1 Métodos directos

1.1 Conceptos básicos

En el anexo final se muestran las instrucciones básicas para operar con matrices (*transpose*, *inv*, *det*, …), obtener información de la matriz (*size*, *length*, …), crear tipos especiales de matrices (*zeros*, *ones*, *eye*, *rand*, …), obtener autovalores (*eig*, *poly*) y muchas más.

Comenzaremos viendo coherencia de operaciones con matrices, y para ello realizaremos funciones simples que sumen o multipliquen matrices. Además utilizaremos esto como repaso de los temas anteriores (condicionales y funciones). También se empleará información sobre el número de argumentos de entrada, a la hora de verificar lo coherencia de la operación.

Consideremos la siguiente función, que suma dos matrices o calcula la inversa

```
function R=opera(A,operacion,B)
%
%Realiza operaciones con una o dos matrices
%Entrada
%   A,B         matrices
%   operacion   suma('+'),resta('-'),multiplicación('*') o inversión(#')
%Salida
%   R           resultado
%
% Verificacion de los argumentos de entrada
switch nargin
    case 2 % La unica operacion admitida es la inversion
        if operacion~='#';error('Dos argumentos y la operacion no es
inversion');end
        [fA,cA]=size(A);
        if fA~=cA;error('La matriz a invertir no es cuadrada');end
        if fA~=rank(A);error('La matriz a invertir no es regular');end
        R=inv(A);
    case 3
        [fA,cA]=size(A);[fB,cB]=size(B);
        switch operacion
            case {'+','-'}
                if fA~=fB|cA~=cB;error('Las matrices no son congruentes');end
                if operacion='-';signo=-1;else;signo=1;end
                R=A+signo*B;
            otherwise
                error('Operación no valida');
        end
    otherwise
        error('Número de argumentos no válido');
end
```

Ejercicios recomendados:

1. Modificar el programa opera.m anterior para que también realice la multiplicación:

El algoritmo para la operación no incluida sería

Paso 1: Caso de operación multiplicación

Paso 2: Si las columnas de la matriz1 no coinciden con las filas de la matriz2,

Paso 3: enviarmensaje de error y finalizar

Paso 4: Calcular el producto de la matriz1 por la matriz2 y almacenarla en R

1.2 Resolución directa de sistemas lineales

La resolución de sistemas lineales es inmediata con MATLAB mediante la instrucción "\". Así, para resolver el sistema $Ax=b$ es suficiente poner `» x=A\b`. Si bien también se habría podido obtener el resultado mediante `» x=inv(A)*b`, esto se debe evitar pues realiza el cálculo de una inversa, que es un problema más complejo. Veámoslo con un ejemplo:

$$\begin{pmatrix} 2 & 4 & -1 & -2 \\ 4 & 0 & 2 & 1 \\ 1 & 3 & -2 & 0 \\ 3 & 2 & 0 & 5 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 10 \\ 7 \\ 3 \\ 2 \end{pmatrix}$$

```
» A=[2,2,-1,-2;4,0,2,1;1,3,-2,0;3,2,0,5];b=[10;7;3;2];x=A\b
» inv(A)*b % Resolución
» max(abs(eig(A))) % Radio espectral de la matriz A
```

Se pide:

2. Utilizar la instrucción \ para resolver el siguiente sistema

$$\begin{pmatrix} 112 & 84 & 48 & 28 \\ 84 & 81 & 77 & -7 \\ 48 & 77 & 146 & -83 \\ 28 & -7 & -83 & 85 \end{pmatrix} X = \begin{pmatrix} -204 \\ -186 \\ -193 \\ 38 \end{pmatrix}$$

A continuación, utilizando las instrucciones *tic* y *toc* se puede verificar que el tiempo de resolución de un sistema es proporcional a un tercio del cubo del orden.

```
orden=round(logspace(1,5,40));N=length(orden);t=zeros(N,1);
for n = 1:N
   A = rand(n,n);b = rand(n,1);
   tic; x = A\b;t(n) = toc;
end
plot(t)
```

Las instrucciones *tic* y *toc* funcionan como inicio y fin de un cronómetro, dando el tiempo en ejecutar un conjunto de instrucciones, y no el número de operaciones, pero se puede ver una correlación entre tiempos y operaciones.

Se pide:

3. Analizar el tiempo de resolución en el caso de matrices triangulares

1.3 Método de Gauss y pivoteo

MATLAB está diseñado para análisis matricial, por lo que las operaciones del método de Gauss se realizan fácilmente.

```
» Ab=[A,b] % Matriz ampliada
```

```
» i=1;j=2;m=Ab(j,i)/Ab(i,i); % calculo del pivote
» Ab(j,:)=Ab(j,:)-m*Ab(i,:); % anulamos la fila j-esima
```

El archivo adjunto *T31gaussps.m* permite la resolución de un sistema por el método de Gauss con pivote simple. Se puede hacer *doc T31gaussps* para ver las opciones de esa función.

Se pide:

4. Resolver el sistema del ejercicio 2 con la función *T31gaussps* y comprobar los valores de los diferentes argumentos de salida.
5. Modificar el archivo anterior para resolver por Gauss con pivote escalado.
6. Idem con pivote doble

1.4 Métodos de Factorización

MATLAB también tiene implementadas las funciones de factorización habituales, correspondientes a los métodos de Doolittle (LU) y Choleski (LDL'). La instrucción ***lu*** factoriza la matriz inicial como producto de dos matrices triangulares, la primera obtenida por permutación de una matriz triangular inferior de diagonal unidad y la otra triangular superior. Por su parte, la instrucción ***chol*** devuelve la factorización de cholesqui.

`» [L,U,P]=lu(A)`
`» [L,U]=lu(A)`
`» Y=lu(A)`
	Devuelve L (matriz triangular inferior de diagonal unidad), U (matriz triangular superior) y P (matriz unitaria de permutación) tales que L*U=P*A. Cuando solo obtienen dos matrices, L=P*L. Si hay un solo resultado, entonces Y=L+U-I

`» U=chol(A)` `» L=chol(A,'lower')`	Devuelve L'(U) o L respectivamente, de forma que L*L'=A

En ambos casos, el sistema se resuelve en dos pasos: Ly=b y Ux=y (respectivamente L'x=y). Se sugiere acudir a la ayuda para una descripción más detallada que la del anexo.

Se pide:

7. Factorizar la matriz del ejercicio 2) por ambos métodos y resolver los sistemas obtenidos.
8. Comparar los resultados de la instrucción */u* con los obtenidos mediante *T31gaussps*.

1.5 Condicionamiento

El condicionamiento de una matriz A se puede calcular en MATLAB aplicando directamente la definición $\kappa(A,p)=\|A\|_p\cdot\|A^{-1}\|_p$ o bien mediante una instrucción específica `cond(A,p)` donde *p* puede ser cualquiera de las siguientes normas: *1,2, inf*. Como ejemplo, vamos a comprobar la sensibilidad de la solución cuando se trabaja con matrices mal condicionadas. Para ello utilizaremos la matriz de Hilbert, y calculamos su condicionamiento:

```
orden=10;A=hilb(orden);b=A(:,orden);x=A\b
c1=cond(A,1),c2=cond(A,2),ci=cond(A,i)
```

Modificamos el sistema para tomar 5 decimales del vector *b* y resolvemos de nuevo

```
c=round(b*100000)/100000;y=A\c
```

Se observa claramente el efecto del condicionamiento. Comprobamos el error porcentual de la solución y su cota, utilizando la norma 1, mediante:

```
e1=100*norm(x-y,1)/norm(x,1) % error porcentual
co1=100*c1*norm(b-c,1)/norm(b,1) % cota del error
```

2 Anexo: Instrucciones necesarias

2.1 Operaciones con matrices

Función	Operación	Función	Operación
A'	Matriz transpuesta de A	transpose(A)	Matriz transpuesta de A
inv(A)	Matriz inversa de la matriz cuadrada A	rank(A)	Rango de la matriz A
det(A)	Determinante de la matriz cuadrada A	length(v)	Devuelve la longitud del vector v
trace(A)	Suma de los elementos de la diagonal de A	size(A)	Devuelve el orden (tamaño) de la matriz A

2.2 Generar tipos especiales de matrices

Función	Operación	Función	Operación
diag(A)	Extraer la diagonal de la matriz A	diag(v)	Matriz diagonal con los elementos de v
tril(A)	Parte triangular inferior de la matriz A	triu(A)	Parte triangular superior de la matriz A
eye(n)	Crea la matriz identidad de orden n-	eye(m,n)	Idem orden mxn.
zeros(n)	Crea la matriz nula de orden n	zeros(m,n)	Idem de orden mxn
ones(n)	Crea la matriz de unos de orden n	ones(m,n)	Idem de orden mxn
rand(n)	Crea una matriz aleatoria uniforme de orden n	rand(m,n)	Idem de orden mxn

2.3 Operaciones Lógicas con Matrices

any(V)	Devuelve 0 si todos los elementos del vector V son nulos, y devuelve 1 si algún elemento de V es no nulo
all(V)	Devuelve 1 si todos los elementos del vector V son no nulos , y devuelve 0 si algún elemento de V es nulo
find(V)	Devuelve los lugares (o índices) que ocupan los elemento no nulos del vector V

2.4 Resolución de Sistemas y factorización

Función	Operación	Función	Operación
X=A\B	Resuelve el sistema A*X=B	X=A/B	Resuelve el sistema X*A=B
lu(A)	Factorización LU	chol(A)	Factorización de Choleski

2.5 Autovalores y Autovectores

Función	Operación	Función	Operación
eig(A)	Autovalores de la matriz A	[V,D]=eig(A)	M matriz diagonal de autovalores D y matriz V de autovectores por columnas
poly(A)	Polinomio característico de A	poly(V)	Vector cuyas raíces son los elementos del vector V