][0]=5
a[1][1]=8
a[1][2]=1

Розв'язок:
X0 = -11.00 X1 = 7.00
```

Рис. 3. Результат програми методу Гауса

Програма реалізує метод Гаусса для розв'язання систем лінійних рівнянь. Користувач вводить кількість невідомих і коефіцієнти розширеної матриці, після чого код перетворює матрицю у верхньотрикутну форму через прямий хід, а потім виконує зворотній хід для обчислення розв'язків, які виводяться на екран.

```python
def f(x):
    return 1 / (1 + x ** 2)

def simpson13(x0, xn, n):
    h = (xn - x0) / n

    integration = f(x0) + f(xn)

    for i in range(1, n):
        k = x0 + i * h

        if i % 2 == 0:
            integration = integration + 2 * f(k)
        else:
            integration = integration + 4 * f(k)

    integration = integration * h / 3

    return integration


lower_limit = float(input("Введіть нижню межу інтеграції: "))
upper_limit = float(input("Введіть верхню межу інтеграції: "))
sub_interval = int(input("Введіть кількість підінтервалів: "))

result = simpson13(lower_limit, upper_limit, sub_interval)
print("Результат інтегрування за методом Сімпсона: %0.6f" % (result))
```

Рис. 4. Лістинг програми методу Сімпсона

```
Введіть нижню межу інтеграції: 0
Введіть верхню межу інтеграції: 1
Введіть кількість підінтервалів: 3
Результат інтегрування за методом Сімпсона: 0.720513
```

Рис. 5. Результат програми методу Сімпсона

Програма використовує метод Сімпсона для чисельного інтегрування, апроксимуючи функцію параболами на підінтервалах і сумуючи ці площі для обчислення інтегралу. Він визначає функцію $f(x)$, запитує у користувача межі інтеграції та кількість підінтервалів, обчислює значення функції у вузлових точках і використовує формулу методу Сімпсона для отримання результату.

Підсумовуючи це дослідження, зазначимо, що було проведено порівняльний аналіз методу Гаусса та методу Сімпсона, які застосовуються для розв'язання систем лінійних рівнянь та чисельного інтегрування відповідно. Метод Гауса показав високу універсальність та стійкість до помилок округлення, але може виявитися менш ефективним у разі великих систем або наявності особливостей у матриці. Метод Сімпсона натомість забезпечує високу точність інтегрування та простоту реалізації, але може бути обчислювально складним для складних функцій або великих інтервалів. Практичні приклади, реалізовані на Python, демонструють специфічні переваги та недоліки кожного методу, що підкреслює важливість вибору відповідного методу залежно від конкретних умов задачі.

Список використаних джерел

1. Верещак Р. Метод Гауса для розв'язання систем лінійних рівнянь: повний огляд. *Методи розв'язування систем лінійних алгебраїчних рівнянь*. 29.11.2023. URL: https://www.mathros.net.ua/metod-gaussa-rozvjazok-systemy-linijnyh-rivnjan-metodom-gaussa.html (дата звернення: 17.05.2024).
2. Метод Сімпсона: Основи та практичне застосування. *mathros.net.ua*. Mar. 3, 2024. URL: https://medium.com/@MathrosNetUa/метод-сімпсона-основи-та-практичне-застосування-e5803a0e9cc1 (дата звернення: 17.05.2024).
3. Gauss elimination method python program. *Codesansar*. URL: https://www.codesansar.com/numerical-methods/gauss-elimination-method-python-program.htm (дата звернення: 17.05.2024).
4. Simpson's Rule. *Mathematical Python.* URL: https://patrickwalls.github.io/mathematical-python/integration/simpsons-rule/ (дата звернення: 17.05.2024).

УДК 004.6

Кизь О. А., *здобувач 2 курсу*
спеціальності 122 Комп'ютерні науки,
науковий керівник:
Якубич К. О., *асистент кафедри*
інформаційних технологій

МЕТОД ГРАДІЄНТНОГО СПУСКУ

Донецький національний університет імені Василя Стуса, м. Вінниця

Методи оптимізації широко використовуються в обчислювальних задачах для пошуку найкращих рішень чи найоптимальніших параметрів. Вони допомагають знаходити мінімуми або максимуми функцій, розв'язувати задачі лінійного та нелінійного програмування, задачі розподілу ресурсів та багато інших. Для вирішення різних завдань у дослідженнях використовуються різні методи. Є алгоритми, які завершують свою роботу після певної кількості кроків, та інші, які пристосовуються до знаходження рішення на певних типах задач, і також є еврис-