

Matrices

Imanol

10/3/2021

R

Matrices

```
row = matrix(c(1,2,3,4), nrow = 1) # Para crear una matriz fila  
row
```

```
##      [,1] [,2] [,3] [,4]  
## [1,]    1    2    3    4
```

```
col = matrix(c(1,2,3,4), ncol = 1) # Para crear una matriz columna  
col
```

```
##      [,1]  
## [1,]    1  
## [2,]    2  
## [3,]    3  
## [4,]    4
```

```
# Creación de matrices con MATRIX
```

```
A = matrix(c(1,1,3,5,2,4,3,-2,-2,2,-1,3), nrow = 3, ncol = 4, byrow = TRUE)  
A
```

```
##      [,1] [,2] [,3] [,4]  
## [1,]    1    1    3    5  
## [2,]    2    4    3   -2  
## [3,]   -2    2   -1    3
```

```
B = matrix(c(1,0,2,3,3,2,1,-2,3), nrow = 3, byrow = FALSE)  
B
```

```
##      [,1] [,2] [,3]  
## [1,]    1    3    1  
## [2,]    0    3   -2  
## [3,]    2    2    3
```

```

# Creación de matrices con BIND

C = rbind(c(1,2,3),c(4,5,6),c(7,8,9)) # Por fila
C

##      [,1] [,2] [,3]
## [1,]     1     2     3
## [2,]     4     5     6
## [3,]     7     8     9

D = cbind(c(1,2,3),c(4,5,6),c(7,8,9)) # Por columna
D

##      [,1] [,2] [,3]
## [1,]     1     4     7
## [2,]     2     5     8
## [3,]     3     6     9

# Para acceder a la matriz

A[3,3] # Elemento a33

## [1] -1

A[1,] # Primera fila

## [1] 1 1 3 5

A[,2] # Segunda columna

## [1] 1 4 2

# Crear matrices de ceros y unos
O = matrix(0, nrow = 3, ncol = 3)
O

##      [,1] [,2] [,3]
## [1,]     0     0     0
## [2,]     0     0     0
## [3,]     0     0     0

Ones = matrix(1, nrow = 3, ncol = 3)
Ones

##      [,1] [,2] [,3]
## [1,]     1     1     1
## [2,]     1     1     1
## [3,]     1     1     1

```

```

# Matriz diagonal
E = diag(c(1,2,3,4,5,6))
E

##      [,1] [,2] [,3] [,4] [,5] [,6]
## [1,]     1     0     0     0     0     0
## [2,]     0     2     0     0     0     0
## [3,]     0     0     3     0     0     0
## [4,]     0     0     0     4     0     0
## [5,]     0     0     0     0     5     0
## [6,]     0     0     0     0     0     6

# Para sacar los elementos de la diagonal de una matriz
diag(A)

## [1] 1 4 -1

# Numero de filas y columnas
nrow(A)

## [1] 3

ncol(A)

## [1] 4

dim(A)

## [1] 3 4

```

Manipulación de Matrices

```

sum(A) # Suma todos los elementos de la matriz

## [1] 19

# Suma por filas y columnas
rowSums(A)

## [1] 10 7 2

colSums(A)

## [1] 1 7 5 6

```

```

# Producto de todos los elementos
prod(A)

## [1] -8640

# Media
mean(A)

## [1] 1.583333

rowMeans(A)

## [1] 2.50 1.75 0.50

colMeans(A)

## [1] 0.3333333 2.3333333 1.6666667 2.0000000

```

Operaciones de Matrices

```

# Traspuesta
A

##      [,1] [,2] [,3] [,4]
## [1,]     1     1     3     5
## [2,]     2     4     3    -2
## [3,]    -2     2    -1     3

t(A)

##      [,1] [,2] [,3]
## [1,]     1     2    -2
## [2,]     1     4     2
## [3,]     3     3    -1
## [4,]     5    -2     3

# Calcular traza de la matriz
sum(diag(A))

## [1] 4

# Operaciones
A = rbind(c(1,2,3),c(4,5,6),c(7,8,9)) # Por fila
B = rbind(c(1,0,2),c(3,0,4),c(5,0,6)) # Por fila
A+B

##      [,1] [,2] [,3]
## [1,]     2     2     5
## [2,]     7     5    10
## [3,]    12     8    15

```

5*A

```
##      [,1] [,2] [,3]
## [1,]    5   10   15
## [2,]   20   25   30
## [3,]   35   40   45
```

A%*%B # Multiplicar matrices

```
##      [,1] [,2] [,3]
## [1,]   22    0   28
## [2,]   49    0   64
## [3,]   76    0  100
```

A*B # Producto elemento a elemento

```
##      [,1] [,2] [,3]
## [1,]    1    0    6
## [2,]   12    0   24
## [3,]   35    0   54
```

Potencia enesima de una matriz
library(Biodem)
mtx.exp(A,4) # (paquete Biodem)

```
##      [,1] [,2] [,3]
## [1,] 7560 9288 11016
## [2,] 17118 21033 24948
## [3,] 26676 32778 38880
```

library(expm)

```
## Loading required package: Matrix

##
## Attaching package: 'expm'

## The following object is masked from 'package:Matrix':
## 
##     expm
```

A%^%4 # (paquete expm)

```
##      [,1] [,2] [,3]
## [1,] 7560 9288 11016
## [2,] 17118 21033 24948
## [3,] 26676 32778 38880
```

Rango e inversa de Matrices

```

# Rango
qr(A)$rank

## [1] 2

# Inversa
#solve(A)           # Si no existe da un error
#round(A%*%solve(A)) # Para ver que me da la matriz identidad

```

Python

Matrices

```

row = [1,2,3] # Para crear una matriz fila
row

## [1, 2, 3]

col = [[1],[2],[3]] # Para crear una matriz columna
col

# Creacion de matrices

## [[1], [2], [3]]

M = [[1,2,3],[4,5,6],[7,8,9]]
M

# Para acceder a la matriz

## [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

M[1][1] # Elemento a11

## 5

M[0][0] # Primera fila primera columna

## 1

M[0]    # Primera fila

# Para acceder a las columnas necesitamos numpy

## El manejo es mas comodo con numpy
##antes eran listas en python, ahora si es una matriz

```

```

## [1, 2, 3]

import numpy as np
M = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(M)
## Ademas tiene dtype() con el que podemos elegir el tipo de dato

## [[1 2 3]
##  [4 5 6]
##  [7 8 9]]

M = np.array([[1,2,3],[4,5,6],[7,8,9]], dtype = complex)
print(M)

## [[1.+0.j 2.+0.j 3.+0.j]
##  [4.+0.j 5.+0.j 6.+0.j]
##  [7.+0.j 8.+0.j 9.+0.j]]

M = np.array([[1,2,3],[4,5,6],[7,8,9]], dtype = float)
print(M)

## Acceso con numpy

## [[1. 2. 3.]
##  [4. 5. 6.]
##  [7. 8. 9.]]

M = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(M)

## [[1 2 3]
##  [4 5 6]
##  [7 8 9]]

M[0][2]

## 3

print(M[1])

## [4 5 6]

M[1,:] # Segunda fila

## array([4, 5, 6])

```

```

M[:,0] # Primera columna

# Crear matrices de ceros y unos

## array([1, 4, 7])

print(np.zeros((5,7)))

## [[0. 0. 0. 0. 0. 0. 0.]
## [0. 0. 0. 0. 0. 0. 0.]
## [0. 0. 0. 0. 0. 0. 0.]
## [0. 0. 0. 0. 0. 0. 0.]
## [0. 0. 0. 0. 0. 0. 0.]]

print(np.ones((5,7)))

# Matriz diagonal

## [[1. 1. 1. 1. 1. 1. 1.]
## [1. 1. 1. 1. 1. 1. 1.]
## [1. 1. 1. 1. 1. 1. 1.]
## [1. 1. 1. 1. 1. 1. 1.]
## [1. 1. 1. 1. 1. 1. 1.]]

x = [1,2,3,4]
N = np.diag(x) # Pasando un vector
N

# Para sacar los elementos de la diagonal de una matriz

## array([[1, 0, 0, 0],
##        [0, 2, 0, 0],
##        [0, 0, 3, 0],
##        [0, 0, 0, 4]])

np.diag(N) # Pasando una matriz

# Numero de filas y columnas

## array([1, 2, 3, 4])

np.shape(M)

## (3, 3)

```

Manipulación de Matrices

```

# Suma todos los elementos de la matriz
np.sum(M)

# Suma por filas y columnas

## 45

np.sum(M, axis = 0) # Fila

## array([12, 15, 18])

np.sum(M, axis = 1) # Columna

# Producto de todos los elementos

## array([ 6, 15, 24])

np.prod(M) # Cuidado con hacer overflow

# Media

## 362880

np.mean(M) # media de toda la matriz

```

```

## 5.0

np.mean(M, axis = 0) # Media por filas

## array([4., 5., 6.])

np.mean(M, axis = 1) # Media por columnas

## array([2., 5., 8.])

```

Operaciones de Matrices

```

# Traspuesta
print(M.transpose())

# Calcular traza de la matriz

## [[1 4 7]
##  [2 5 8]
##  [3 6 9]]

```

```

print(M.trace())

# Suma matrices

## 15

A = np.array([[1,2],[2,0]])
B = np.array([[3,0],[1,4]])
print(A+B)

# Producto escalar por matriz

## [[4 2]
##  [3 4]]

print(5*A)

# Producto de matrices

## [[ 5 10]
##  [10  0]]

print(A.dot(B))

# Producto elemento a elemento

## [[5 8]
##  [6 0]]

print(A*B)

# Potencia de una matriz

## [[3 0]
##  [2 0]]

print(np.linalg.matrix_power(A,5))

## [[65 58]
##  [58 36]]

```

Rango e inversa de Matrices

```

# Rango
np.linalg.matrix_rank(A)

## 2

```

```

np.linalg.matrix_rank(B)

# Inversa

## 2

print(np.linalg.inv(A))

## [[ 0.      0.5 ]
## [ 0.5   -0.25]]

print(np.linalg.inv(A).dot(A)) # Comprobamos (nos deberia dar la matriz identidad)

## [[1.  0.]
## [ 0.  1.]]

```

Matlab

Matrices

```

% Suma de elementos de una matriz
M = [1 2 3; 4 5 6; 7 8 9];
suma = sum(sum(M)) % Primer sum suma por fila y segundo sum por columna

% Suma de elementos por filas y por columnas
M = [1 2 3; 4 5 6; 7 8 9];
sumaFil = sum(M,2)
sumaCol = sum(M,1) % O sumaCol = sum(M)

% Multiplicar elementos de una matriz
M = [1 2 3; 4 5 6; 7 8 9];
producto = prod(prod(M))

% Media
M = [1 2 3; 4 5 6; 7 8 9];
media = mean(mean(M))
mediaFil = mean(M,2)
mediaCol = mean(M,1)

```

Operaciones con matrices

```

% Traspuesta
M = [1 2 3; 4 5 6; 7 8 9];
M'

% Traza de la matriz

```

```

A = [1 -1; 0 3];
B = [2 1; -1 0];
trace(A)
trace(B)

% Suma matrices
%Siempre que sean de la misma dimension se pueden llevar a cabo
A = [1 -1; 0 3];
B = [2 1; -1 0];
A+B

% Producto escalar por matrices
A = [1 -1; 0 3];
2*A

% Producto de matrices
A = [1 -1; 0 3];
B = [2 1; -1 0];
A*B

% Potencia de matrices
A = [1 -1; 0 3];
B = [2 1; -1 0];
A^3
B^4

```

Rango e inversa de Matrices

```

% Rango
A = [1 -1; 0 3];
B = [2 1; -1 0];
rank(A)
rank(B)

% Inversa
A = [1 -1; 0 3];
B = [2 1; -1 0];
inv(A)
inv(A)*A % Comprobamos
inv(B)
inv(B)*B % Comprobamos

```