Matrizes e Álgebra linear em Python com numpy

Carlos Tavares

28 de Outubro de 2024

Python possui ferramentas poderosas para manipulação de matrizes e operações de álgebra linear.

Numpy é uma biblioteca popular que fornece suporte para arrays multidimensionais, operações matemáticas eficientes e uma ampla gama de funções de álgebra linear.

Definição de matrizes em Python

Definição de matrizes em Python

Para criar matrizes com Numpy, usamos a função array. Aqui está um exemplo de como definir uma matriz 2×2:

$$A = \text{np.array}([[1, 2], [3, 4]])$$

Podemos também criar matrizes especiais, como matrizes identidade ou matrizes de zeros:

$$I = \text{np.eye}(3) \quad \text{(matriz identidade } 3 \times 3\text{)}$$

$$Z = \text{np.zeros}((2, 3)) \quad \text{(matriz de zeros } 2 \times 3\text{)}$$

Operações Básicas com Matrizes i

Numpy permite realizar operações básicas como soma, subtração e multiplicação por escalar diretamente em matrizes:

$$B = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$C = A + B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix}$$

Para a multiplicação por escalar:

$$D = 2 \cdot A = 2 \cdot \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 2 & 4 \\ 6 & 8 \end{bmatrix}$$

Operações Básicas com Matrizes ii

Para multiplicação de matrizes, usamos np.dot(A, B):

$$E = A \cdot B = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix}$$

Determinantes e Matrizes Inversas i

Numpy também pode calcular o determinante e a inversa de uma matriz:

$$\text{Determinante de } A: \quad \text{np.linalg.det}(A) = -2.0$$

$$\text{Inversa de } A: \quad \text{np.linalg.inv}(A) = \begin{bmatrix} -2 & 1 \\ 1.5 & -0.5 \end{bmatrix}$$

A matriz inversa é definida apenas para matrizes quadradas cujo determinante não é zero.

Autovalores e Autovetores i

Outra operação importante em álgebra linear é encontrar os valores próprios e vectores próprios de uma matriz:

$$\lambda, v = \text{np.linalg.eig}(A)$$

onde λ representa os valores próprios e v os vectores próprios da matriz A.

Os autovalores indicam as direções de maior variação, enquanto os autovetores fornecem as direções correspondentes.

Sistemas Lineares com Numpy

Para resolver sistemas de equações lineares da forma $Ax = B$ em Python, podemos usar a função numpy.linalg.solve da biblioteca Numpy. Neste sistema:

- A: matriz de coeficientes do sistema.
- x: vetor de incógnitas.
- B: vetor de constantes.

A função numpy.linalg.solve é eficiente e adequada para resolver sistemas de equações lineares, aplicando métodos otimizados de álgebra linear.

Exemplo: i

Considere o sistema de equações:

$$\begin{cases} 3x + 2y = 5 \\ x - y = 1 \end{cases}$$

Podemos escrever na forma matricial $Ax = B$ como:

$$A = \begin{bmatrix} 3 & 2 \\ 1 & -1 \end{bmatrix}, \quad B = \begin{bmatrix} 5 \\ 1 \end{bmatrix}$$

Exemplo: ii

Com o seguinte código Python, pode-se resolver o sistema de equações:

```
import numpy as np
A = np.array ([[3, 2], [1, -1]])
B = np.array ([5, 1])
x = np.linalg.solve (A, B)
print("Soluções:", x)
```

A saída será: $x = \begin{bmatrix} 1.4 \\ -0.4 \end{bmatrix}$, indicando as soluções para x e y.

Exercícios Práticos i

Exercício 1: Crie uma matriz 3×3 com valores à sua escolha. Encontre sua matriz inversa e verifique se a multiplicação da matriz pela sua inversa resulta na matriz identidade.

Exercício 2: Calcule os valores próprios e vectores próprios da matriz $A = \begin{bmatrix} 4 & 2 \\ 1 & 3 \end{bmatrix}$. Interprete o resultado.

Perguntas?