Лабораторна робота № 11

Використання модулів. Опрацювання масивів та матриць засобами модуля NumPy.

Мета роботи: *ознайомлення з можливостями створення та використання масивів за допомогою функціоналу модуля NumPy та використання функцій цього модуля до прикладних задач лінійної алгебри.*

Короткі теоретичні відомості

NumPy — це бібліотека Python, яка використовується для роботи з масивами. Тут є також функціонал для застосуваня в сфері лінійної алгебри, перетворень Фур'є та роботи з матрицями. Проєкт з відкритим вихідним кодом NumPy був створений у 2005 році Тревісом Оліфантом.

У Python можна працювати з базовим функціоналом списків, які використовуються в якості масивів, але в такій ролі списки опрацьовуються доволі повільно, у порівнянні з масивами в інших тнхнологіях програмування. NumPy організовує роботу з об'єктами масивів, що працюють в десятки разів швидше за списки Python.

Об'єкт масиву ndarray надає багато допоміжних функцій, які роблять роботу з ним досить легкою. Масиви часто використовуються в науці про дані, де швидкість і ресурси дуже важливі. Масиви NumPy зберігаються в одному безперервному місці в пам'яті, на відміну від списків, що дозволяє процесам отримувати до них доступ і маніпулювати ними дуже ефективно. Окрім цього, NumPy оптимізовано для роботи з новітніми архітектурами ЦП.

NumPy — це бібліотека Python, яка частково написана на Python, але більшість частин, які потребують швидкого обчислення, написані на C або C++.

Для роботи з масивами використовується NumPy. Об'єкт масиву в NumPy називається ndarray. Можна створювати ndarray-об'єкт за допомогою функції array():

```
import numpy as np

arr = np.array([1, 2, 3, 4, 5])

print(arr)

print(type(arr))
```

Функції array можна передавати список, кортеж або будь-який об'єкт типу iterable, з якого можна утворити масив.

Масиви в NumPy можуть мати різну розмірність, зокрема, розглядаються:

0-D масиви, або скаляри, це елементи в масиві. Кожне значення в масиві є масивом 0-D.

1-D масиви, елементами яких є 0-D масиви.

2-D масиви, тобто, масиви, елементами якого є 1-D масиви.

3-D масиви - масиви, елементами яких є 2-D масиви.

і т.д.

За допомогою 2-D масивів можна опрацьовувати матриці. NumPy має цілий підмодуль, присвячений матричним операціям ***numpy.mat***. Окрім цього, NumPy має підмодуль ***numpy.linalg***, що містить функції для розв'язування задач лінійної алгебри, зокрема ***det()*** для обчислення визначників, та ***inv()*** для знаходження обереженої матриці.

Завдання

Використовуючи функціонал модуля numpy реалізувати програму для розв'язування задачі з лінйної алгебри, відповідно до варіанта.

1. Виконати процес розв'язування задачі при заданих в умові значеннях коефіцієнтів чи координат за допомогою командної стрічки Python;
2. Записати цей процес, для заданих в умові параметрів, у вигляді коду python-програми в PyCharm;
3. Вдосконалити створену програму, передбачивши ввід корисутвачем параметрів задачі та її розв'язування при довільних допустимих значеннях коефіцієнтів чи координат.

Варіанти завдань

1. Знайти матрицю X з рівняння

$$A \cdot X \cdot B = E, \text{ де } A = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 7 & -8 & 1 \end{pmatrix}, B = \begin{pmatrix} 2 & 3 & 4 \\ 5 & 4 & 3 \\ 4 & 3 & -2 \end{pmatrix},$$

а E – одинична матриця розміру 3x3.

2. Розв'язати систему лінійних алгебраїчних рівнянь

$$\begin{cases} 2x_1 + x_2 + 3x_3 + 4x_4 = 29, \\ 5x_1 + 2x_2 + 4x_3 + 3x_4 = 33, \\ -x_1 - 3x_2 + 2x_3 + 2x_4 = 7, \\ 4x_1 + 4x_2 + 3x_3 - 2x_4 = 13, \end{cases}$$

матричним методом. Перевірити правильність отриманого розв'язку, підставивши його компоненти в систему і знайшовши стовпець вільних членів.

3. Обчислити об'єм піраміди $ABCD$ з вершинами в точках $A(3,1,5)$, $B(-1,7,5)$, $C(3,-1,9)$ та $D(7,7,7)$. Визначити довжину висоти DN, проведеної з вершини D до грані ABC.

4. Перевірити, чи вектори $a_1(3,4,11)$, $a_2(-1,7,3)$, $a_3(3,-2,7)$ утворюють базу в тривимірному просторі. Якщо так, то розкласти за векторами цієї бази вектор $b(3,5,7)$.

5. Знайти матрицю X з рівняння

$$B \cdot X \cdot A = E, \text{ де } A = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 7 & -8 & 1 \end{pmatrix}, B = \begin{pmatrix} 2 & 3 & 4 \\ 5 & 4 & 3 \\ 4 & 3 & -2 \end{pmatrix},$$

а E – одинична матриця розміру 3x3.

6. Визначити відстань від точки $D(7,7,7)$ до площини, проведеної через точки $A(1,2,3)$, $B(3,-5,5)$, $C(7,6,5)$.

7. Знайти матрицю X з рівняння

$$A \cdot X = B, \text{ де } A = \begin{pmatrix} 4 & 3 & 2 & 1 \\ 1 & 2 & 3 & -3 \\ 3 & 1 & 2 & 1 \\ 7 & -8 & 1 & -3 \end{pmatrix}, B = \begin{pmatrix} 2 & 3 & 1 & 4 \\ -1 & 1 & 1 & -5 \\ 5 & 4 & 1 & 3 \\ 4 & 3 & 7 & -2 \end{pmatrix}.$$

8. Піраміду $ABCO$ задано координатами вершин в точках $A(3,4,5)$, $B(-1,7,5)$, $C(3,-2,11)$, точка O – початок координат. Визначити довжину висоти DN, проведеної з вершини D до грані ABC.

9. Знайти матрицю X з рівняння

$$A \cdot X + B = E, \text{ де } A = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 7 & -8 & 1 \end{pmatrix}, B = \begin{pmatrix} 2 & 3 & 4 \\ 5 & 4 & 3 \\ 4 & 3 & -2 \end{pmatrix},$$

а E – одинична матриця розміру 3x3.

10. Точки $A(3,1,5)$, $B(-1,7,5)$ та $C(3,-1,9)$ задано їхніми координатами в декартовій системі координат. Знайти об'єм паралелепіпеда, побудованого на векторах $\overrightarrow{OA}$, $\overrightarrow{OB}$ та $\overrightarrow{OC}$, де точка O – початок координат.

11. Знайти матрицю X з рівняння

$$A \cdot X = B \cdot C, \text{ де } A = \begin{pmatrix} 4 & 3 & 2 & 1 \\ 1 & 2 & 3 & -3 \\ 3 & 1 & 2 & 1 \\ 7 & -8 & 1 & -3 \end{pmatrix},$$

$$B = \begin{pmatrix} 2 & 3 & 1 \\ -1 & 2 & 3 \\ 5 & 4 & 1 \\ 4 & 3 & 7 \end{pmatrix}, C = \begin{pmatrix} 2 & 3 & 3 & 1 \\ 5 & 4 & -2 & 1 \\ 4 & 3 & 1 & 7 \end{pmatrix}.$$

12. Обчислити об'єм та площу повної поверхні піраміди $ABCD$ з вершинами в точках $A(3,1,5)$, $B(-1,7,5)$, $C(3,-1,9)$ та $D(7,7,7)$.

13. Знайти матрицю X з рівняння

$X \cdot B = A$, де $A = \begin{pmatrix} 1 & 4 & 2 & 3 \\ 4 & 7 & 1 & 1 \\ 3 & 2 & 1 & 2 \\ 7 & 2 & -8 & 1 \end{pmatrix}$, $B = \begin{pmatrix} 2 & 1 & 3 & 4 \\ 5 & 2 & 4 & 3 \\ -1 & -3 & 2 & 2 \\ 4 & 4 & 3 & -2 \end{pmatrix}$,

а E – одинична матриця розміру 3x3.

14. Перевірити, чи сумісна система лінійних алгебраїчних рівнянь

$$\begin{cases} 2x_1 + x_2 + 3x_3 + 4x_4 = 29, \\ 5x_1 + 2x_2 + 4x_3 + 3x_4 = 33, \\ -x_1 - 3x_2 + 2x_3 + 2x_4 = 7, \\ 4x_1 + 4x_2 + 3x_3 - 2x_4 = 13. \end{cases}$$

Якщо система сумісна і має єдиний розв'язок, знайти цей розвязок, використовуючи матричний метод.

15. Знайти матрицю X з рівняння

$X \cdot B - A = E$, де $A = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \\ 7 & -8 & 1 \end{pmatrix}$, $B = \begin{pmatrix} 2 & 3 & 4 \\ 5 & 4 & 3 \\ 4 & 3 & -2 \end{pmatrix}$,

а E – одинична матриця розміру 3x3.

16. Розв'язати систему лінійних алгебраїчних рівнянь

$$\begin{cases} x_1 + 2x_2 + 3x_3 = 1, \\ -5x_1 + 3x_2 + 5x_3 = 2, \\ 2x_1 + 3x_2 + 4x_3 = 4, \end{cases}$$

методом Крамера та за допомогою оберненої матриці і порівняти результати.

Приклад виконання завдання

Варіант № 16. Умова задачі: Використовуючи функціонал модуля ***numpy*** реалізувати програму для розв'язування задачі з лінйної алгебри, відповідно до варіанта. Розв'язати систему лінійних алгебраїчних рівнянь

$$\begin{cases} x_1 + 2x_2 + 3x_3 = 1, \\ -5x_1 + 3x_2 + 5x_3 = 2, \\ 2x_1 + 3x_2 + 4x_3 = 4, \end{cases}$$

методом Крамера та за допомогою оберненої матриці і порівняти результати.

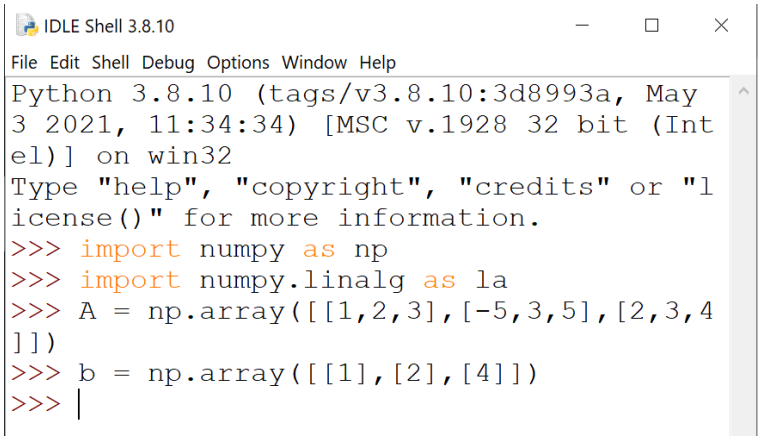
Для обчислень потрібно задати матрицю системи $A = \begin{pmatrix} 1 & 2 & 3 \\ -5 & 3 & 5 \\ 2 & 3 & 4 \end{pmatrix}$, стов-

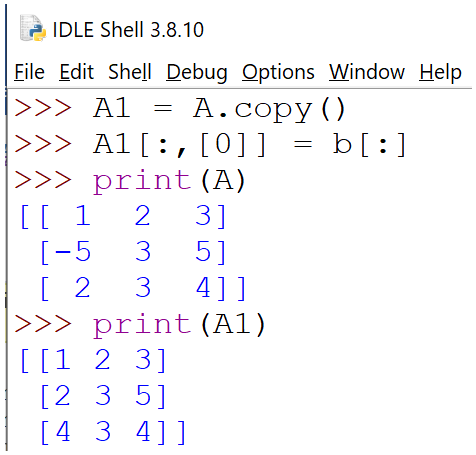
пець вільних членів $b = \begin{pmatrix} 1 \\ 2 \\ 4 \end{pmatrix}$. Далі слід обчислити детермінант матриці A та

порахувати визначники $\Delta_1 = \begin{vmatrix} 1 & 2 & 3 \\ 2 & 3 & 5 \\ 4 & 3 & 4 \end{vmatrix}$, $\Delta_2 = \begin{vmatrix} 1 & 1 & 3 \\ -5 & 2 & 5 \\ 2 & 4 & 4 \end{vmatrix}$, $\Delta_1 = \begin{vmatrix} 1 & 2 & 1 \\ -5 & 3 & 2 \\ 2 & 3 & 4 \end{vmatrix}$

Виконуємо процес розв'язування задачі для заданих в умові значень коефіцієнтів за допомогою командної стрічки Python

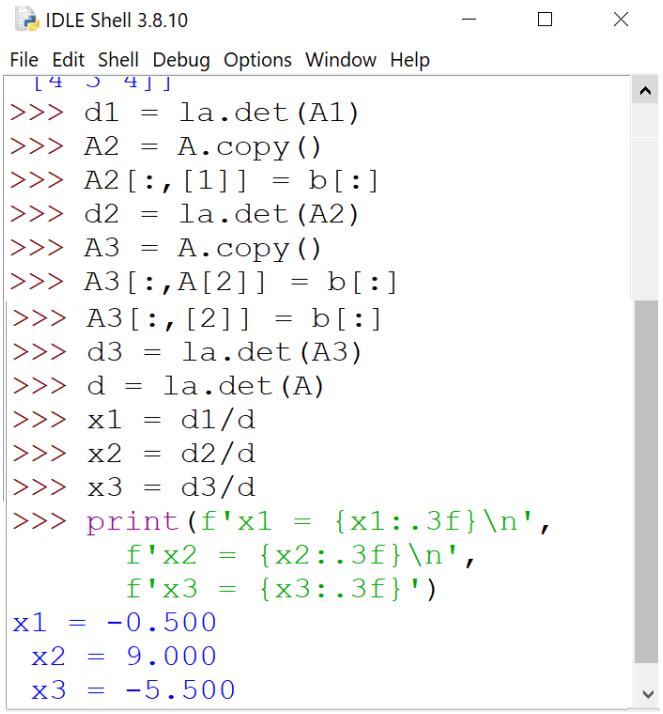
Запускаємо інтрепретатор Python в режимі командної стрічки (можна також скористатися головним вікном IDLE Python). Спочатку підключаємо модулі ***numpy*** та numpy.linalg. Далі вводимо матрицю коефіцієнтів та ствпець вільних членів.



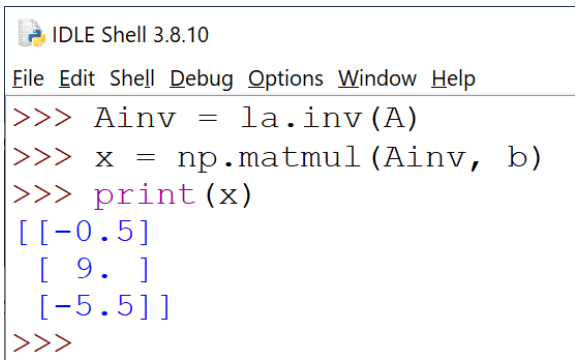


Далі створюємо копії матриці A та, заповняючи відповідні стовпці матриці значеннями стовпця b, знаходимо визначники Δ_1, Δ_2, Δ_3.

Після цього, знаходимо визначник матриці A, та ділимо на нього Δ_1, Δ_2, Δ_3, щоб отримати компоненти розв'язку системи.



Для отримання розв'язку системи іншим способом, знаходимо обернену до A матрицю, та множимо її на стовпець вільних членів зліва. Знову виводимо отриманий результат.



Як бачимо, розв'язки співпадають.

Записуємо використані команди у python-файл за допомогою PyCharm.

Отримаємо програму:

```
# модуль для роботи з масивами
import numpy as np
# модуль з функціями лінійної алгебри
import numpy.linalg as la
# матриця A з коефіцієнтів системи рівнянь фіксована
A = np.array([[1,2,3],[-5,3,5],[2,3,4]])
# стовпець вільних членів b фіксований
b = np.array([[1],[2],[4]])
# метод Крамера
A1 = A.copy()  # копія матриці A для знаходження
A1[:, [0]] = b[:]  # визначника "Дельта 1"
d1 = la.det(A1)
A2 = A.copy()  # копія матриці A для знаходження
A2[:, [1]] = b[:]  # визначника "Дельта 2"
d2 = la.det(A2)
A3 = A.copy()  # копія матриці A для знаходження
A3[:, [2]] = b[:]  # визначника "Дельта 3"
d3 = la.det(A3)
d = la.det(A)
x1 = d1 / d
x2 = d2 / d
x3 = d3 / d
print('''Розв\'язок системи, отриманий
                 методом Крамера: ''')
print('x1 = %10.3f' % x1)
print('x2 = %10.3f' % x2)
print('x3 = %10.3f' % x3)
# Матричний метод
Ainv = la.inv(A)
```

```
x = np.matmul(Ainv, b)
print('''Розв\'язок системи, отриманий
           за допомогою оберненої матриці: ''')
print('x1 = %10.3f' % x[0])
print('x2 = %10.3f' % x[1])
print('x3 = %10.3f' % x[2])
```

В отриманому коді заміняємо фіксовані матрицю та стовпець вільних членів, надавши можливість користувачеві вводити ці дані з клавіатури:

```
import numpy as np
# модуль з функціями лінійної алгебри
import numpy.linalg as la
# задаємо порожню матрицю та заповняємо її з клавіатури
A = np.empty([3, 3], dtype=float)
for i in range(3):
    for j in range(3):
        A[i, j] = float(input('A'+str(i+1)+str(j+1)+' = '))
# стовпець вільних членів вводимо з клавіатури
b = np.empty([3,1],dtype=float)
for i in range(3):
    b[i] = float(input('b' + str(i + 1) + ' = '))
# метод Крамера
A1 = A.copy()  # копія матриці A для знаходження
A1[:, [0]] = b[:]  # визначника "Дельта 1"
d1 = la.det(A1)
A2 = A.copy()  # копія матриці A для знаходження
A2[:, [1]] = b[:]  # визначника "Дельта 2"
d2 = la.det(A2)
A3 = A.copy()  # копія матриці A для знаходження
A3[:, [2]] = b[:]  # визначника "Дельта 3"
d3 = la.det(A3)
d = la.det(A)
```

```
x1 = d1 / d
x2 = d2 / d
x3 = d3 / d
print('''Розв\'язок системи, отриманий
                 методом Крамера: ''')
print('x1 = %10.3f' % x1)
print('x2 = %10.3f' % x2)
print('x3 = %10.3f' % x3)
# Матричний метод
Ainv = la.inv(A)
x = np.matmul(Ainv, b)
print('''Розв\'язок системи, отриманий
           за допомогою оберненої матриці: ''')
print('x1 = %10.3f' % x[0])
print('x2 = %10.3f' % x[1])
print('x3 = %10.3f' % x[2])
```

Запитання для самоконтролю

1. Для розв’язування якич задач створено модуль ***numpy***?
2. Який функціонал зібрано в модулі ***numpy.linalg*** бібліотеки ***NumPy***?
3. Як обчислити детермінант матриці?
4. Як засобами бібліотеки ***NumPy*** знайти обернену матрицю?
5. Який тип даних використовує бібліотека ***NumPy*** для опрацювання масивів?