UNIVERSIDAD TECNOLÓGICA DE PEREIRA
Programa de tecnología Eléctrica

Introducción a Matlab
Taller No. 1

Programación, TE243
Primer semestre de 2014
Ing: José Norbey Sánchez F.
Grupo:

1.1 MATLAB definición

MATLAB es un lenguaje de alto desempeño diseñado para realizar cálculos técnicos. MATLAB integra el cálculo, la visualización y la programación en un ambiente fácil de utilizar donde los problemas y las soluciones se expresan en una notación matemática. MATLAB es un sistema interactivo cuyo elemento básico de datos es el arreglo que no requiere de dimensionamiento previo.
El nombre abreviado de MATLAB es "MATrix LABoratory", es un programa para realizar cálculos numéricos con ***vectores*** y ***matrices***. Como caso particular puede también trabajar con números escalares, tanto reales como complejos. Una de las capacidades más atractivas es la de realizar una amplia variedad de ***gráficos*** en dos y tres dimensiones. MATLAB tiene también un lenguaje de programación propio (lenguaje M).

MATLAB se utiliza ampliamente en:

- Cálculos numéricos
- Desarrollo de algoritmos
- Modelado, simulación y prueba de prototipos
- Análisis de datos, exploración y visualización
- Graficación de datos con fines científicos o de ingeniería
- Desarrollo de aplicaciones que requieran de una interfaz gráfica de usuario (*GUI*, *Graphical User Interface*).

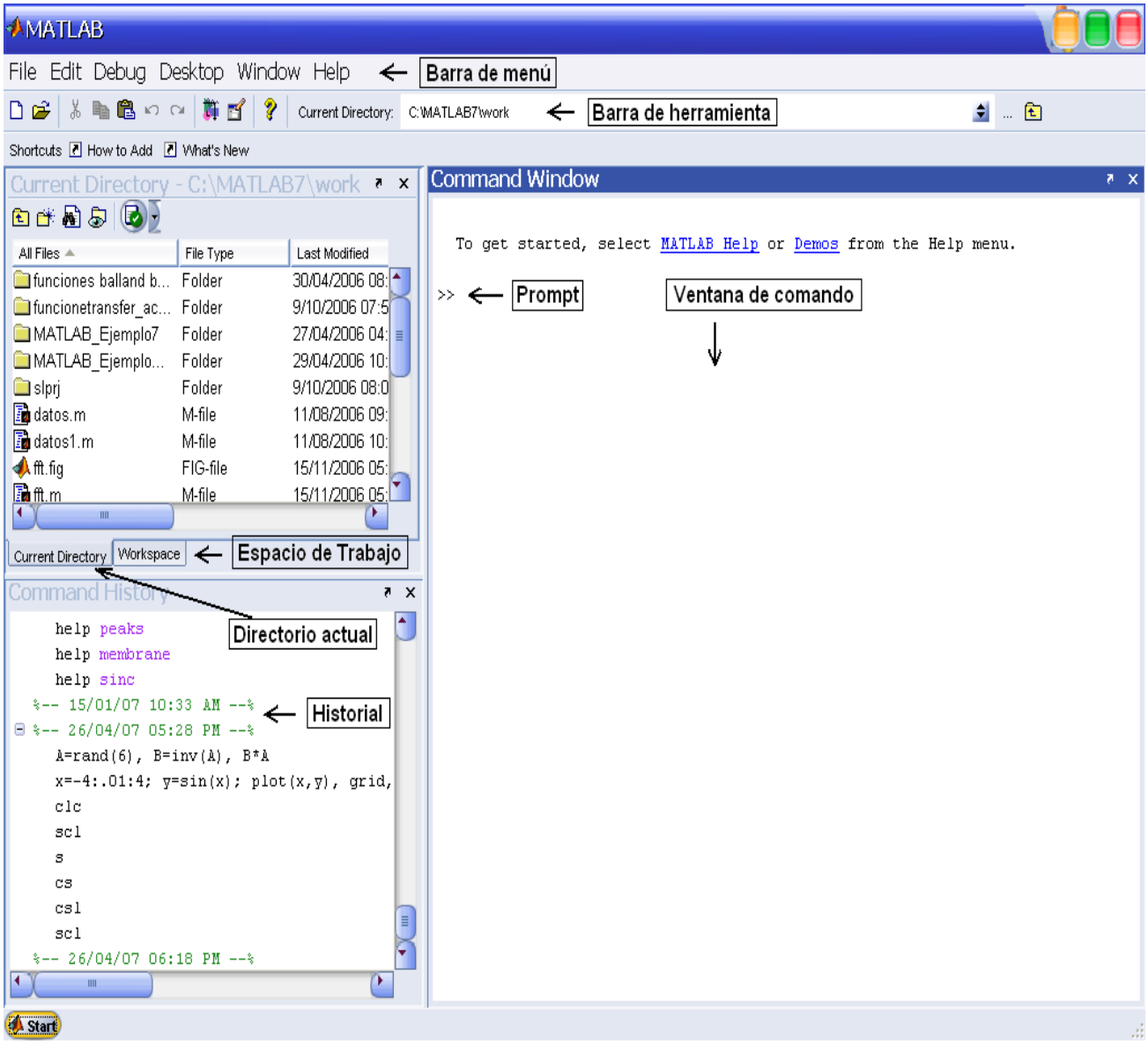
En el ámbito académico y de investigación, es la herramienta estándar para los cursos introductorios y avanzados de matemáticas, ingeniería e investigación. En la industria MATLAB es la herramienta usada para el análisis, investigación y desarrollo de nuevos productos tecnológicos. La ventaja principal de MATLAB es el uso de familias de comandos de áreas específicas llamadas *toolboxes*.

Los *toolboxes* son grupos de comandos de MATLAB (archivos M) que extienden el ambiente de MATLAB para resolver problemas de áreas específicas de la ciencia e ingeniería. Por ejemplo, existen *toolboxes* para las áreas de Procesamiento Digital de Señales, Sistemas de Control, Redes Neuronales, Lógica Difusa, Wavelets, etc.

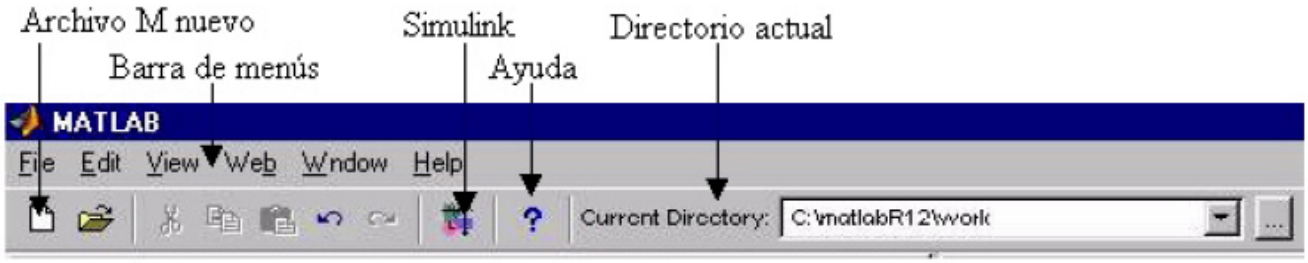
1.2 Ventanas principales al abrir Matlab.

En el siguiente gráfico, se muestran las principales ventanas al cargar el programa de Matlab:

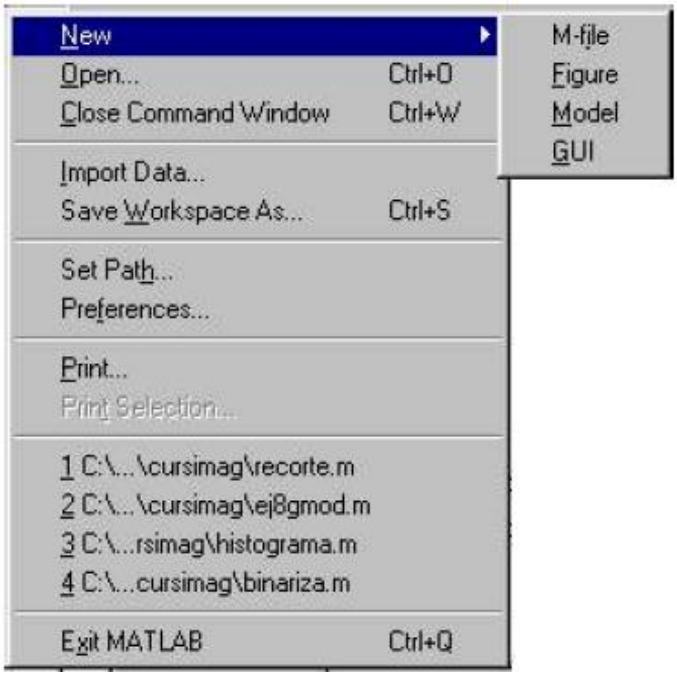
Comand Window: es la ventana de comandos o ventana principal y es en la que se trabaja y en la que se introducen todos los comandos. En esta ventana aparece el ***prompt*** característico de MATLAB (»), y significa que el programa está listo para recibir instrucciones.
Workspace: es el espacio de trabajo, muestra información sobre las variables que se han creado
Comand History: recoge el historial de todos los comandos introducidos
Current Directory: es el directorio actual y contiene los ficheros *.m de la carpeta.



En la barra de herramientas, se encuentra simulink, archivo M nuevo, directorio actual, ayuda etc.



A través de la barra de menús, se accede a las operaciones que no están disponibles en la barra de herramientas. El menú principal es *File*, en el se encuentra la gran mayoría de las operaciones no presentes en la barra de herramientas. En la siguiente figura se muestra por ejemplo el menú *File.*



1.3 Expresiones en MATLAB

MATLAB como cualquier lenguaje de programación proporciona expresiones matemáticas, pero a diferencia de la mayoría de ellos, las expresiones matemáticas que maneja involucran matrices completas, ver ejemplos. Las expresiones se dividen en:

clock → Devuelve fecha y hora del sistema
fix(clock) → Fecha y hora del sistema con enteros
k = 2^10 → 2 elevado a la 10
x = rand → un número aleatorio entre 0 <= x < 1
X = fix(rand * 10) → número entero aleatorio entre 0 <= X < 10
r = 2 ; sup = pi * r ^ 2 → Calcula la superficie de un círculo de radio 2

1.3.1 Números

MATLAB utiliza una notación decimal convencional con punto decimal opcional y el signo *menos* para indicar números negativos. La notación científica utiliza la letra ***e*** para especificar el factor de escala en potencias de 10. Los números imaginarios utilizan ya sea la i o la j como sufijo. A continuación se presentan varios números permitidos.

3	–99	0.0001
9.6397238	1.60210e–20	6.02252e23
1i	–3.14159j	3e5i

Todos los números se almacenan internamente, usando el formato ***long*** especificado por el estandar en punto flotante IEEE. Los nnúmeros en punto flotante tienen una precisión finita aproximadamente de 16 dígitos decimales y un rango finito aproximadamente de 10^{-308} a 10^{+308}

1.3.2 Variables

MATLAB, a diferencia de la mayoría de los lenguajes de programación no requiere de ningún tipo de declaraciones de tipo de datos (entero, punto flotante, complejos, etc.) ni de dimensionamiento. Cuando MATLAB encuentra una nueva variable, automáticamente crea la variable y reserva la cantidad de localidades de memoria necesarias. Si la variable ya existía dentro del espacio de trabajo actual, simplemente cambia el contenido, si se requiere, o de ser necesario agrega más localidades de memoria a la variable para contener más datos, ejemplo.

Num = 25

Esta variable crea una matriz de 1 x 1 llamada Num y almacena el valor de 25. MATLAB es *case sensitive*, es decir distingue entre mayúsculas y minúsculas; es decir A y a no son la misma variable.

1.3.3 Operadores

Las expresiones utilizan los operadores aritméticos comunes. Los operadores aritméticos son los mismos que en cualquier lenguaje de programación y se sigue un orden de evaluación similar al que se utiliza en los demás lenguajes de programación. En la siguiente tabla, se muestran los operadores aritméticos más comunes en MATLAB, y en la siguiente los operadores relacionales y lógicos.

Operación matemática	**Operador**
Suma	+
Resta	-
Multiplicación	*
División	/
Potencia	^
Transpuesta compleja conjugada	‘
Especifica el orden de evaluación	( )

Operadores relacionales	Operadores lógicos
<menor que	& and
>mayor que	| or
<= menor o igual que	~ negación lógica
>= mayor o igual que	
== igual que	
~= distinto que	

Si una comparación se cumple el resultado es 1 (true), mientras que si no se cumple es 0 (false).

1.3.4 Funciones

MATLAB proporciona un gran número de funciones matemáticas simples y avanzadas. La gran mayoría de estas funciones acepta argumentos complejos. Las funciones más comunes, como ***sqrt*** y ***sin*** son parte del núcleo de MATLAB y están programadas en bajo nivel para hacerlas más eficientes y no es posible acceder a su código. Otras funciones están programadas en archivos con extensión M y su código está disponible para revisiones o modificaciones. Muchas funciones especiales proporcionan o requieren valores de constantes útiles. MATLAB incorpora constantes matemáticas y cierta simbología, la cual se muestra en la siguiente tabla.

Constante	Significado
pi	3.14159265…
i	Unidad imaginaria, $\sqrt{-1}$
j	Igual que i
eps	Precisión relativa en punto flotante, 2^{-52}
realmin	Número más pequeño representable en punto flotante, 2^{-1022}
realmax	Número más grande representable en punto flotante, (2-e) 2^{1023}
Inf	Infinito
NaN	No es un número

1.3.5 Expresiones

A continuación se presentan algunos ejemplos de programación de expresiones literales que se transcriben en Matlab.

Expresión: $\rho = \frac{(1+\sqrt{5})}{2}$

Es necesario teclear:
rho = (1+sqrt(5))/2
rho =
1.6180

Expression: $|3+4i|$

Es necesario teclear:
a = abs(3+4i)
a =
5

1.4 Generación de vectores y matrices

1.4.1 Escalares y vectores

La mejor manera de familiarizarse con MATLAB consiste en aprender a manejar las matrices. En MATLAB, una matriz es un arreglo rectangular de números. Las matrices de 1x1 se conocen como escalares, y las matrices con una sola columna o renglón se conocen como vectores. Estas matrices y/o vectores pueden contener datos tanto numéricos como no numéricos. Los datos pueden introducirse en MATLAB de diferentes maneras:

- Como una lista explícita de elementos
- Cargando los datos de un archivo externo
- Generados por otras funciones
- Creados por archivos M creados por el usuario.

Para asignar un escalar a una variable, se introducen los datos de la siguiente forma. Dar a la siguiente variable "dato", el valor de 2001.

```
dato = 2001
```

y MATLAB responde de la siguiente manera:

```
dato =
2001
```

Para introducir una matriz o un vector, se siguen los siguientes convencionalismos

- Separar los elementos de una columna usando espacios en blanco
- Usar punto y coma (;) para indicar el fin de una columna o el fin del vector
- Encerrar la lista de elementos con paréntesis rectangulares [ ]

Ejemplo: Se desea introducir el siguiente vector y = {0.2944 -1.3362 0.7143 1.6236 -0.6918 0.8580}, se teclea en la línea de comando así:

```
Y = [0.2944 -1.3362 0.7143 1.6236 -0.6918 0.8580]
```

y MATLAB despliega el vector así.

```
0.2944    -1.3362    0.7143    1.6236    -0.6918    0.8580
```

Ejemplo: se define un vector columna de 0 a 9, para obtener su cuadrado y elevar a la ***n*** en base 2

```
n = (0:9)';
pows = [n  n.^2  2.^n]
```

pows =

```
 0   0   1
 1   1   2
 2   4   4
 3   9   8
 4  16  16
 5  25  32
 6  36  64
 7  49 128
 8  64 256
 9  81 512
```

Ejemplo: crear un vector de 10 valores aleatorios entre 0 y 9 con distribución uniforme

```
N = fix(10*rand(1,10))
```

```
N =
9     2     6     4     8     7     4     0     8     4
```

Nota: el comando *fix* redondea el valor al entero inferior inmediato. El primer número es la fila y el segundo representa los datos del vector, observe que no va transpuesto.

1.4.2 Matrices

Muchos de los comandos de MATLAB permiten generar vectores o matrices de datos de algunas características. Por ejemplo, secuencias aleatorias con cierta distribución, escalones unitarios, matrices o vectores cero, etc. En la siguiente tabla, se muestran algunos comandos para generar algunas matrices especiales.

Comando	**Función**
Zeros	Todos los elementos de la matriz son ceros
Ones	Todos los elementos de la matriz son unos
Rand	Genera una matriz con de elementos con distribución uniforme
Randn	Genera una matriz con elementos con distribución normal
Para definir matrices	
[ ]	constructor
,	separador de columnas o en lugar (,) puede utilizarse un espacio
;	separador de filas o en lugar de (;) puede utilizarse un retorno de carro

Ejemplo: Generar una matriz de ceros de 2 renglones por 4 columnas

```
Z = zeros(2,4)
Z =
  0     0     0     0
  0     0     0     0
```

Ejemplo: Generar una matriz de 3 columnas por 3 renglones con todos sus elementos igual a 5.

```
F = 5*ones(3,3)
F =
  5     5     5
  5     5     5
  5     5     5
```

Ejemplo: se desea introducir la siguiente matriz literal

$$A = \begin{bmatrix} 16 & 3 & 2 & 13 \\ 5 & 10 & 11 & 8 \\ 9 & 7 & 6 & 12 \\ 4 & 15 & 14 & 1 \end{bmatrix}$$

Se Teclea en la línea de comando
A = [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
MATLAB responde de la siguiente manera:

```
A =
   16    3    2   13
    5   10   11    8
    9    6    7   12
    4   15   14    1
```

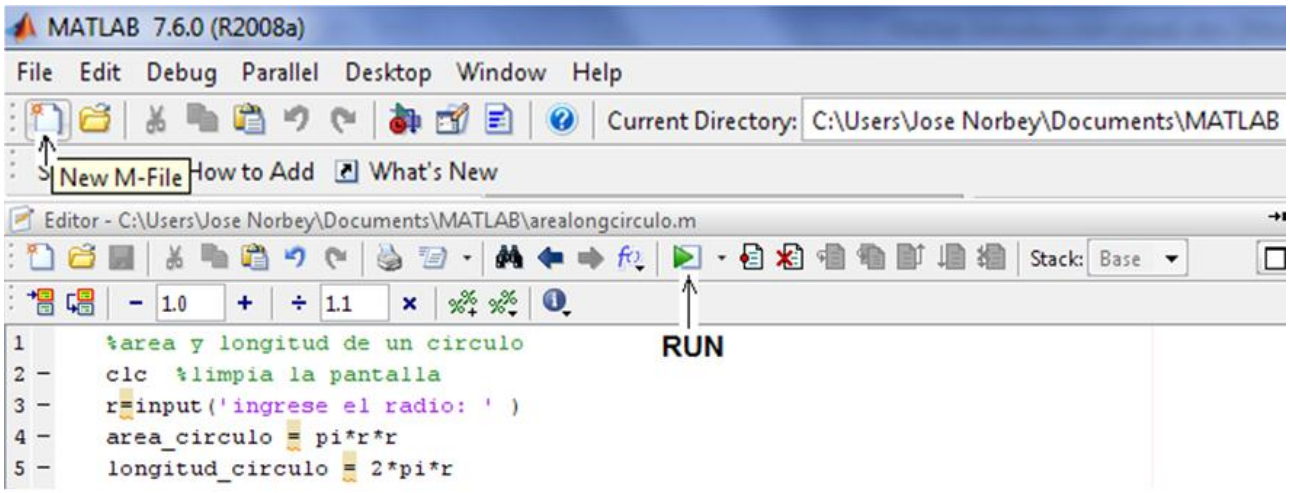
Una vez que se he introducido la matriz en la línea de comando, esta permanece en el espacio de trabajo, y para invocarla solo es necesario teclear A para referirse a la matriz.

Ejemplo: generar una matriz de 4 x 4 con números aleatorios y con distribución normal.

```
R = randn(4,4)
R =
-0.4326      -1.1465      0.3273      -0.5883
-1.6656       1.1909      0.1746       2.1832
 0.1253       1.1892     -0.1867      -0.1364
 0.2877      -0.0376      0.7258       0.1139
```

1.5 Crear archivos con extensión .m

Para crear un archivo con extensión m, se elige la opción New M-File como se observa en el gráfico; luego se teclea el código por ejemplo: cálculo del área y longitud de un círculo y se guarda con un nombre (arealongcirculo.m) y finalmente se ejecuta con RUN.




```
ingrese el radio: 1
r =
    1
area_circulo =
   3.1416
longitud_circulo =
   6.2832
```

1.6 Concatenación

La concatenación es el proceso de unir pequeñas matrices o vectores para crear otra matriz o vector de mayor tamaño. Este proceso ya se ha usado en algunos de los ejemplo anteriores al concatenar elementos individuales para formar vectores o matrices mediante el operador concatenación, formado por el par de corchetes, [ ].

Ejemplo: formar dos vectores de 4 elementos y concatenarlos para formar un solo vector de 8 elementos.

```
a=[1 2 3 4];
b=[5 6 7 8];
c=[a b]
c =
1        2        3        4        5        6        7        8
```

Ejemplo: Formar una matriz de 2 renglones por 4 columnas con los vectores a y b del ejemplo anterior.

```
d=[a; b]
d =
1        2        3        4
5        6        7        8
```

1.7 Operador elemento a elemento

En MATLAB existe también la posibilidad de aplicar ***elemento a elemento*** los operadores matriciales **(.*, .^, .\ y ./)** Para ello basta precederlos por un punto (**.**). Por ejemplo:

```
[1 2 3 4]^2
??? Error using ==> ^
Matrix must be square.
```

```
[1 2 3 4].^2
ans =
1        4        9        16
```

```
[1 2 3 4]*[1 -1 1 -1]
??? Error using ==> *
```

```
[1 2 3 4].*[1 -1 1 -1]
ans =
1        -2        3        -4
```

1.8 Cambio de base de un número binario

Sea el siguiente número binario de 5 bits: 10111, pasar a las siguientes bases:

Octal: (base 8)
Hex: (base 16)
Dec: (base 10)

bin2dec('10111')	23 (lo primero es pasar de binario a decimal)
dec2bin(23) 10111	10111
dec2base(23, 2)	10111 (base 2, es binario)
dec2base(23, 8)	27 (base 8, es octal)
dec2base(23, 16)	17 (base 16, es hexadecimal)
dec2hex(23) 17	

TALLER No.1 PARA RESOLVER EN MATLAB

Realizar los siguientes ejercicios:

1- Crear dos vectores de 10 datos c/u y guardarlos, luego llamarlos y realizar la suma vectorial, resta y multiplicación y división punto a punto.

2- Crear dos matrices de 8 x 5 y guardarlas, luego llamarlas y realizar la suma vectorial, resta y multiplicación y división punto a punto.

3- Hacer una tabla de multiplicar del 1 al 10. Calcular la inversa **inv(A)**, la transpuesta **A'**

4- Crear un archivo con extensión .m, con los siguientes datos x, y, z , donde cada variable debe entrar por teclado y calcule: $W = \sqrt{seno(x) \cdot \tan(y) + z^2}$

5- De la misma manera que el anterior ejercicio, convertir de binario a decimal un número binario de 8 bits.