Repaso álgebra lineal

Alexis Frías Domínguez

9/8/2020

Repaso de algebra lineal

Función apply()

- **apply(matriz, MARGIN=. . . , FUN=funcion)** para aplicar otras funciones a las filas o las columnas de una matriz
  - **MARGIN** ha de ser 1 si queremos aplicar la función por filas; 2 si queremos aplicarla por columnas; c(1,2) si la queremos aplicar a cada entrada

```
a = c(1:12)
a = matrix(a, ncol = 3)
a
```

```
##      [,1] [,2] [,3]
## [1,]    1    5    9
## [2,]    2    6   10
## [3,]    3    7   11
## [4,]    4    8   12
```

```
cuadrado = function(vec){vec**2}
apply(a, MARGIN = c(1,2), FUN = cuadrado)
```

```
##      [,1] [,2] [,3]
## [1,]    1   25   81
## [2,]    4   36  100
## [3,]    9   49  121
## [4,]   16   64  144
```

Operaciones

- **t(matriz)** para obtener la transpuesta de la matriz

```
t(a)
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    5    6    7    8
## [3,]    9   10   11   12
```

- + para sumar matrices

```
b = matrix(c(12:23), ncol = 3)
b
```

```
##      [,1] [,2] [,3]
## [1,]   12   16   20
## [2,]   13   17   21
## [3,]   14   18   22
## [4,]   15   19   23
```

```
a+b
```

```
##      [,1] [,2] [,3]
## [1,]   13   21   29
## [2,]   15   23   31
## [3,]   17   25   33
## [4,]   19   27   35
```

- "*" para el producto de un escalar por una matriz

```
a*b
```

```
##      [,1] [,2] [,3]
## [1,]   12   80  180
## [2,]   26  102  210
## [3,]   42  126  242
## [4,]   60  152  276
```

- "%*%" para multiplicar matrices
- **mtx.exp(matriz,n)** para elevar la matriz a n
  - Del paquete **Biodem**
    - No calculas las potencias exactas, las aproxima

```
library(Biodem)
test<-matrix(c(1:16), 4,4)
pow.test<-mtx.exp(test,10)
pow.test
```

```
##              [,1]         [,2]         [,3]         [,4]
## [1,] 2.568153e+14 5.933637e+14 9.299121e+14 1.266461e+15
## [2,] 2.908200e+14 6.719305e+14 1.053041e+15 1.434152e+15
## [3,] 3.248247e+14 7.504974e+14 1.176170e+15 1.601843e+15
## [4,] 3.588295e+14 8.290642e+14 1.299299e+15 1.769534e+15
```

- %% para elevar matrices
  - Del paquete **expm**
    - No calcula las potencias exactas, las aproxíma

Operaciones

- **det(matriz)** para calcular el determinante de la matriz c = matrix(1:4, 2,2) det(c)
- **qr(matriz)$rank** para calcular la inversa de una matriz invertible

```
qr(a)$rank
```

```
## [1] 2
```

- **solve(matriz)** para calcular la inversa de una matriz invertible
  - También sirve para resolver sistemas de ecuaciones lineales. Para ello introducimos **solve(matriz, b)** donde b es el vector de términos independientes.

```
y = matrix(1:4, ncol = 2)
solve(y)
```

```
##      [,1] [,2]
## [1,]   -2  1.5
## [2,]    1 -0.5
```

```
j = c(1:2)
j
```

```
## [1] 1 2
```

```
solve(y,j)
```

```
## [1] 1 0
```

Valores y vectores propios

- **eigen(matriz)** para calcular los valores (vaps) y vectores propios (veps)
  - **eigen(matriz)$values** nos da el vector con los vaps de la matriz en orden decreciente de su valor absoluto y repetidos tantas veces como su multiplicidad algebraica.
  - **eigen(matriz)$vectors** nos da una matriz cuyas columnas son los veps de la matriz.

```
eigen(y)
```

```
## eigen() decomposition
## $values
## [1]  5.3722813 -0.3722813
##
## $vectors
##            [,1]       [,2]
## [1,] -0.5657675 -0.9093767
## [2,] -0.8245648  0.4159736
```