PRÁCTICA 1

Sistemas de ecuaciones lineales.

1.1 Introducción

MATLAB es un programa de cálculo científico que permite, no solo una gran variedad de manipulaciones matemáticas sino que se emplea como entorno de trabajo en diferentes campos de la ingeniería. En general, con MATLAB se puede:

- realizar operaciones aritméticas (como una calculadora pero con posibilidad de mayor precisión),
- realizar cálculo simbólico y en aritmética exacta,
- programar en un lenguaje interpretado,
- realizar gran variedad de gráficos,
- acceder a paquetes con aplicación a temas diversos como el tratamiento de señales, simulación de circuitos, etc.

Existen dos formas de actuar con MATLAB :

1. modo interactivo: donde a través de instrucciones directas al programa se obtienen de forma inmediata los resultados.
2. modo programado: mediante uno o varios ficheros escritos en el lenguaje de programación de MATLAB se opera con un mayor número de instrucciones de MATLAB que permite realizar operaciones y cálculos complejos.

La interacción con MATLAB se realiza fundamentalmente a través de la ventana de comandos. En ella escribiremos las ordenes para el programa y recibiremos, en la mayoría de los casos, la respuesta. Por ello es *muy importante observar en la pantalla el resultado de cada operación.* Así mismo, hay que tener en cuenta que:

- MATLAB distingue mayúsculas y minúsculas.
- El usuario escribe detrás del símbolo >>.
- Para ejecutar hay que pulsar la tecla “enter” después de escribir la orden.

1.2 Primeras operaciones

Para comenzar realizaremos algunas operaciones aritméticas y con matrices. Primero unas operaciones aritméticas:

```
>> 2+2
>> 2+3/2
>> 1+5*3
```

MATLAB trata por defecto todos los números como una calculadora, esto es, con una misma cantidad de cifras decimales de precisión. Además, para MATLAB casi todo son matrices, los vectores son matrices fila o columna. Incluso los escalares los trata como matrices 1×1.

Veamos en las instrucciones que siguen cómo se define una matriz 3×2, se transpone de dos modos diferentes y se realizan diferentes operaciones (los resultados los verás en la pantalla).

```
>> A=[3 4; 2 1; -1 0]
>> B=transpose(A)
>> C=A'
>> B+C
>> b+C
>> A+B
>> B+C;
```

Como habrás observado, si colocas un punto y coma al final de la línea no aparecerá el resultado aunque la operación se realiza.

1.3 Otras operaciones

Ahora veremos más operaciones que involucran a las matrices: productos de matriz por escalar, producto de matrices y de matriz por vector.

```
>> A
>> 3*A
>> B=[4 5 6; 1 2 3];
>> A*B
>> B*A
>> A*A
>> v=[2;-2]
>> A*v
```

Ejercicios

1. Realiza las operaciones siguientes

$$\begin{pmatrix} 1 & -1 & 0 \\ 3 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix} \begin{pmatrix} 4 & -1 & -3 \\ 3 & 2 & -1 \\ 1 & 2 & 5 \end{pmatrix}, \qquad \begin{pmatrix} 2 \\ -1 \\ 3 \end{pmatrix} \begin{pmatrix} 3 & 1 & 0 \end{pmatrix}$$

1.4 Algunas funciones

Además de las operaciones anteriores, MATLAB dispone de funciones que realizan tareas más complejas sobre matrices. Por ahora las que necesitaremos son las siguientes:

- `rank(A)`, calcula el rango de la matriz A
- `det(A)`, calcula el determinante de la matriz A.
- `inv(A)`, calcula la inversa de la matriz A.
- `eye(n)`, devuelve la matriz identidad de orden n.
- `help` *`comando`*, muestra una breve descripción de la función de MATLAB *`comando`*.

Ejercicios

1. Usando el comando `help` verifica la información sobre las funciones `inv`, `det`. Averigua del mismo modo qué hace la función `zeros` y pruéba alguno de sus usos.
2. Empleando las funciones anteriores, calcula el rango, el determinante y la inversa de alguna de las matrices anteriores.

1.5 Sistemas lineales

Veamos a continuación cómo MATLAB nos permite extraer elementos, filas y columnas de una matriz:

```
>> A
>> A(3,2)
>> A(1,:)
>> A(:,2)
```

Al igual que podemos extraer filas y columnas, podemos componer una matriz a partir de vectores:

```
>> u=[1;2;3]
>> v=[4;5;6]
>> C=[u v]
>> C(1,:)
>> D=[u ; v]
```

Con esta información, podemos realizar operaciones elementales sobre matrices que son la base de la eliminación gaussiana. Dada una matriz

```
>> A=[0 1 1;1 1 1;2 -1 2]
```

vamos a permutar las dos primeras filas. Para ello emplearemos un vector intermedio (`aux`) que almacenará provisionalmente una fila mientras realizamos el intercambio

```
>> aux=A(1,:)
>> A(1,:)=A(2,:)
>> A(2,:)=aux
```

siguiente operación sobre A:

```
>> A(3,:)=A(3,:)-2*A(1,:)
```

Ejercicios

1. Continúa las operaciones del ejemplo anterior hasta conseguir averiguar el rango de la matriz. Comprueba el resultado con ayuda de la función `rank`.

2. Queremos resolver el sistema lineal

$$\begin{aligned} x + y + z &= 3, \\ y + 2z &= 3, \\ 2x + 2y + 2z &= 6. \end{aligned}$$

   Realiza las operaciones elementales a la matriz ampliada del sistema hasta que averigües cuántas soluciones tiene el sistema anterior.

3. Las operaciones elementales se pueden realizar multiplicando a izquierda por una matriz elemental apropiada. Realiza cualquiera de las operaciones anteriores definiendo una matriz elemental apropiada y comprueba el resultado realizando la misma operación como hemos hecho anteriormente.

Nota: dado un sistema lineales escrito en forma matricial, $Ax = b$, MATLAB proporciona *siempre valores que son una solución del sistema en algún sentido*, incluido el caso de sistemas incompatibles, introduciendo:
>> `A\b`.

1.6 Matrices elementales

Las operaciones elementales se pueden realizar multiplicando a izquierda por una matriz elemental apropiada. Por ejemplo, para realizar las operaciones realizadas en el ejemplo anterior podemos definir, para la permutación:

```
>> P=eye(3);
>> P(1,1)=0; P(2,2)=0; P(1,2)=1; P(2,1)=1;
>> P
>> P*A
para realizar la segunda operación
>> P2=eye(3);
>> P2(3,1)=-2
>> P2*P*A
```

Calcular en cada paso la matriz elemental apropiada para cada operación es pesado y repetitivo. Para este tipo de tareas MATLAB permite construir al usuario sus propias funciones para abreviar los cálculos, esto es, empleamos el modo programado de MATLAB . Dispondremos de tres funciones que calculan cada una de las matrices elementales que necesitamos: pij(n,i,j), pijt(n,i,j,t), qit(n,i,t)

Los ficheros correspondientes tienen el mismo nombre de la función con la extensión .m. Como ejemplo, mostramos aquí la función para calcular la matriz que permuta dos filas (cuando se escribe el símbolo %, lo que viene a continuación se considera un comentario que se ignora).

```
function p=pij(n,i,j)
%
% esta funcion calcula la matriz elemental de orden n que permite
% permutar las filas i y j.
p=eye(n); % partimos inicialmente de la identidad de orden n.
p(i,i)=0;p(j,j)=0; % modificamos los elementos necesarios.
p(i,j)=1;p(j,i)=1;
%
return
```

De este modo, las dos operaciones anteriores, después de haber definido las funciones apropiadamente, se reducirán a:

```
>> pijt(3,3,1,-2)*pij(3,1,2)*A
```

Ejercicios

1. Continúa las operaciones del ejemplo anterior hasta conseguir averiguar el rango de la matriz. Comprueba el resultado con ayuda de la función *rank*.

1.7 Descomposición LU

Toda matriz A se puede transformar, empleando tan solo operaciones elementales, en una matriz triangular superior U. Si las operaciones elementales se describen mediante matrices elementales P_1, P_2,...,P_m tendremos

$$P_m \cdots P_2 P_1 A = U.$$

Si por un lado reunimos las operaciones de permutación en una matriz P y por otro lado reunimos las demás operaciones en otra matriz Q tendremos que

$$Q\,P\,A = U \qquad \Rightarrow \qquad P\,A = Q^{-1}\,U$$

Se prueba que la matriz $L = Q^{-1}$ es triangular inferior con unos en la diagonal principal con lo que resulta

$$P\,A = L\,U.$$

Es decir, salvo posibles permutaciones (recogidas todas en la matriz P), una matriz cualquiera puede expresarse como el producto de una matriz triangular inferior L con unos en la diagonal principal por una matriz triangular superior U. Es la llamada descomposición LU.

Conocida la descomposición LU de una matriz y suponiendo que no ha habido permutaciones, resolver el sistema $Ax = b$ es equivalente a resolver dos sistemas más simples (triangulares):

$$Ax = b \Leftrightarrow (LU)x = b \Leftrightarrow L\underbrace{(Ux)}_{=c} = b \Leftrightarrow Lc = b,\ Ux = c.$$

Para llevar a cabo la descomposición LU con MATLAB se puede, entre otras formas proceder del modo siguiente. Le "pegamos" a la matriz A la matriz identidad del mismo orden, si es de orden 3

`>> B=[A eye(3)]`

Realizamos las operaciones normalmente y quedarán registradas en la parte de la derecha, de modo que al final del proceso de eliminación tendremos, si no ha habido permutaciones: `[U` $\mathtt{L}^{-1}$`]`.

Ejercicios

1. Dada la matriz
$$A = \begin{pmatrix} 2 & 1 & 1 \\ 4 & 1 & 0 \\ -2 & 2 & 1 \end{pmatrix}$$
llévala a la forma triangular superior mediante operaciones elementales y calcula su descomposición LU.

2. Resolver empleando la factorización LU el sistema
$$\begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 4 \\ 2 & -1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$$

Nota: también existe una orden de MATLAB que permite calcular la descomposición LU de una matriz, es la orden `lu`.