

Tarea8

Imanol

13/3/2021

A1

R

```
library(matlib)

A = rbind(c(0,2,-3,4),c(0,0,-5,-1),c(5,-1,-2,0),c(-2,0,4,6))
luA = LU(A)

# Le puedo pedir la matriz P (se ha permutado la primera y segunda fila)
luA$P
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    0    1    0    0
## [2,]    1    0    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

```
# Le puedo pedir la matriz L
luA$L
```

```
##      [,1] [,2]      [,3] [,4]
## [1,]    1  0.0  0.000000    0
## [2,]    0  1.0  0.000000    0
## [3,]    0 -0.5  1.000000    0
## [4,]    0  0.0 -1.142857    1
```

```
# Le puedo pedir la matriz U
luA$U
```

```
##      [,1] [,2] [,3]      [,4]
## [1,] 0.000000    0 -5.0 -1.000000
## [2,] 0.000000    2 -3.0  4.000000
## [3,] 5.000000    0 -3.5  2.000000
## [4,] 3.714286    0  0.0  8.285714
```

```
# Comprobacion
round(luA$L%*%luA$U) == luA$P%*%A
```

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

Python

```
import scipy
import scipy.linalg
```

```
A1 = scipy.array([[0,2,-3,4], [0,0,-5,-1], [5,-1,-2,0], [-2,0,4,6]])
```

```
## <string>:1: DeprecationWarning: scipy.array is deprecated and will be removed in SciPy 2.0.0, use numpy
```

```
P, L, U = scipy.linalg.lu(A1)
```

```
# Le puedo pedir la matriz P
P # Hay permutacion
```

```
# Le puedo pedir la matriz L
```

```
## array([[0., 1., 0., 0.],
##        [0., 0., 1., 0.],
##        [1., 0., 0., 0.],
##        [0., 0., 0., 1.]])
```

```
L
```

```
# Le puedo pedir la matriz U
```

```
## array([[ 1. ,  0. ,  0. ,  0. ],
##        [ 0. ,  1. ,  0. ,  0. ],
##        [ 0. ,  0. ,  1. ,  0. ],
##        [-0.4 , -0.2 , -0.52,  1. ]])
```

```
U
```

```
# Comprobacion
```

```
## array([[ 5. , -1. , -2. ,  0. ],
##        [ 0. ,  2. , -3. ,  4. ],
##        [ 0. ,  0. , -5. , -1. ],
##        [ 0. ,  0. ,  0. ,  6.28]])
```

```
L.dot(U) == (P.transpose()).dot(A1) # Hay que transponer la matriz P porque Python permuta por columnas
```

```
## array([[ True,  True,  True,  True],
##        [ True,  True,  True,  True],
##        [ True,  True,  True,  True],
##        [ True,  True,  True,  True]])
```

Matlab

```
A1 = [0 2 -3 4;
      0 0 -5 -1;
      5 -1 -2 0;
      -2 0 4 6]

% Factorizacion
[L,U,P] = lu(A1) % Hay permutaciones

% Comprobacion
round(L*U) == P*A1
```

A2

R

```
library(matlib)

A2 = rbind(c(1,2,-1,4),c(0,-1,5,8),c(2,3,1,4),c(1,-1,6,4))
luA = LU(A2)

# Le puedo pedir la matriz P (no se ha permutado ninguna fila)
luA$P
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

```
# Le puedo pedir la matriz L
luA$L
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    2    1    1    0
## [4,]    1    3    4    1
```

```
# Le puedo pedir la matriz U
luA$U
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2   -1    4
## [2,]    0   -1    5    8
## [3,]    0    0   -2  -12
## [4,]    0    0    0   24
```

```
# Comprobacion
luA$L%*%luA$U == A2
```

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

Python

```
import scipy
import scipy.linalg

A2 = scipy.array([[1,2,-1,4], [0,-1,5,8], [2,3,1,4], [1,-1,6,4]])
P, L, U = scipy.linalg.lu(A2)
```

```
# Le puedo pedir la matriz P
P # Hay permutacion
```

```
# Le puedo pedir la matriz L
```

```
## array([[0., 0., 0., 1.],
##        [0., 0., 1., 0.],
##        [1., 0., 0., 0.],
##        [0., 1., 0., 0.]])
```

```
L
```

```
# Le puedo pedir la matriz U
```

```
## array([[ 1.         ,  0.         ,  0.         ,  0.         ],
##        [ 0.5        ,  1.         ,  0.         ,  0.         ],
##        [ 0.         ,  0.4        ,  1.         ,  0.         ],
##        [ 0.5        , -0.2        , -0.14285714,  1.         ]])
```

```
U
```

```
# Comprobacion
```

```
## array([[ 2.          ,  3.          ,  1.          ,  4.          ],
##        [ 0.          , -2.5         ,  5.5         ,  2.          ],
##        [ 0.          ,  0.          ,  2.8         ,  7.2         ],
##        [ 0.          ,  0.          ,  0.          ,  3.42857143]])
```

```
L.dot(U) == (P.transpose()).dot(A2) # Hay que transponer la matriz P porque Python permuta por columnas
```

```
## array([[ True,  True,  True,  True],
##        [ True,  True,  True,  True],
##        [ True,  True,  True,  True],
##        [ True,  True,  True,  True]])
```

Matlab

```
A2 = [1 2 -1 4;
      0 -1 5 8;
      2 3 1 4;
      1 -1 6 4]

% Factorizacion
[L,U,P] = lu(A2) % Hay permutaciones

% Comprobacion
L*U == P*A2
```

A3

R

```
library(matlib)

A3 = rbind(c(2,4,-2,0),c(3,7,5,-4),c(-1,2,-2,5),c(6,1,0,2))
luA = LU(A3)

# Le puedo pedir la matriz P (no se ha permutado ninguna fila)
luA$P
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

```
# Le puedo pedir la matriz L
luA$L
```

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

```
## [1,] 1.0 0 0.000000 0
## [2,] 1.5 1 0.000000 0
## [3,] -0.5 4 1.000000 0
## [4,] 3.0 -11 -2.685714 1
```

```
# Le puedo pedir la matriz U
luA$U
```

```
##      [,1] [,2] [,3] [,4]
## [1,] 2 4 -2 0.0
## [2,] 0 1 8 -4.0
## [3,] 0 0 -35 21.0
## [4,] 0 0 0 14.4
```

```
# Comprobacion
round(luA$L%*%luA$U) == A3
```

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

Python

```
import scipy
import scipy.linalg

A3 = scipy.array([[2,4,-2,0], [3,7,5,-4], [-1,2,-2,5], [6,1,0,2]])
P, L, U = scipy.linalg.lu(A3)
```

```
# Le puedo pedir la matriz P
P # Hay permutacion
```

```
# Le puedo pedir la matriz L
```

```
## array([[0., 0., 1., 0.],
##        [0., 1., 0., 0.],
##        [0., 0., 0., 1.],
##        [1., 0., 0., 0.]])
```

```
L
```

```
# Le puedo pedir la matriz U
```

```
## array([[ 1.          ,  0.          ,  0.          ,  0.          ],
##        [ 0.5          ,  1.          ,  0.          ,  0.          ],
##        [ 0.33333333,  0.56410256,  1.          ,  0.          ],
##        [-0.16666667,  0.33333333,  0.7606383 ,  1.          ]])
```

U

Comprobacion

```
## array([[ 6.         ,  1.         ,  0.         ,  2.         ],
##        [ 0.         ,  6.5        ,  5.         , -5.         ],
##        [ 0.         ,  0.         , -4.82051282,  2.15384615],
##        [ 0.         ,  0.         ,  0.         ,  5.36170213]])
```

```
import numpy as np
```

```
np.round(L.dot(U)) == (P.transpose()).dot(A3) # Hay que transponer la matriz P porque Python permuta po
```

```
## array([[ True,  True,  True,  True],
##        [ True,  True,  True,  True],
##        [ True,  True,  True,  True],
##        [ True,  True,  True,  True]])
```

Matlab

```
A3 = [2 4 -2 0;
      3 7 5 -4;
      -1 2 -2 5;
      6 1 0 2]
```

% Factorizacion

```
[L,U,P] = lu(A3) % Hay permutaciones
```

% Comprobacion

```
round(L*U) == P*A3
```

A4

R

```
library(matlib)
```

```
A4 = rbind(c(0,2,3,1),c(0,4,-1,5),c(2,0,3,1),c(1,-4,5,6))
luA = LU(A4)
```

Le puedo pedir la matriz P (Permutan la fila 1 y 2)

```
luA$P
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    0    1    0    0
## [2,]    1    0    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

```
# Le puedo pedir la matriz L
luA$L
```

```
##      [,1] [,2]      [,3] [,4]
## [1,]    1    0 0.000000    0
## [2,]    0    1 0.000000    0
## [3,]    0    0 1.000000    0
## [4,]    0   -2 3.666667    1
```

```
# Le puedo pedir la matriz U
luA$U
```

```
##      [,1] [,2] [,3]      [,4]
## [1,] 0.000000    4   -1 5.000000
## [2,] 0.000000    2    3 1.000000
## [3,] 2.000000    0    3 1.000000
## [4,] -6.333333    0    0 4.333333
```

```
# Comprobacion
round(luA$L%*%luA$U) == luA$P%*%A4
```

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

Python

```
import scipy
import scipy.linalg

A4 = scipy.array([[0,2,3,1], [0,4,-1,5], [2,0,3,1], [1,-4,5,6]])
P, L, U = scipy.linalg.lu(A4)
```

```
# Le puedo pedir la matriz P
P # Hay permutacion
```

```
# Le puedo pedir la matriz L
```

```
## array([[0., 0., 1., 0.],
##        [0., 1., 0., 0.],
##        [1., 0., 0., 0.],
##        [0., 0., 0., 1.]])
```

```
L
```

```
# Le puedo pedir la matriz U
```

```
## array([[ 1.          ,  0.          ,  0.          ,  0.          ],
##        [ 0.          ,  1.          ,  0.          ,  0.          ],
##        [ 0.          ,  0.5        ,  1.          ,  0.          ],
##        [ 0.5        , -1.          ,  0.71428571,  1.          ]])
```

U

Comprobacion

```
## array([[ 2.          ,  0.          ,  3.          ,  1.          ],
##        [ 0.          ,  4.          , -1.          ,  5.          ],
##        [ 0.          ,  0.          ,  3.5         , -1.5         ],
##        [ 0.          ,  0.          ,  0.          , 11.57142857]])
```

L.dot(U) == (P.transpose()).dot(A4) # Hay que transponer la matriz P porque Python permuta por columnas

```
## array([[ True,  True,  True,  True],
##        [ True,  True,  True,  True],
##        [ True,  True,  True,  True],
##        [ True,  True,  True,  True]])
```

Matlab

```
A4 = [0 2 3 1;
      0 4 -1 5;
      2 0 3 1;
      1 -4 5 6]
```

% Factorizacion

[L,U,P] = lu(A4) % Hay permutaciones

% Comprobacion

*L*U == P*A4*