4. Лабораторна робота. Застосування бібліотеки NumPy до задач лінійної алгебри

Мета: засвоїти можливості роботи бібліотеки NumPy до математичних обчислень, таких як операції з багатовимірними масивами, операції з матрицями, розв'язання систем лінійних рівнянь, використання вбудованих математичних функцій.

Теоретичні відомості та методичні рекомендації

Бібліотека NumPy (Numerical Python) є основним інструментом для наукових обчислень у Python. Вона забезпечує ефективні методи роботи з багатовимірними масивами та містить велику кількість функцій для виконання математичних операцій. Бібліотека NumPy значно спрощує математичні обчислення, роблячи їх ефективнішими, швидшими та зручнішими для реалізації. Вона незамінна у чисельному аналізі, машинному навчанні, моделюванні, інженерії та багатьох інших галузях.

NumPy має велику кількість підмодулів. Зазвичай для звичайного використання NumPy потрібен лише основний простір імен і менший набір підмодулів. Решта – це підмодулі спеціального призначення.

Рекомендовані простори імен для загального використання:

- numpy;
- numpy.exceptions;
- numpy.fft;
- numpy.linalg;
- numpy.polynomial;
- numpy.random;
- numpy.strings;
- numpy.testing;
- numpy.typing.

Простори імен спеціального призначення:

- numpy.ctypeslib;
- numpy.dtypes;
- numpy.emath;
- numpy.lib;
- numpy.rec;
- numpy.version.

Застарілі простори імен, які бажано не використовувати для нового коду:

- numpy.char;
- numpy.distutils;
- numpy.f2py;
- numpy.ma;
- numpy.matlib.

В NumPy основним об’єктом для зберігання та обробки числових даних є багатовимірний масив (`numpy.ndarray`). Найчастіше це одномірна послідовність або двовимірна таблиця, заповнені елементами одного типу, як правило, числами, які проіндексовані кортежом додатних цілих чисел. У NumPy елементи цього кортежу називаються осями (див. рис. 4.1), а число осей рангом. Цей об’єкт значно ефективніший за стандартні списки Python, оскільки зберігає всі елементи одного типу та підтримує векторизовані операції.

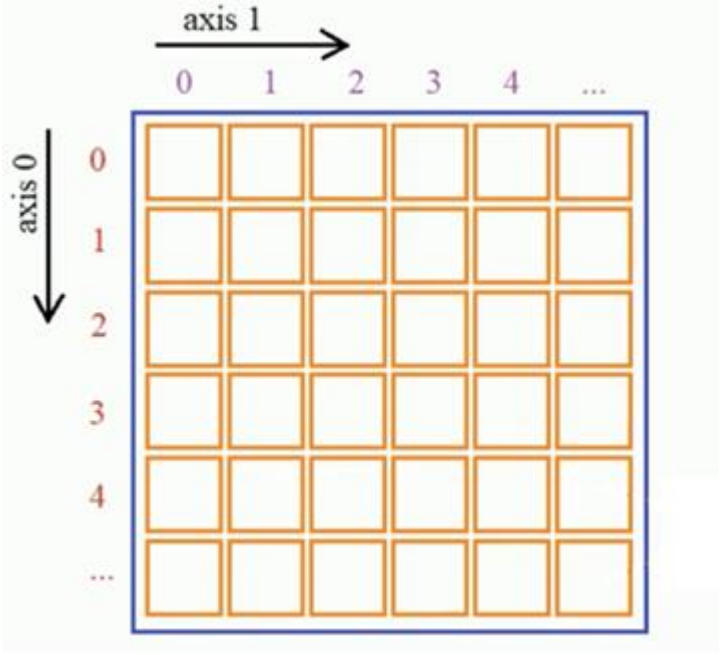

Рисунок 4.1 – Напрямок осей у двовимірному масиві у NumPy

Масив `numpy.ndarray` можна створити за допомогою функції `np.array(object, dtype)`, де `object` – список або інша структура даних, що конвертується в `ndarray`, `dtype` – (необов’язково) тип даних масиву (`int`, `float`, `complex`, `bool`, тощо). Вона дозволяє конвертувати звичайні списки Python у масиви NumPy, які більш ефективні для числових обчислень.

Наприклад,

```
import numpy as np

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

arr_float = np.array([1, 2, 3], dtype=float)
print(arr_float)  # [1. 2. 3.]

arr_bool = np.array([0, 1, 2], dtype=bool)
print(arr_bool)  # [False  True  True]

arr_2d = np.array([[1, 2, 3], [4, 5, 6]])
```

```
print(arr_2d)  # [[1 2 3]  #  [4 5 6]]
```

Кожен `ndarray` має корисні методи:

- `shape` – розмірність `ndarray`;
- `size` – загальна кількість елементів `ndarray`;
- `ndim` – кількість вимірів `ndarray`;
- `dtype` – тип даних `ndarray`;
- `itemsize` – розмір одного елемента (в байтах) `ndarray`.

Наприклад,

```
import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]])

print(arr.shape)  # (2, 3) - розмірність (рядки, стовпці)
print(arr.size)   # 6 - загальна кількість елементів
print(arr.ndim)   # 2 - кількість вимірів
print(arr.dtype)  # int32 або int64 - тип даних
print(arr.itemsize)  # розмір одного елемента (в байтах)
```

NumPy підтримує матричні операції. Нехай треба задати матрицю 3×2. Треба вказати зовнішній список у квадратних дужках `[]`. А в середині вкладені списки. У результаті ми отримаємо *двовимірний масив* розмінністю 3 рядка і 2 стовпчика:

```
import numpy as np

a = np.array([[1, 2], [3, 4], [5, 6]])
print(a)
```

В результаті отримаємо матрицю 3×2:

```
[[1 2]
 [3 4]
 [5 6]]
```

NumPy містить кілька вбудованих функцій для генерації матриць різних типів:

- `np.zeros(shape, dtype=float)` – створює матрицю нулів заданої форми;
- `np.ones(shape, dtype=float)` – створює матрицю одиниць заданої форми;
- `np.full(shape, fill_value)` – створює матрицю, заповнену певним числом `fill_value`;
- `np.eye(N, M=None, k=0)` – створює одиничну матрицю (або діагональну, якщо `M` не `None`, `k` – номер діагоналі);
- `np.random.rand(N, M)` – створює матрицю випадкових чисел з діапазону `[0, 1)`;

– `np.arange(start, stop, step)` – створює масив чисел у заданому діапазоні;
– `np.linspace(start, stop, num)` – створює масив рівномірно розподілених чисел.

Ці функції корисні для лінійної алгебри, моделювання, машинного навчання та інших задач.

NumPy містить кілька функцій для формування масивів на основі існуючих даних. Вони дозволяють змінювати структуру, розмірність і вміст масивів без необхідності створювати їх з нуля.

Функції копіювання та конвертації:

– `np.array(object, copy=True)` – створює новий масив з існуючих даних;
– `np.copy(arr)` – створює повну копію масиву.

Зміна форми масиву:

– `np.reshape(arr, new_shape)` – змінює форму масиву без зміни даних;
– `arr.flatten()` та `arr.ravel()` – перетворюють багатовимірний масив в одновимірний.

Додавання нових осей:

– `np.expand_dims(arr, axis)` – додає новий вимір;
– `arr[:, np.newaxis]` – альтернативний спосіб.

Об'єднання масивів:

– `np.concatenate((arr1, arr2), axis)` – об'єднує масиви вздовж заданої осі;
– `np.hstack()` та `np.vstack()` – горизонтальне з'єднання та вертикальне з'єднання.

Розбиття масивів:

– `np.split(arr, sections, axis)` – розбиває масиву на рівні частини;
– `np.hsplit()` та `np.vsplit()` – спрощені варіанти попередніх команд.

Повторення елементів:

– `np.tile(arr, reps)` – повторює масив певну кількість разів;
– `np.repeat(arr, repeats, axis)` – повторює кожен елемент певну кількість разів.

NumPy надає широкий набір матричних операцій, які дозволяють виконувати базові та розширені лінійні алгебраїчні обчислення (використовуючи підмодуль `linalg`):

– + і – – додавання та віднімання;
– – множення елементів (елементне множення);
– `@` або `np.dot()` – матричне множення;

– `T` – Транспонування;
– `inv` – обернена матриця;
– `det` – визначник;
– `trace` – слід матриці;
– `matrix_rank` – ранг матриці;
– `eig` – власні значення;
– `svd` – сингулярний розклад;

Наприклад, знайдемо транспоновану і обернену матрицю:

```
import numpy as np

A = np.array([(1, -2, 3), (4, 5, -6), (-7, 8, 9)],
'float64')
print(A)

A_T = A.T
print(A_T)

A_inv = np.linalg.inv(A)
print(A_inv)
```

У результаті отримаємо:

```
[[ 1. -2.  3.]
 [ 4.  5. -6.]
 [-7.  8.  9.]]
[[ 1.  4. -7.]
 [-2.  5.  8.]
 [ 3. -6.  9.]]
[[ 0.32978723  0.14893617 -0.0106383 ]
 [ 0.0212766   0.10638298  0.06382979]
 [ 0.23758865  0.0212766   0.04609929]]
```

Системи $Ax = b$ алгебраїчних рівнянь можна розв'язувати, використовуючи метод `solve` з підмодуля `linalg`.

Завдання до лабораторної роботи

1. Створити масиви

$$A = \begin{pmatrix} -8 & 10-n & h-2 \\ 5 & 2h+n & n+8 \\ -2 & 3h-n & -1 \end{pmatrix},\ B = \begin{pmatrix} -7 & -n & -h+1 \\ 4 & h & 6-n \\ 3 & n & 7 \end{pmatrix},\ C = \begin{pmatrix} n \\ h \\ n+h \end{pmatrix},$$

де n – номер варіанту, h – остання цифра номеру вашої групи.

1) Перевірити, однакові чи неоднакові масиви A та B.
2) Знайти добуток матриць $A \cdot B$.
3) Замінити знак елементів у масиві A, значення яких знаходяться між 1 та 10.

4) Замінити знак елементів у масиві B, значення яких знаходяться між 5 та 8.
5) Знайти мінімальний та максимальний елемент масивів A та B.
6) Перетворити масив A із float в int.
7) Знайти діагональні елементи добутку матриць $A \cdot B$.
8) Поміняти місцями 2 рядки у матриці B.
9) Обчислити ранги матриць A та B.
10) Знайти 3 найбільших значень у масиві A. Знайти 2 найбільших значень у масиві B.
11) Розв'язати систему лінійних рівнянь: $Ax = C$ двома способами.

2. Створити одновимірний масив даних, в якому буде вказана зарплата вчителя по місяцям за поточний рік (брати у діапазоні від 7500 гривень до 15500 гривень). Нехай податок на доходи фізичних осіб обчислюється так. Базова ставка становить 12%. Якщо в якомусь місяці заробіток вчителя за рік складе більше 50000 гривень, то на частину року яка залишилася (не включаючи цей місяць) встановлюється ставка в 20%. Наприклад, якщо вчитель заробляє щомісяця 8000 гривень, то до липня вчитель отримає $8000 \cdot 7 = 56000$ гривень і починаючи з серпня податок на доходи фізичних осіб нараховуватиметься за ставкою 20%. Написати функцію `calculate()`, яка приймає на вхід масив, що містить дохід за кожен місяць року, починаючи з першого та повертає загальну суму податку, який належить заплатити за рік.

Питання для самоконтролю

1. Охарактеризуйте можливості бібліотеки NumPy мови Python.
2. Що таке numpy.ndarray?
3. Наведіть корисні корисні методи `ndarray`.
4. Розкажіть про функції для формування масивів на основі існуючих даних у NumPy.
5. Чи можна виконувати матричні дії NumPy?
6. Охарактеризуйте особливості роботи з матрицями у NumPy.
7. Як виконується елементне множення у NumPy?
8. Якими способами можна розв'язати системи алгебраїчних рівнянь у NumPy?