Diplomado : Herramientas de Programación para Ciencias e Ingeniería

Módulo: MATLAB (Clase 3)

Docente: Juan Sebastián Salcedo Gallo

Universidad Nacional de Colombia Sede Manizales

Contenido

- Matrices
- Manipulación de matrices
- Solución de sistemas de ecuaciones lineales (A mano y usando MATLAB)
- Introducción Método Simbólico MATLAB

Matrices

En matemática, una matriz es un arreglo bidimensional de números.

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

Las matrices se utilizan para múltiples aplicaciones y sirven, en particular, para representar los coeficientes de los sistemas de ecuaciones lineales, entre muchas aplicaciones más (Transformaciones).

Matrices

Para efectos de este curso, asumirémos que una matriz es un arreglo bidimensional de los coeficientes de las variables independientes de un sistema de ecuaciones.

$$\begin{cases} 2a + 3b = 5 \\ 5a + 2b = 2 \end{cases}$$

El número de columnas de una matriz se representa con la letra m, y el número de filas, con la letra n. Existe un caso especial de matriz donde m == n, estas matrices se conocen como matrices cuadradas.

Vectores

Para crear un vector en MATLAB introducimos los valores deseados separados por espacios (o comas) todo ello entre corchetes. Si lo que queremos es crear una matriz lo hacemos de forma análoga pero separando las filas con (;)

```
>> x = [0.0:0.1:1.0]
x =

        0      1/10      1/5      3/10      2/5      1/2      3/5      7/10      4/5      9/10      1
```

Generalmente usaremos letras mayúsculas para nombrar matrices y minúsculas para vectores y escalares **(Por convención).**

Ejemplos Vectores y Matrices

```
>> x = [5 7 -2 4 -6]    % es un vector, los elementos los separamos con espacios
x =
    5    7   -2    4   -6

>> y = [2,1,3,7]        % es otro vector, los elementos los separamos con comas
y =
    2    1    3    7

>> z = [0 1 2,3 4,5]    % es otro vector, da igual separar los elementos por comas o espacios
z =
    0    1    2    3    4    5
```

Ejemplos Vectores y Matrices

```
>> A = [1 2 3; 4 5 6]  % es una matriz con 2 filas y 3 columnas
A =
    1   2   3
    4   5   6
```

```
>> A = [1 2 3; 4 5 6; 7 8 9]
A =

        1       2       3
        4       5       6
        7       8       9
```

Dada una matriz(m,n), m simboliza el número de columnas y n es número de filas.

Direccionamiento de Elementos de Vectores

Para acceder a los elementos individuales de un vector lo haremos utilizando sub-índices, así x(n) sería el n-ésimo elemento del vector x. Si queremos acceder al último podemos indicarlo usando **end como sub-índice.**

```
>> x = [5 7 -2 4 -6];
>> x (2)          % segundo elemento del vector x
ans =
    7

>> x (end)        % último elemento del vector x
ans =
   -6
```

Direccionamiento de Elementos de Vectores

Podemos acceder a bloques de elementos a la vez, usando la notación de dos puntos (:), así x(m:n) nos dá todos los elementos desde el punto m-ésimo hasta el n-ésimo término del vector x.

```
>> x (2:4)      % devuelve desde el segundo al cuarto elemento del vector x
ans =
    7   -2    4
```

Podemos si introducimos un número entre m y n, este significará el incremento o decremento (si es negativo) en los índices.

```
>> x (1:2:5)          % devuelve el primero, tercero y quinto elemento del vector x
ans =
    5   -2   -6
```

Direccionamiento de Elementos de Matrices

En una matriz funciona de forma similar. Ahora debemos tener en cuenta que es un arreglo bidimensional.

Para escribir toda una fila usaremos (:) en el índice que corresponde a las filas.

```
>> A (2,:)                % escribe la segunda fila de la matriz
ans =
    4    5    6
```

Direccionamiento de Elementos de Matrices

De forma similar, si queremos imprimir en consola todos los elementos de determinada columna, usaremos (:) nuevamente, esta vez en el espacio de las columnas.

```
>> A (:,2)                    % escribe la segunda columna de la matriz
ans =
    2
    5
```

Igual que en los vectores, podemos hacer que la consola nos muestre ciertas porciones de filas o columnas realizando recortes en los arreglos, tal y como se hacía para vectores.

Direccionamiento de Elementos en Matrices

MATLAB tiene además una forma de identificar cada elemento de una matriz, usando solo un valor, y no dos, como vimos anteriormente.

A(1)	A(3)	A(5)
A(2)	A(4)	A(6)

Como la matriz que teníamos era:

```
A =
     1   2   3
     4   5   6
```

```
>> A (5)        % accede al elemento 5 de la matriz, es decir, igual que si escribiéramos A (1,3)
ans =
    3
```

Construcción Abreviada de Algunos Vectores

Sin tener que definir un vector escribiendo cada uno de sus elementos, podemos usar las siguientes sentencias.

- (a:b) crea un vector que comienza en a, termina en b, aumentando de 1 en 1.
- (a:c:b) crea un vector que comienza en a, termina en b, con incrementos de c.
- linspace(a,b,c) genera un vector linealmente espaciado entre los valores a y b, con c elementos.
- linspace(a, b) genera un vector linealmente espaciado entre a y b con 100 elementos.

Construcción Abreviada de Algunos Vectores

- logspace(a,b,c) genera un vector logarítmicamente espaciado entre los valores 10^a y 10^b con c elementos.
- logspace(a,b) genera un vector logarítmicamente espaciado entre los valores 10^a y 10^b con 50 elementos.

Ejemplos Construcción de Algunos Vectores

```
>> (1:7)          % crea un vector que comienza en 1, aumenta de 1 en 1 y acaba en 7
ans =
    1    2    3    4    5    6    7

>> (1:4:10)     % comenzando en 1, aumenta de 4 en 4 hasta el 10 y por eso acaba en 9
ans =
    1    5    9

>> (50:-7:1)    % crea un vector que comenzando en 50, disminuye de 7 en 7 hasta el 1
ans =
   50   43   36   29   22   15    8    1

>> linspace (2,6,3)     % genera un vector desde el 2 al 6 con 3 elementos equidistantes
ans =
    2    4    6
```

Ejemplo

Cree un vector con los números del 1 al 10 y sume sus elementos.

```
>> x = [1:10]
x =

         1          2          3          4          5          6          7          8          9         10

>> sum_elements = sum(x)
sum_elements = 55
```

Ejercicio

Cree un vector con los números del 1 al 100 y sume sus elementos, usando un ciclo for y usando el comando sum( ) de MATLAB.

Solución

```
>> suma = 0;
>> for i=1:10;
       suma = suma + x(i);
   end;
>> suma
suma = 55
>>
```

Construcción de Algunas Matrices

Al igual que con vectores, existen sentencias que ayudan a crear más rápidamente algunas matrices.

- zeros(n) crea una matriz cuadrada nxn de ceros.
- zeros(m,n) crea una matriz mxn de ceros.
- ones(n) crea una matriz cuadrada nxn de unos.
- ones(m,n) crea una matriz mxn de unos.
- rand(n) crea una matriz cuadrada nxn de números aleatorios con distribución uniforme (0,1).
- rand(m,n) crea una matriz mxn de números aleatorios con distribución uniforme (0, 1).

Construcción de Algunas Matrices

- randn(n) crea una matriz cuadrada nxn elementos con distribución normal (0,1).
- randn(m,n) crea una matriz mxn de números aleatorios con distribución normal (0,1).
- eye(n) crea una matriz cuadrada nxn de unos en la diagonal y ceros al resto.
- magic(n) crea una matriz cuadrada nxn de enteros de modo que sumen los mismo las filas, las columnas y la diagonal (“sudoku”).

Ejemplos Creación de Matrices

```
>> zeros (3)          % matriz cuadrada 3 x 3 de ceros
ans =
    0     0     0
    0     0     0
    0     0     0
```

```
>> eye (2)            % matriz  identidad o unidad
ans =
    1     0
    0     1
```

```
>> magic (4)          % matriz mágica 4 x 4
ans =
   16     2     3    13
    5    11    10     8
    9     7     6    12
    4    14    15     1
```

Operaciones Básicas con Matrices

Símbolo	Expresión	Operación
+	**A + B**	Suma de Matrices
-	**A - B**	Resta de Matrices
*	**A * B**	Multiplicación de Matrices
.*	**A.*B**	Multiplicación Término a Término de Matrices
/	**A/B**	División de Matrices
./	**A./B**	División Término a Término
^	**A^n**	Potenciación (n entero)
.^	**A.^B**	Potenciación Término a Término

Operaciones Básicas con Matrices

Símbolo	Expresión	Operación
‘	**A’**	Trasposición Compleja Conjugada (?)
.’	**A.’**	Trasposición de Matrices.

Funciones Para el Análisis de Matrices

Función	**¿Qué hace?**
cond(A)	Mide qué tan sensible de inversión es una matriz. Sensible si cond(A) >> 1.
det(A)	Determinante de una Matriz
diag(v)	Crea una matríz diagonal con el vector v sobre la diagonal
diag(A)	Extrae la diagonal de la matriz A.
eig(A)	Valores Propios
inv(A)	**Matriz inversa de A**
length(A)	Máxima dimensión de A

Funciones para el Análisis de Matrices

Función	**¿Qué hace?**
norm(A)	Norma de A
rref(A)	Reducción mediante la eliminación de Gauss de una matriz
pinv(A)	pseudo-Inversa de A
rank(A)	Rango de A. Es el número de filas (o columnas) linealmente independientes
size(A)	Dimensiones de A
trace(A)	Suma de los elementos de la diagonal principal de una matriz

Introducción Symbolic MATLAB

- Considere el siguiente sistema de ecuaciones lineales:

$$
\begin{aligned}
2x + y + z &= 2 \\
-x + y - z &= 3 \\
x + 2y + 3z &= -10
\end{aligned}
$$

Declare el sistema de ecuaciones de la siguiente forma:

```
syms x y z
eqn1 = 2*x + y + z == 2;
eqn2 = -x + y - z == 3;
eqn3 = x + 2*y + 3*z == -10;
```

Introducción a Symbolic MATLAB

Use el comando ***equationsToMatrix*** para convertir las ecuaciones en la forma Ax = B. El segundo término de ***equationsToMatrix*** especifica las variables independientes en las ecuaciones.

```
[A,B] = equationsToMatrix([eqn1, eqn2, eqn3], [x, y, z])
```

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

B =
   2
   3
 -10
```

Introducción a Symbolic MATLAB

Use linsolve(A,B) para solucionar el sistema de ecuaciones lineales y encontrar los valores de las variables independientes.

```
X = linsolve(A,B)
```

```
X =
   3      x
   1      y
  -5      z
```

Ejercicios

- Construya las tablas de la verdad estudiadas en la clase pasada, usando Matrices. Para esto es necesario aplicar los operadores lógicos y manipulación de matrices, puesto que es necesario agregar la columna con los resultados de aplicar los operadores lógicos.

Ejercicio

Resolver a mano usando el método de la adjunta y después usando MATLAB el siguiente sistema de ecuaciones lineales 2x2.

$$\begin{cases} 2a + 3b = 5 \\ 5a + 2b = 2 \end{cases}$$

R/. a = -0.3636

b = 1.9090

Ejercicio

Resolver el siguiente sistema de ecuaciones Lineales 3x3, a mano y usando MATLAB.

$$\begin{cases} 4a + 2b + 7c = 1 \\ 7a + 5b + 3c = 2 \\ 11a + 9b + 4c = 3 \end{cases}$$

R/. a = 8/9

b = -3/19

c = -1/19

Sección de Ejercicios

3. El comando "magic(N)" genera una matriz de dimensiones $N \times N$ cuyos elementos forman un "*sudoku*" resuelto. Cree una matriz de esta forma que se llame A con N = 10 y otra matriz que se llame B con N = 20.

Luego construya una nueva matriz C de dimensiones 10 x 15, donde las primeras 3 columnas sean las columnas 4, 5, 6 de la matriz A. Las otras 12 columnas de C son dadas por las columnas de B desde la 1 a la 12 con sólo las últimas 10 filas.

Ejecute por último el comando "sum (sum (C))", esto suma todos los elementos de la matriz C. El resultado de esta suma debe ser 25575.

Solución

```
>> D
D =  25575
>> C = [A(:, 4:6), B(11:end,1:12)]
C =

     8    15    67   201   199   198   204   205   195   194   208   209   191   190   212
    14    16    73   180   222   223   177   176   226   227   173   172   230   231   169
    20    22    54   160   242   243   157   156   246   247   153   152   250   251   149
    21     3    60   261   139   138   264   265   135   134   268   269   131   130   272
     2     9    61   281   119   118   284   285   115   114   288   289   111   110   292
    83    90    42   100   302   303    97    96   306   307    93    92   310   311    89
    89    91    48    80   322   323    77    76   326   327    73    72   330   331    69
    95    97    29   341    59    58   344   345    55    54   348   349    51    50   352
    96    78    35   361    39    38   364   365    35    34   368   369    31    30   372
    77    84    36    20   382   383    17    16   386   387    13    12   390   391     9

>> sum(sum(C))
ans =  25575
>>
```