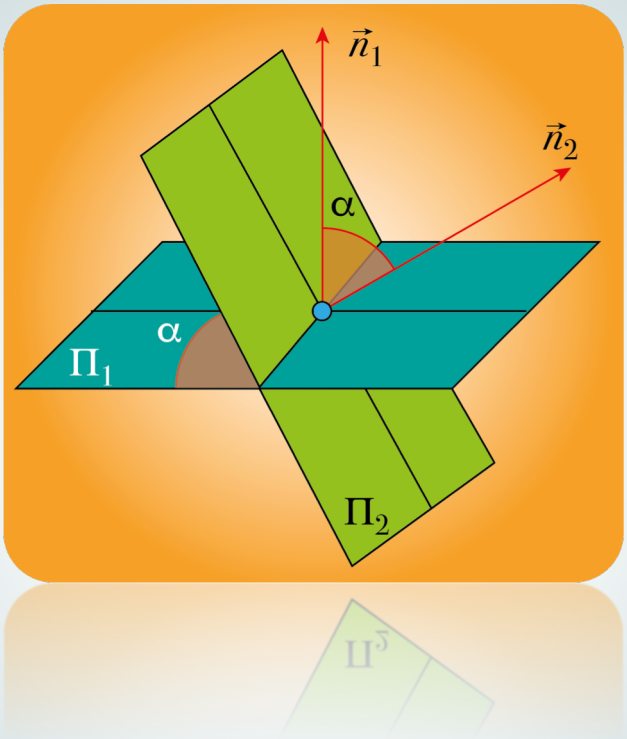
Álgebra y Geometría

Práctica 2. Trabajo con matrices



Rodrigo García Manzanas
Ruth Carballo Fidalgo

Departamento de Matemática Aplicada y
Ciencias de la Computación

Grado en Ingeniería Civil

G1954: Álgebra y Geometría

Práctica 2: Trabajo con matrices

Rodrigo García Manzanas (rodrigo.manzanas@unican.es)

Ruth Carballo Fidalgo (ruth.carballo@unican.es)

Objetivos

- Afianzar el manejo básico de matrices
- Calcular la forma escalonada, escalonada reducida y el rango de una matriz
- Calcular la factorización de matrices LU y la de Choleski

Forma escalonada (y rango)

Veamos con un ejemplo cuál sería el procedimiento para obtener una forma escalonada (no reducida) de una matriz dada utilizando matrices elementales:

```
A = [2 4 -3; 1 2 1.5; 5 5 1]  % matriz 3x3 de la que voy a calcular su
```

```
A = 3×3
    2.0000    4.0000   -3.0000
    1.0000    2.0000    1.5000
    5.0000    5.0000    1.0000
```

```
% escalonada por filas
I = eye(3);  % matriz identidad
E1 = I; E1(2,:) = E1(2,:) - (1/2)*E1(1,:); A1 = E1*A  %E1: primera matriz elemental,
```

```
A1 = 3×3
     2     4    -3
     0     0     3
     5     5     1
```

```
% A1: matriz A tras la primera operación elemental
E2 = I; E2(3,:) = E2(3,:) - (5/2)*E2(1,:); A2 = E2*A1  %E2: segunda matriz elemental,
```

```
A2 = 3×3
    2.0000    4.0000   -3.0000
         0         0    3.0000
         0   -5.0000    8.5000
```

```
% A2: matriz A tras la segunda operación elemental
E3 = I; E3([2 3],:) = E3([3 2],:); A3 = E3*A2  %E3: tercera matriz elemental,
```

```
A3 = 3×3
    2.0000    4.0000   -3.0000
         0   -5.0000    8.5000
         0         0    3.0000
```

```
% A3: matriz A tras la tercera operación elemental (ya tengo la forma escalonada)
```

La matriz $A3$ es equivalente a la matriz de partida A y, por tanto, tendrá el mismo rango. Dado que $A3$ tiene tres pivotes podemos deducir que el rango de A será 3 (podríamos haber llegado hasta aquí aplicando directamente el comando *rank*):

```
 rank(A)  % comprobación rápida
```

```
ans = 3
```

El procedimiento que hemos seguido nos permite obtener la factorización en matrices elementales de la matriz de partida, A:

```
E = E3*E2*E1;  % producto de matrices elementales
A - inv(E)*A3  % comprobación
```

```
ans = 3×3
     0     0     0
     0     0     0
     0     0     0
```

Como sabes, esta factorización en matrices elementales es útil para el cálculo de inversas y determinantes:

```
inv(A) - inv(A3)*E  % comprobación
```

```
ans = 3×3
10^-18 ×
         0         0         0
         0         0         0
         0         0   -0.9252
```

```
det(A) - det(inv(E))*det(A3)  % fíjate en que A = E^-1*A3
```

```
ans = 0
```

Forma escalonada reducida (y rango)

Sea la matriz $A = \begin{bmatrix} -2 & 1 & 0 \\ 0 & 4 & 1 \\ 1 & 0 & 1 \\ -3 & 7 & 2 \end{bmatrix}$

```
A = [-2 1 0; 0 4 1; 1 0 1; -3 7 2];  % defino la matriz A
```

El comando *rref* permite encontrar la forma escalonada reducida (por filas) directamente:

```
Ared = rref(A)  % Ared es la forma reducida por filas
```

```
Ared = 4×3
     1     0     0
     0     1     0
     0     0     1
     0     0     0
```

rref admite un segundo argumento de salida que nos devuelve un índice que identifica a las columnas pivotales:

```
[Ared, cp] = rref(A)  % cp es un vector que contiene los índices que
```

```
Ared = 4×3
     1     0     0
     0     1     0
     0     0     1
     0     0     0
cp = 1×3
     1     2     3
```

```
% identifican las columnas pivotales
```

Por tanto, el rango de la matriz puede ser fácilmente obtenido a partir de este vector de índices:

```
length(cp)  % rango = número de columnas pivotales
```

```
ans = 3
```

```
rank(A)  % comprobación
```

```
ans = 3
```

Factorización LU

El comando *lu* de MATLAB permite hallar la factorización LU de una matriz dada:

```
A = [6 1 2 4; 1 4 1 0; 3 1 4 1; 0 0 1 2]  % matriz a factorizar
```

```
A = 4×4
     6     1     2     4
     1     4     1     0
     3     1     4     1
     0     0     1     2
```

```
[L, U] = lu(A)  % L es la matriz triangular inferior y U la triangular
```

```
L = 4×4
    1.0000         0         0         0
    0.1667    1.0000         0         0
    0.5000    0.1304    1.0000         0
         0         0    0.3433    1.0000
U = 4×4
    6.0000    1.0000    2.0000    4.0000
         0    3.8333    0.6667   -0.6667
         0         0    2.9130   -0.9130
         0         0         0    2.3134
```

```
% superior que buscamos
A - L*U  % comprobación
```

```
ans = 4×4
10^-15 ×
         0         0         0         0
         0         0         0         0
         0         0         0         0
         0         0         0    0.2220
```

En el caso de que la matriz de partida A no admita factorización LU (por ejemplo por tener un 0 en el elemento $a_{1,1}$), el propio comando *lu* permite hallar una factorización $PA = LU$, donde P es la matriz de permutaciones. Recuerda que este tipo de factorización **no es única**.

```
A = [0 4 -3 2; 1 2 1.5 3; 5 5 1 4] % matriz a factorizar (no tiene porqué ser
```

```
A = 3×4
         0    4.0000   -3.0000    2.0000
    1.0000    2.0000    1.5000    3.0000
    5.0000    5.0000    1.0000    4.0000
```

```
% cuadrada)
[L, U, P] = lu(A)    % P es la matriz de permutaciones necesaria para que PA = LU
```

```
L = 3×3
    1.0000         0         0
         0    1.0000         0
    0.2000    0.2500    1.0000
U = 3×4
    5.0000    5.0000    1.0000    4.0000
         0    4.0000   -3.0000    2.0000
         0         0    2.0500    1.7000
P = 3×3
     0     0     1
     1     0     0
     0     1     0
```

```
P*A - L*U   % comprobación
```

```
ans = 3×4
10^-15 ×
         0         0         0         0
         0         0         0         0
         0         0    0.2220         0
```

Factorización de Cholesky

MATLAB también dispone de un comando específico para el cálculo de la factorización de Cholesky, *chol.* Como ya se ha visto, este tipo de factorización sólo se puede aplicar en el caso de matrices reales simétricas definidas positivas. Creemos por tanto, en primer lugar, una matriz de este tipo:

```
A = [2 1 0 0; 1 4 1 0; 0 1 4 1; 0 0 1 2]  % matriz 4x4 simétrica definida positiva
```

```
A = 4×4
     2     1     0     0
     1     4     1     0
```

```
     0     1     4     1
     0     0     1     2
```

```
issymmetric(A)  % operador booleano (ó lógico) que me dice si una matriz dada
```

```
ans = logical
   1
```

```
% es simétrica (1) o no (0)
det(A(1,1)), det(A(1:2, 1:2)), det(A(1:3, 1:3)), det(A)  % todos los menores
```

```
ans = 2
ans = 7
ans = 26
ans = 45
```

```
% principales son positivos -> definida positiva
```

Aplicamos ahora el comando *chol* para la factorización Cholesky:

```
L = chol(A, 'lower')  % matriz que necesitamos para la factorización de Cholesky
```

```
L = 4×4
    1.4142         0         0         0
    0.7071    1.8708         0         0
         0    0.5345    1.9272         0
         0         0    0.5189    1.3156
```

```
% El argumento de entrada opcional "lower" fuerza que la matriz L sea
% triangular inferior (por defecto, "chol" utiliza el argumento "upper",
% que da lugar a una matriz triangular superior, en cuyo caso la
% factorización resultante sería A = L'*L)
```

La matriz L que hemos hallado debe cumplir la relación $A = LL^t$. Comprobémoslo:

```
A - L*L'  % comprobación
```

```
ans = 4×4
10^-15 ×
   -0.4441         0         0         0
         0         0         0         0
         0         0    0.4441         0
         0         0         0    0.2220
```

Ten en cuenta que si la matriz de partida no fuera simétrica y definida positiva, el comando *chol* seguirá operando sin dar lugar a un error. Sin embargo, la matriz devuelta en este caso no cumplirá que $A = LL^t$

```
A = [2 1 0 0; 1 4 1 0; 0 1 4 1; 0 0 -1 2]  % matriz 4x4 NO simétrica
```

```
A = 4×4
     2     1     0     0
     1     4     1     0
     0     1     4     1
     0     0    -1     2
```

```
issymmetric(A)  % operador booleano (ó lógico) que me dice si una matriz dada
```

```
ans = logical
   0
```

```
% es simétrica (1) o no (0)
det(A(1,1)), det(A(1:2, 1:2)), det(A(1:3, 1:3)), det(A)  % todos los menores
```

```
ans = 2
ans = 7
ans = 26
ans = 59
```

```
% principales son positivos -> definida positiva
L = chol(A, 'lower')  % aplico el comando "chol"
```

```
L = 4×4
    1.4142         0         0         0
    0.7071    1.8708         0         0
         0    0.5345    1.9272         0
         0         0   -0.5189    1.3156
```

```
A - L*L'  % no se cumple que A = L*L'
```

```
ans = 4×4
   -0.0000         0         0         0
         0         0         0         0
         0         0    0.0000    2.0000
         0         0         0    0.0000
```

Ejercicios propuestos

Ejercicio 1:

Halla una forma escalonada de la siguiente matriz:

$$A = \begin{bmatrix} 2 & 4 & -3 & 2 \\ 1 & 2 & 1.5 & 3 \\ 5 & 5 & 1 & 4 \end{bmatrix}$$

Calcula su rango y halla una matriz cuadrada que multiplicada por esa forma escalonada proporcione la matriz A.

Ejercicio 2:

Hallar la factorización LU de las siguientes matrices:

$$A = \begin{bmatrix} 1 & -2 & 3 \\ 2 & -6 & 5 \\ -1 & -4 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & -2 & 3 \\ 3 & -6 & 9 \\ 2 & -3 & 1 \end{bmatrix}$$

Calcula sus inversas (si existieran) a partir de dicha factorización.

Ejercicio 3:

Halla la factorización de Cholesky (si la hubiera) y utilízala para calcular las inversas de las siguientes matrices:

$$A = \begin{bmatrix} 14 & 5 & 5 \\ 5 & 5 & 1 \\ 5 & 1 & 2 \end{bmatrix} \qquad B = \begin{bmatrix} 14 & 12 & 11 \\ 42 & 126 & 15 \\ 11 & 33 & 14 \end{bmatrix}$$