МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ЗВІТ ПРО ВИКОНАННЯ

КОНТРОЛЬНОЇ РОБОТИ

із дисципліни: «Програмування мовою Python»

на тему «Робота з бібліотеками мови Python»

Виконав студент групи КНз-31с Коноз Р.В.

Варіант 3

Перевірила Парфененко Ю.В.

Суми 2024

Table of Contents

[ВСТУП 3](#_Toc186895129)

[1 Застосування бібліотек Python для розв’язання задач лінійної алгебри 4](#_Toc186895130)

[1.1 NumPy 4](#_Toc186895131)

[1.2 SciPy 4](#_Toc186895132)

[1.3 SymPy 5](#_Toc186895133)

[1.4 TensorFlow 5](#_Toc186895134)

[1.5 Порівняння 6](#_Toc186895135)

[2 Використання бібліотеки NumPy для розв’язання задач лінійної алгебри 7](#_Toc186895136)

[2.1 Постановка задачі 7](#_Toc186895137)

[2.2 Алгоритм розв’язання задачі 7](#_Toc186895138)

[2.3 Програмна реалізація 8](#_Toc186895139)

[2.4 Текст програми 9](#_Toc186895140)

[2.5 Контрольний приклад 10](#_Toc186895141)

[ВИСНОВКИ 11](#_Toc186895142)

[СПИСОК ЛІТЕРАТУРИ 13](#_Toc186895143)

# ВСТУП

У сучасному світі інформаційних технологій Python займає провідне місце серед мов програмування завдяки своїй простоті, гнучкості та багатофункціональності. Особливою перевагою Python є величезна екосистема бібліотек, яка дозволяє ефективно вирішувати задачі з різних галузей, включаючи математичні обчислення. Математичні операції, такі як лінійна алгебра, диференційні рівняння, оптимізація та обробка великих обсягів даних, є основою для багатьох наукових досліджень, інженерних задач і прикладних застосувань.

Бібліотеки Python, такі як **NumPy**, **SciPy**, **SymPy**, **Pandas**, **TensorFlow** та інші, забезпечують широкий набір інструментів для виконання як базових, так і складних математичних операцій. Їх використання дозволяє автоматизувати рутинні обчислення, підвищити точність і швидкість виконання задач, а також створювати високоефективні алгоритми для аналізу даних, моделювання та машинного навчання.

Значущість дослідження полягає в аналізі можливостей Python та його бібліотек для вирішення математичних задач, їхнього впливу на розвиток сучасної науки та технологій. У цьому контексті важливо вивчити функціонал ключових бібліотек, їх переваги та недоліки, а також можливості комбінування для розв’язання багатокомпонентних задач.

Метою цієї роботи є систематизація знань про Python-бібліотеки для математичних обчислень, аналіз їхнього функціоналу та практичного застосування, а також демонстрація ефективності їх використання у вирішенні задач лінійної алгебри, оптимізації та обробки великих даних.

# Застосування бібліотек Python для розв’язання задач лінійної алгебри

Python має великий набір бібліотек, які спрощують розв’язання задач лінійної алгебри. Ці бібліотеки забезпечують ефективну роботу з матрицями, векторами, обчисленням визначників, обернених матриць, вирішенням систем лінійних рівнянь тощо. У цьому аналізі ми розглянемо найбільш популярні бібліотеки: **NumPy**, **SciPy**, **SymPy**, і **TensorFlow**.

## NumPy

NumPy – це основна бібліотека Python для роботи з багатовимірними масивами, матрицями та виконання лінійної алгебри. В її можливості входять як підтримка базових операцій над векторами та матрицями (додавання, множення, тд.), так і функції для знаходження оберненої матриці, обчислення визначника, розв’язання систем лінійних рівнянь. Також одним з приємних особливостей цієї бібліотеки є її швидкість обробки за рахунок використання оптимізованих C-бібліотек.

Приклад:

Розв’язати систему лінійних рівнянь.

A computer screen shot of a code

Description automatically generated

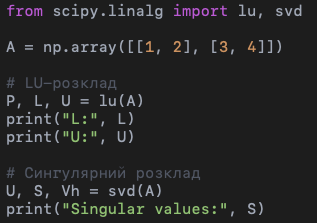
1. NumPy розв’язання системи лінійних рівнянь

## SciPy

SciPy в свою чергу пропонує набагато більш розширений та продвинутий функціонал того ж самого NumPy для більш складних математичних обчислень. Модуль scipy.linalg пропонує набагато розвиненіші функції для лінійної алгебри.

Приклад:

Розкласти матрицю сингулярно.



1. SciPy розкладання матриці сингулярно

## SymPy

SymPy є ідеальним вибором для задач, які потребують точного аналітичного розв’язку, таких як спрощення виразів, розв’язання рівнянь, диференціювання та інтегрування. Ця бібліотека корисна для навчання, досліджень і теоретичної математики, але для швидких числових обчислень краще використовувати NumPy або SciPy. SymPy чудово доповнює екосистему Python для вирішення математичних задач.

Приклад:

Інтегрувати вираз

A black screen with white text

Description automatically generated

1. SymPy інтегрування виразу

## TensorFlow

TensorFlow — це потужна бібліотека для числових обчислень і машинного навчання, яка оптимізована для роботи з тензорами (багатовимірними масивами). Окрім основної сфери застосування в нейронних мережах, TensorFlow ефективно використовується для вирішення задач лінійної алгебри, забезпечуючи високу продуктивність завдяки GPU/TPU-акселерації.

Приклад:

Обчислення визначника та оберненої матриці.

A computer screen shot of a black background

Description automatically generated

1. TensorFlow обчислення визначника та оберненої матриці

## Порівняння

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Бібліотека | Призначення | Основні функції | Швидкість | Простота використання | Символьні обчислення |
| NumPy | Загальні обчислення | Визначник, обернена матриця, розв’язання рівнянь | Висока | Легка | Ні |
| SciPy | Складні розклади матриць | SVD, LU,QR | Висока | Помірна | Ні |
| SymPy | Символьна математика | Точні аналітичні розв’язки | Низька | Помірна | Так |
| TensorFlow | Великі обчислення, ML | GPU/TPU оптимізація | Дуже висока | Складна | Ні |

# Використання бібліотеки NumPy для розв’язання задач лінійної алгебри

## Постановка задачі

Система лінійних рівнянь є однією з базових задач лінійної алгебри. Вона широко використовується в багатьох галузях, таких як фізика, інженерія, економіка, обчислювальна математика тощо. Формально система лінійних рівнянь виглядає так:

*Ах = b*

Нам необхідно розвязати систему лінійних рівнянь, тобто знайти такий вектор *х*, який задовільняє систему. Перш за все нам треба прийняти систему рівнянь у вигляді вхідних даних та конвертувати її в зрозумілі для програми матрицю *А* та вектор *b*. Далі перевіряємо чи є ця матриця квадратною та невиродженою. Після всіх цих маніпуляцій з перевірками ми можемо почати розв’язування системи рівнянь методом лінійної алгебри. Отримавши результат ми маємо його перевірити на точність шляхом підстановки *х* у початкову систему та перевірки рівності *Ах = b.*

Для прикладу виконання задачі розглянемо систему рівнянь:

У матричному вигляді це:

Наша ціль знайти вектор , який розв’язує цю систему.

## Алгоритм розв’язання задачі

Етап 1: Перевірка вхідних даних

1. Прийняти матрицю *А* і вектор *b* від користувача як вхідні дані.
2. Перевірка чи матриця є квадратною

Етап 2: Перевірка визначника

Обчислити det(A) за допомогою функції numpy.linalg.det

Якщо det(A) = 0, повідомити користувачу, що система не має єдиного розв’язку.

Етап 3: Розв’язання системи

Використати функцію numpy.linalg.solve, щоб знайти таке *х*, де *Ax = b*

Етап 4: Перевірка точності результату

Обчислити добуток *Ах* і порівняти з *b*

## Програмна реалізація

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Тип | Опис | Призначення |
| numpy | Модуль | Основна бібліотека для виконання математичних обчислень. | Робота з лінійними рівняннями |
| numpy.linalg.det | Метод | Обчислення визначника матриці. | Обчислити визначник матриці. |
| numpy.linalg.solve | Метод | Розв’язання системи рівнянь. | Розв’язати систему рівнянь |
| numpy.dot | Метод | Множення матриць | Перемножити матриці для перевірки точності розв’язку. |
| input | Вбудована функція | Введення даних користувачем. | Отримати матрицю та вектор від користувача |
| print | Вбудована функція | Виведення результатів. | Вивести результати |

Таблиця 2.1 - Таблиця компонентів програми

|  |  |  |
| --- | --- | --- |
| **Назва** | **Тип даних** | **Призначення** |
| A | numpy.ndarray | Матриця коефіцієнтів |
| b | numpy.ndarray | Вектор правих частин |
| det\_A | float | Визначник матриці A |
| x | numpy.ndarray | Розв’язок системи Ax = b |
| Ax | numpy.ndarray | Результат множення A\*x |
| is\_square | bool | Перевірка, чи матриця квадратна |

Таблиця 2.2 – Таблиця символьних імен

## Текст програми

import numpy as np

def main():

# Введення даних

print("Введіть матрицю A (у вигляді рядків через кому, наприклад: 1 2, 3 4):")

A = np.array([list(map(float, row.split())) for row in input().split(",")])

print("Введіть вектор b (у вигляді чисел через пробіл, наприклад: 5 6):")

b = np.array(list(map(float, input().split())))

# Перевірка, чи квадратна матриця

is\_square = A.shape[0] == A.shape[1]

if not is\_square:

print("Матриця A не є квадратною. Система рівнянь не може бути вирішена.")

exit()

# Обчислення визначника

det\_A = np.linalg.det(A)

if det\_A == 0:

print("Матриця вироджена, система не має єдиного розв’язку.")

else:

# Розв’язання системи рівнянь

x = np.linalg.solve(A, b)

print("Розв’язок системи рівнянь (x):")

print(x)

# Перевірка точності розв’язку

Ax = np.dot(A, x)

if np.allclose(Ax, b):

print("Розв’язок перевірено: Ax ≈ b")

else:

print("Розв’язок неточний.")

if \_\_name\_\_ == '\_\_main\_\_':

main()

## Контрольний приклад

**A screen shot of a computer

Description automatically generated**

1. Приклад роботи програми

# ВИСНОВКИ

У результаті виконання індивідуального завдання була створена програма для розв’язання квадратних систем лінійних рівнянь із використанням бібліотеки NumPy. Основною метою розробки було забезпечити інструмент, здатний точно та швидко обчислювати розв’язок систем виду *Ax = b*, де *А* — квадратна матриця коефіцієнтів, а *b* — вектор правих частин.

Програма реалізує такі ключові функції:

1. Введення даних користувачем у вигляді матриці *A* та вектора *b*.
2. Перевірка коректності введених даних, зокрема перевірка квадратності матриці.
3. Перевірка виродженості матриці шляхом обчислення її визначника *det(A)*.
4. Використання функції *numpy.linalg.solve* для точного знаходження вектора *x*, який задовольняє рівняння *Ax = b*.
5. Виведення розв’язку та повідомлення користувача про можливі помилки, такі як виродженість матриці.

Розроблена програма дозволяє швидко розв’язувати системи рівнянь і забезпечує високу точність результатів завдяки використанню спеціалізованих інструментів бібліотеки NumPy. Застосування цієї програми демонструє важливість лінійної алгебри в різних сферах, таких як фізика, інженерія, економіка та обчислювальна математика.

Програма є простим, але ефективним інструментом для роботи з квадратними системами лінійних рівнянь. Її гнучкість і інтуїтивно зрозумілий інтерфейс роблять її корисною для навчання основам лінійної алгебри. Завдяки автоматизованому обчисленню визначника і перевірці коректності вхідних даних програма здатна допомогти уникнути типових помилок і надати надійні результати навіть у разі складних обчислень.

Попри свої переваги, програма має певні обмеження. Вона призначена виключно для роботи з квадратними системами рівнянь і не враховує випадки переозначених або недоозначених систем. У майбутньому програма може бути розширена для роботи з такими системами шляхом реалізації методів найменших квадратів або інших підходів до розв’язання неквадратних задач. Також можливе впровадження візуалізації даних для наочного аналізу результатів.

Таким чином, розроблена програма є важливим кроком до автоматизації обчислювальних процесів у лінійній алгебрі. Вона є корисною як для студентів, які вивчають основи математики, так і для практичного застосування в різних галузях. Реалізація цієї програми підтверджує, що Python і його бібліотеки надають потужний інструментарій для розв’язання математичних задач і створення універсальних програм.

# СПИСОК ЛІТЕРАТУРИ

1. Python’s Requests Library [Електронний ресурс] – URL: <https://realpython.com/python-requests/>