Types of Matrices

October 20, 2020

0.0.1 EJEMPLO 1

La función que se necesita para calcular el triangulo inferior de una matriz dada es `tril()`, mientras que para calular el triángulo inferior se ocupa la función `triu()`.

[2]:
```
# Matrices triangulares

from numpy import array, tril, triu

# Definiendo una matriz cuadrada
M = array([
    [1, 2, 3],
    [1, 2, 3],
    [1, 2, 3]
])
print(M)
# Calculando la matriz triangular inferior y superior
lower = tril(M)
upper = triu(M)
print(lower)
print(upper)
```

```
[[1 2 3]
 [1 2 3]
 [1 2 3]]
[[1 0 0]
 [1 2 0]
 [1 2 3]]
[[1 2 3]
 [0 2 3]
 [0 0 3]]
```

0.0.2 EJEMPLO 2

La función que se necesita es `diag()`.

[3]:
```
# Matriz diagonal

from numpy import array, diag
```

```
# Definiendo una matriz cuadrada
M = array([
    [1, 2, 3],
    [1, 2, 3],
    [1, 2, 3]
])
print(M)
# Extrayendo la diagonal como un vector
d = diag(M)
print(d)
# Creando la matriz diagional a partir de un vector
D = diag(d)
print(D)
```

```
[[1 2 3]
 [1 2 3]
 [1 2 3]]
[1 2 3]
[[1 0 0]
 [0 2 0]
 [0 0 3]]
```

0.0.3 EJEMPLO 3

Se utiliza la función `identity()` para crear una matriz identidad con un tamaño específico.

[1]:
```
# Matriz identidad

from numpy import identity
I = identity(3) # Matriz identidad de orden 3
print(I)
```

```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

0.0.4 EJEMPLO 4

[7]:
```
# Matriz ortogonal

from numpy import array
from numpy.linalg import inv
# Definiendo la matriz ortogonal
Q = array([
    [1, 0],
    [0, -1]
])
```

```
print(Q)
# Probando la equivalencia de la inversa
V = inv(Q)
print(Q.T) # Imprimir la transpuesta de la matriz Q
print(V)
# Probando la equivalencia de la identidad
I = Q.dot(Q.T)
print(I)
```

```
[[ 1  0]
 [ 0 -1]]
[[ 1  0]
 [ 0 -1]]
[[ 1.  0.]
 [-0. -1.]]
[[1 0]
 [0 1]]
```