MÉTODOS NUMÉRICOS
GUÍA DE LABORATORIO NRO. 5
RESOLUCIÓN DE SISTEMAS DE ECUACIONES LINEALES EN MATLAB

OBJETIVOS:

Utilizar comandos y funciones para resolver sistemas de ecuaciones lineales.
Resolver problemas mediante el uso del MATLAB.

MEDIOS Y MATERIALES EDUCATIVOS:

Guía de laboratorio, computadora, software de Matlab, tutoriales y manuales de Matlab, apuntes, Internet y flash memory.

INFORME:

Realizar un informe del laboratorio realizado, puede ser individual o de un máximo de dos estudiantes.

TAREA 1. OPERADORES PARA EL MANEJO DE MATRICES Y SISTEMAS DE ECUACIONES LINEALES

Los operadores de división matricial

Sea A una matriz cualquiera y sea B otra matriz con el mismo número de filas que A1. Entonces, la "solución" del "sistema" (en realidad un sistema lineal por cada columna de B)

A X = B

se calcula en MATLAB mediante el operador no estándar " \" denominado backward slash, (barra inversa en español):

```
X = A \ B
X = mldivide(A, B)     %Equivalente a lo anterior
```

Hay que pensar en él como el operador de "división matricial por la izquierda".
De forma similar, si C es una matriz con el mismo número de columnas

que A, entonces la solución del sistema
X A = C
se obtiene en MATLAB mediante el operador“/ ” (“división matricial por la derecha”):

```
X = C / A
X = mrdivide(C, A)          %Equivalente a lo anterior
```

Estos operadores se aplican incluso si A no es una matriz cuadrada. Lo que se obtiene de estas operaciones es, lógicamente, distinto según sea el caso.
De forma resumida, si A es una matriz cuadrada e y es un vector columna, c = A\y es la solución del sistema lineal Ac=y, obtenida por diferentes algoritmos, en función de las características de la matriz A (diagonal, triangular, simétrica, etc.). Si A es singular o mal condicionada, se obtendrá un mensaje de alerta (warning). Si A es una matriz rectangular, A\y devuelve una solución de mínimos cuadrados del sistema Ac=y.
Para una descripción completa del funcionamiento de estos operadores, así como de la elección del algoritmo aplicado, véase la documentación de MATLAB correspondiente, tecleando en la ventana de comandos:

```
doc mldivide
```

Ejemplo.

Calcular la solución del sistema (compatible determinado)

$$
\begin{aligned}
2x_1 + x_2 - x_3 &= -1 \\
2x_1 - x_2 + 3x_3 &= -2 \\
3x1 - 2x2 &= 1
\end{aligned}
$$

Solución.

```
>> A = [2, 1, -1; 2, -1, 3; 3, -2, 0];
>> b = [-1; -2; 1];
>> x = A\b x =

-0.3636
-1.0455
-0.7727
```

TAREA 2. FUNCIONES PARA EL MANEJO DE MATRICES Y SISTEMAS DE ECUACIONES LINEALES

Instrucción	Acción
`A = [a11,a12,...,a1n ; a21,a22,...,a2n ; ...; am1,am2,...,amn ]`	crea una matriz $m \times n$ y la nombra como A. **Nota**. Las comas se pueden substituir por espacios. Los puntos y comas separan las filas de A.
`x + y`	calcula la suma vectorial $\mathbf{x} + \mathbf{y}$.
`c*x`	calcula el producto escalar $c\mathbf{x}$.
`norm (x)`	calcula la norma de $\mathbf{x}$.
`dot (x, y)`	calcula el producto punto entre $\mathbf{x}$ e $\mathbf{y}$.
`cross (x, y)`	calcula el producto cruz entre $\mathbf{x}$ y $\mathbf{y}$.
`eye(n)`	produce la matriz identidad de orden n.
`ones(m, n)`	produce la matriz de 1's de tamaño $m \times n$.
`zeros(m, n)`	produce la matriz de 0's de tamaño $m \times n$.
`rand(m, n)`	produce una matriz aleatoria $m \times n$ con entradas en $(0,1)$.
`diag ( [a, b, c ] )`	construye una matriz diagonal con entradas a, b y c.
`A ( i, :)`	genera la fila i-ésima de A.
`A ( :, j )`	genera la columna j-ésima de A.
`A ( :, [ m, n, k ] )`	genera la columnas m, n y k de A.
`triu ( A )`	genera la parte triangular superior de A.
`tril ( A )`	genera la parte triangular inferior de A.
`diag ( A )`	genera la diagonal de A.
`[ A b ]`	genera la *matriz aumentada* $[A \mid b]$.
`rref ( A )`	reduce A mediante la eliminación *Gauss-Jordan*.
`A + B`	calcula la suma matricial $A + B$.
`c*A`	calcula el producto escalar cA.
`A*B`	calcula el producto matricial AB (si está definido)
`A'`	genera la transpuesta de A.
`A^k`	calcula A^k.
`inv( A )`	calcula la inversa de A.
`det( A )`	calcula el determinante de A.
`poly( A )`	genera los coeficientes de $p_A(\lambda) = a_n\lambda^n + \cdots + a_1\lambda + a_0$ en el orden: $a_n, \ldots, a_1, a_0$.
`eig( A )`	calcula los valores propios de A.
`[ P D ] = eig( A )`	calcula vectores y valores propios de A. **Nota**. Si A es simétrica entonces P es ortogonal.
`null( A, 'r' )`	genera una base *racional* (no ortogonal) para nul (A).
`null( A )`	genera una base *ortogonal* para nul (A).
`[ Q R ] = qr( A )`	produce la factorización QR de A.
`[ L U ] = lu( A )`	produce (de existir) la factorización LU de A.
`[ L U P ] = lu( A )`	produce la factorización $PA = LU$ de A, si A no factoriza LU.
`format rat`	Formato de números racional.
`format long, format short`	Formato de números con 15 y 5 dígitos decimales, respectivamente.

Probar las funciones de la tabla.

TAREA 3. PROBLEMAS

a) Hacer un algoritmo para una función que descomponga una matriz A en dos matrices LU, programar en Matlab y comparar con la función lu.
b) Hacer un programa que evalué un sistema de ecuaciones lineales mediante la descomposición LU.
c) Hacer un programa que calcule el condicional de una matriz.
d) Hacer un programa que descomponga una matriz en dos matrices QR y comparar con la función qr de Matlab.
e) Resolver los siguientes sistemas de ecuaciones mediante las funciones Gauss, Gauss-Jordan y descomposición LU:

i. $3x_1 - 0.1x_2 - 0.2x_3 = 7.85$
$0.1x_1 + 7x_2 - 0.3x_3 = -19.3$
$0.3x_1 - 0.2x_2 + 10x_3 = 71.4$

ii. $10x_1 + 2x_2 - x_3 = 27$
$-3x_1 - 6x_2 + 2x_3 = -61.5$
$x_1 + x_2 - 5x_3 = -21.5$