

Use_PY_in_Linear_Algebra

June 22, 2018

1 Use PY in Linear Algebra

1.1 Chapter Zero

1. dot product 2. cross product $n \times n - 1 \times 3$.

1.1.1 python sympy numpy

sympy numpy

sympy sympy*sympydot sympycross det()inv()adjugate()

numpy numpy*arraymatrix numpydotmatrixarray numpycross numpydet() .T .H .J .A 2

1.2 Chapter One Matrix

$$m \times n \text{ (row) } n \text{ (column) } \mathbf{A}_{2 \times 3} = \begin{bmatrix} 5 & 2 & 7 \\ 1 & 3 & 4 \end{bmatrix} \mathbf{A}_{2 \times 3} \mathbf{A}_{3 \times 4} \mathbf{B}_{4 \times 4} = \begin{bmatrix} 5 & 2 & 7 & 6 \\ 1 & 3 & 4 & 2 \\ 7 & -1 & 9 & 0 \\ 8 & 2 & -2 & 3 \end{bmatrix} \mathbf{A}_{3 \times 4} \mathbf{A}_{2,2} a_{2,2}$$

python numpy ndarray matrix

In [11]: *#first method to store the matrix*

```
import numpy as np
```

```
a = np.matrix('5 2 7;1 3 4')
```

```
b = np.matrix('15 2 7 6;1 3 4 2;7 -1 9 0;8 2 -2 3')
```

```
print(a)#print matrix a
```

```
print(b)#print matrix b
```

```
[[5 2 7]
 [1 3 4]]
[[15  2  7  6]
 [ 1  3  4  2]
 [ 7 -1  9  0]
 [ 8  2 -2  3]]
[[5 2 7]
```

```

[1 3 4]]
[[15  2  7  6]
 [ 1  3  4  2]
 [ 7 -1  9  0]
 [ 8  2 -2  3]]

```

In [12]: *#second method to store the matrix*

```
import numpy as np
```

```

a = np.matrix([[5,2,7],[1,3,4]])
b = np.matrix([[5,2,7,6],[1,3,4,2],[8,2,-2,3]])
print(a)#print matrix a
print(b)#print matrix b

```

```

[[5 2 7]
 [1 3 4]]
[[ 5  2  7  6]
 [ 1  3  4  2]
 [ 8  2 -2  3]]
[[5 2 7]
 [1 3 4]]
[[ 5  2  7  6]
 [ 1  3  4  2]
 [ 8  2 -2  3]]

```

LaTeX `matrix` `ndarray` `getA()` `ndarray` `matrix` `asmatrix()`

In [13]: *b = a.getA()#turn a into ndarray*

```

print(b)
print(type(b))# the type of b
c = np.asmatrix(b)
print(c)
print(type(c))# the type of c

```

```

[[5 2 7]
 [1 3 4]]
<class 'numpy.ndarray'>
[[5 2 7]
 [1 3 4]]
<class 'numpy.matrixlib.defmatrix.matrix'>
[[5 2 7]
 [1 3 4]]
<class 'numpy.ndarray'>
[[5 2 7]
 [1 3 4]]
<class 'numpy.matrixlib.defmatrix.matrix'>

```