Python. Macierze

B.Kowal

10 listopada 2021

Numpy Array

```python
import numpy as np

A = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]], dtype = np.float64)

print("macierz A:\n", A)
print("liczba wierszy len(A): ",len(A))
print("liczba kolumn len(A[0]): ",len(A[0]))
print("liczba kolumn i wierszy np.shape(A): ",np.shape(A))
print("drugi wiersz, trzecia kolumna A[1,2]: ", A[1,2])
```

```
macierz A:
 [[ 1.  2.  3.  4.]
 [ 5.  6.  7.  8.]
 [ 9. 10. 11. 12.]]
liczba wierszy len(A):  3
liczba kolumn len(A[0]):  4
liczba kolumn i wierszy np.shape(A):  (3, 4)
drugi wiersz, trzecia kolumna A[1,2]:  7.0
```

Numpy Matrix

```python
import numpy as np

A = np.matrix([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]], dtype = np.float64)
B = np.matrix('0, 2, 4, 6; 8, 10, 12, 14; 16, 18, 20, 22', dtype = np.float64)

print("macierz A:\n", A)
print("macierz B:\n", B)
```

```
macierz A:
 [[ 1.  2.  3.  4.]
 [ 5.  6.  7.  8.]
 [ 9. 10. 11. 12.]]
macierz B:
 [[ 0.  2.  4.  6.]
 [ 8. 10. 12. 14.]
 [16. 18. 20. 22.]]
```

Wypisywanie kolumn, wierszy, wycinków macierzy

```python
import numpy as np

A = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]], dtype = np.float64)

print("drugi wiersz A[1]: ", A[1])
print("trzecia kolumna A[:,2]: ", A[:,2])
print("pierwszy wiersz, ostatni element A[0,-1]: ", A[0,-1])
print("wycinek macierzy,\n wiersze drugi-trzeci, kolumny trzecia-czwarta A[1:3,2:4]: \n", A[1:3,2:4])
```

```
drugi wiersz A[1]:  [5. 6. 7. 8.]
trzecia kolumna A[:,2]:  [ 3.  7. 11.]
pierwszy wiersz, ostatni element A[0,-1]:  4.0
wycinek macierzy,
 wiersze drugi-trzeci, kolumny trzecia-czwarta A[1:3,2:4]:
 [[ 7.  8.]
 [11. 12.]]
```

Dodawanie kolumn i wierszy

```python
print("Dodanie wiersza:")
A2=np.append(A, [[7,7,7,7]], axis=0)
print(A2,'\n')

print("Dodanie kolumny:")
A3=np.append(A, np.transpose([[3,3,3]]), axis=1)
print(A3,'\n')
```

```
Dodanie wiersza:
[[ 1.  2.  3.  4.]
 [ 5.  6.  7.  8.]
 [ 9. 10. 11. 12.]
 [ 7.  7.  7.  7.]]


Dodanie kolumny:
[[ 1.  2.  3.  4.  3.]
 [ 5.  6.  7.  8.  3.]
 [ 9. 10. 11. 12.  3.]]
```

Definiowanie macierzy

```
1  print("Macierz wypelniona jedna liczba:")
2  A = np.full((2,3), 7)
3  print(A,'\n')
4  print("Macierz jednostkowa:")
5  B = np.eye(3)
6  print(B,'\n')
7  print("Macierz poteg:")
8  C = np.array([[(x+1)**p for p in range(4)] for x in range(4)])
9  print(C,'\n')
```

```
Macierz wypełniona jedną liczbą:
[[7 7 7]
 [7 7 7]]

Macierz jednostkowa:
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]

Macierz potęg:
[[ 1  1  1  1]
 [ 1  2  4  8]
 [ 1  3  9 27]
 [ 1  4 16 64]]
```

Zamiana miejscami kolumn i wierszy

```
import numpy as np
A = np.array([[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]], dtype =
↪  np.float64)
print("macierz A:\n", A)
#zamiana miejscami wierszy 1. z 3.
A[[0, 2]] = A[[2, 0]]
print("macierz A po zamianie miejscami wierszy 1. z 3.:\n", A)
#zamiana miejscami kolumn 1. z 3.
A[:, [0, 2]] = A[:, [2, 0]]
print("macierz A po zamianie miejscami kolumn 1. z 3.:\n", A)
```

```
macierz A:
 [[ 1.  2.  3.  4.]
 [ 5.  6.  7.  8.]
 [ 9. 10. 11. 12.]]
macierz A po zamianie miejscami wierszy 1. z 3.:
 [[ 9. 10. 11. 12.]
 [ 5.  6.  7.  8.]
 [ 1.  2.  3.  4.]]
macierz A po zamianie miejscami kolumn 1. z 3.
 [[11. 10.  9. 12.]
 [ 7.  6.  5.  8.]
 [ 3.  2.  1.  4.]]
```

Wyznacznik, odwrotność, ślad, rząd, suma

```python
import numpy as np
B = np.array([[1, 1, 3], [4, 1, 8], [6, 1, 1]], dtype = np.float64)
print("wyznacznik det(B): ", np.linalg.det(B))
print("odwrotnosc inv(B): \n", np.linalg.inv(B))
print("slad: ",np.trace(B))
print("transpozycja: \n", np.transpose(B))
print("rzad: ", np.linalg.matrix_rank(B))
print("suma elementow: ", np.sum(B))
```

```
wyznacznik det(B):  31.0
odwrotność inv(B):
 [[-0.22580645  0.06451613  0.16129032]
 [ 1.41935484 -0.5483871   0.12903226]
 [-0.06451613  0.16129032 -0.09677419]]
ślad:  3.0
transpozycja:
 [[1. 4. 6.]
 [1. 1. 1.]
 [3. 8. 1.]]
rząd:   3
suma elementów:  26.0
```

Działania na macierzach

```
import numpy as np
A = np.array([[1,2],[3,4]], dtype=np.float64)
B = np.array([[1,2],[2,1]], dtype=np.float64)
print("A: ", '\n', A, '\n\n', "B: ",'\n', B,'\n')

print("Dodawanie i odejmowanie macierzy:")
print("A + B: ", '\n', A + B, '\n\n', "A - B: ", A - B,'\n')
print("Mnozenie i dzielenie elementow macierzowych:")
print("A * B: ", '\n', A * B, '\n\n', "A / B: ", A / B,'\n')
print("Mnozenie elementow macierzowych przez liczbe:")
print("2 * A: ", '\n', 2*A ,'\n')
print("Mnozenie macierzy:")
print("A.B: ", '\n', np.dot(A, B),'\n')
```

Działania na macierzach

```
A:
 [[1. 2.]
 [3. 4.]]

 B:
 [[1. 2.]
 [2. 1.]]

Dodawanie i odejmowanie macierzy:
A + B:
 [[2. 4.]
 [5. 5.]]

 A - B:  [[0. 0.]
 [1. 3.]]

Mnozenie i dzielenie elementow macierzowych:
A * B:
 [[1. 4.]
 [6. 4.]]

 A / B:  [[1.  1. ]
 [1.5 4. ]]

Mnozenie elementow macierzowych przez liczbe:
2 * A:
 [[2. 4.]
 [6. 8.]]

Mnozenie macierzy:
A.B:
 [[ 5.  4.]
 [11. 10.]]
```

Przeszukiwanie macierzy

```
import numpy as np
A = np.array([[0, 3],  [1, 6],  [7, 2]], dtype = float)
print("macierz A:\n", A)
print("Wartość logiczna A!=0",'\n',A!=0)
print("Indeksy elementow nierównych 0")
print(np.argwhere(A!=0))

print("Indeksy elementow mniejszych od 4 i nierownych 0")
print(np.argwhere((A<4) & (A!=0)))

print("Indeksy elementow w pierwszej kolumnie nierownych 0")
print(np.argwhere(A[:,0]!=0))
```

Przeszukiwanie macierzy

```
macierz A:
 [[0. 3.]
 [1. 6.]
 [7. 2.]]
Wartość logiczna A!=0
 [[False  True]
 [ True  True]
 [ True  True]]
Indeksy elementow nierównych 0
[[0 1]
 [1 0]
 [1 1]
 [2 0]
 [2 1]]
Indeksy elementow mniejszych od 4 i nierownych 0
[[0 1]
 [1 0]
 [2 1]]
Indeksy elementow w pierwszej kolumnie nierownych 0
[[1]
 [2]]
```