Una matriz es una transformación lineal

```
a = pi / 4;

Rc = [cos(a) -sin(a) ; sin(a) cos(a)] % Rotamos
```

```
Rc = 2×2
    0.7071   -0.7071
    0.7071    0.7071
```

```
x = [1 ; 2];
y = Rc * x;

xangle = atan2d(x(2), x(1))
```

```
xangle = 63.4349
```

```
yangle = atan2d(y(2), y(1))
```

```
yangle = 108.4349
```

Polinomio característico

```
A = gallery(3)
```

```
A = 3×3
  -149   -50  -154
   537   180   546
   -27    -9   -25
```

```
p = poly(A) % Polinomio característico de A
```

```
p = 1×4
    1.0000   -6.0000   11.0000   -6.0000
```

```
l = roots(p) % Valores propios de A
```

```
l = 3×1
    3.0000
    2.0000
    1.0000
```

```
radioEspectral = max(abs(l))
```

```
radioEspectral = 3.0000
```

Función eigenvalores

```
[V,D] = eig(A)
```

```
V = 3×3
    0.3162   -0.4041   -0.1391
   -0.9487    0.9091    0.9740
   -0.0000    0.1010   -0.1789
D = 3×3
    1.0000         0         0
```

```
         0    2.0000         0
         0         0    3.0000
```

```
% Cada columna de V tiene norma 1; es un vector propio normalizado
diag(D)
```

```
ans = 3×1
    1.0000
    2.0000
    3.0000
```

```
traza = trace(A)
```

```
traza = 6
```

```
determinante = det(A)
```

```
determinante = 6.0000
```

Eigenvalores de potencias

```
A2 = A^2
```

```
A2 = 3×3
        -491        -164        -504
        1905         636        1932
        -135         -45        -131
```

```
[V2, D2] = eig(A2)
```

```
V2 = 3×3
   -0.3162    0.4041   -0.1391
    0.9487   -0.9091    0.9740
   -0.0000   -0.1010   -0.1789
D2 = 3×3
    1.0000         0         0
         0    4.0000         0
         0         0    9.0000
```

```
A3 = A^3
```

```
A3 = 3×3
       -1301        -434       -1330
        5523        1842        5586
        -513        -171        -505
```

```
[V3, D3] = eig(A3)
```

```
V3 = 3×3
    0.1391   -0.3162   -0.4041
   -0.9740    0.9487    0.9091
    0.1789    0.0000    0.1010
D3 = 3×3
   27.0000         0         0
         0    1.0000         0
         0         0    8.0000
```

```
[VI, DI] = eig(inv(A))
```

```
VI = 3×3
   -0.3162    0.4041   -0.1391
    0.9487   -0.9091    0.9740
    0.0000   -0.1010   -0.1789
DI = 3×3
    1.0000         0         0
         0    0.5000         0
         0         0    0.3333
```

Valores y vectores complejos

```
AC = [3 -2 ; 4 -1]
```

```
AC = 2×2
     3    -2
     4    -1
```

```
[V4, D4] = eig(AC) % complejos y conjugados
```

```
V4 = 2×2 complex
   0.4082 + 0.4082i   0.4082 - 0.4082i
   0.8165 + 0.0000i   0.8165 + 0.0000i
D4 = 2×2 complex
   1.0000 + 2.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   1.0000 - 2.0000i
```

Matrices simétricas

```
% una matriz es simétrica
S = [1 2 3 ; 2 4 5 ; 3 5 6]
```

```
S = 3×3
     1     2     3
     2     4     5
     3     5     6
```

```
ST = transpose(S) % también se puede con S'
```

```
ST = 3×3
     1     2     3
     2     4     5
     3     5     6
```

```
resp = isequal(S, ST)
```

```
resp = logical
   1
```

```
[VS, DS] = eig(S) % todos los valores propios de una matriz simétrica son reales
```

```
VS = 3×3
    0.7370    0.5910    0.3280
    0.3280   -0.7370    0.5910
   -0.5910    0.3280    0.7370
DS = 3×3
   -0.5157         0         0
         0    0.1709         0
```

```
         0         0   11.3448
```

```
% checar que son ortonormales los vectores columna de S
```

Una matriz A es matrices positivas definidas es simétrica y además `x' * Ax` es mayor que cero para toda `x > 0`.

```
PD = [2 -1 0 ; -1 2 -1 ; 0 -1 2]
```

```
PD = 3×3
     2    -1     0
    -1     2    -1
     0    -1     2
```

```
PDT = PD'
```

```
PDT = 3×3
     2    -1     0
    -1     2    -1
     0    -1     2
```

```
simetrica = isequal(PD, PDT)
```

```
simetrica = logical
   1
```

```
eig(PD) % Todos son reales y positivos
```

```
ans = 3×1
    0.5858
    2.0000
    3.4142
```

Matrices triangulares y diagonales

```
NT = [1 0 0 ; 1 2 0 ; 2 3 3] % triangular inferior
```

```
NT = 3×3
     1     0     0
     1     2     0
     2     3     3
```

```
eig(NT) % son los elementos de la diagonal
```

```
ans = 3×1
     3
     2
     1
```

```
MD = [2 0 0 ; 0 3 0 ; 0 0 5]
```

```
MD = 3×3
     2     0     0
     0     3     0
     0     0     5
```

```
eig(MD)
```

```
ans = 3×1
     2
     3
     5
```

```
% Para matrices trianguales y matrices diagonales, los valores propios son
% los elementos de la diagonal.
```

Vectores

```
v = [16 5 9 4 2 11 7 14]
```

```
v = 1×8
    16     5     9     4     2    11     7    14
```

```
v(3) % extraer un elemento
```

```
ans = 9
```

```
v156 = v([1 5 6]) % extraer 1, 5 y 6 del vector
```

```
v156 = 1×3
    16     2    11
```

```
v37 = v(3:7)% extraer del 3 al 7
```

```
v37 = 1×5
     9     4     2    11     7
```

```
v58 = v(5:end)
```

```
v58 = 1×4
     2    11     7    14
```

```
v_impar = v(1:2:end)
```

```
v_impar = 1×4
    16     9     2     7
```

```
v_par = v(2:2:end)
```

```
v_par = 1×4
     5     4    11    14
```

```
v([2 3]) = 30
```

```
v = 1×8
    16    30    30     4     2    11     7    14
```

```
v([2 3 4]) = [10 15 20]
```

```
v = 1×8
    16    10    15    20     2    11     7    14
```

```
v2 = [v(5:end) v(1:4)]
```

```
v2 = 1×8
     2    11     7    14    16    10    15    20
```

```
v2Marcelo = v([5:end 1:4])
```

```
v2Marcelo = 1×8
     2    11     7    14    16    10    15    20
```

Matrices

```
A = magic(4) % la suma es la misma en las filas y columnas
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
sumf1 = sum(A(1, :))
```

```
sumf1 = 34
```

```
sumc3 = sum(A(:, 3))
```

```
sumc3 = 34
```

```
elem10 = A(2, 3)
```

```
elem10 = 10
```

```
A(:) % los muestra todos columna por columna
```

```
ans = 16×1
    16
     5
     9
     4
     2
    11
     7
    14
     3
    10
     ⋮
```

```
minA = min(A) % regresa el min de cada columna
```

```
minA = 1×4
```

```
     4     2     3     1
```

```
minglobal = min(min(A))
```

```
minglobal = 1
```

```
[valmin, indice] = min(A(:)) % ubicacion en el vector columna
```

```
valmin = 1
indice = 16
```

```
[renglon, columna] = ind2sub(size(A), indice) % ubicacion en la matriz
```

```
renglon = 4
columna = 4
```

Multiplicación de matrices

```
A = [1 2 -1 ; 2 1 -2 ; -3 1 1]
```

```
A = 3×3
     1     2    -1
     2     1    -2
    -3     1     1
```

```
x = [3 ; 1 ; 2]
```

```
x = 3×1
     3
     1
     2
```

```
y = A * x
```

```
y = 3×1
     3
     3
    -6
```

```
A(1, :) * x
```

```
ans = 3
```

```
A(2, 2:3) * x(2:3)
```

```
ans = -3
```

Intercambio de renglones

```
A([2 3],:) = A([3 2],:) % intercambiar renglones 2 y 3
```

```
A = 3×3
     1     2    -1
    -3     1     1
     2     1    -2
```

```
A = magic(4)
```

```
A = 4×4
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
```

```
% matriz de permutacion
P = [0 0 0 1 ; 1 0 0 0 ; 0 0 1 0 ; 0 1 0 0]
```

```
P = 4×4
     0     0     0     1
     1     0     0     0
     0     0     1     0
     0     1     0     0
```

```
PA = P * A % intercambiar los renglones de A segun de la matriz de permutacion
```

```
PA = 4×4
     4    14    15     1
    16     2     3    13
     9     7     6    12
     5    11    10     8
```