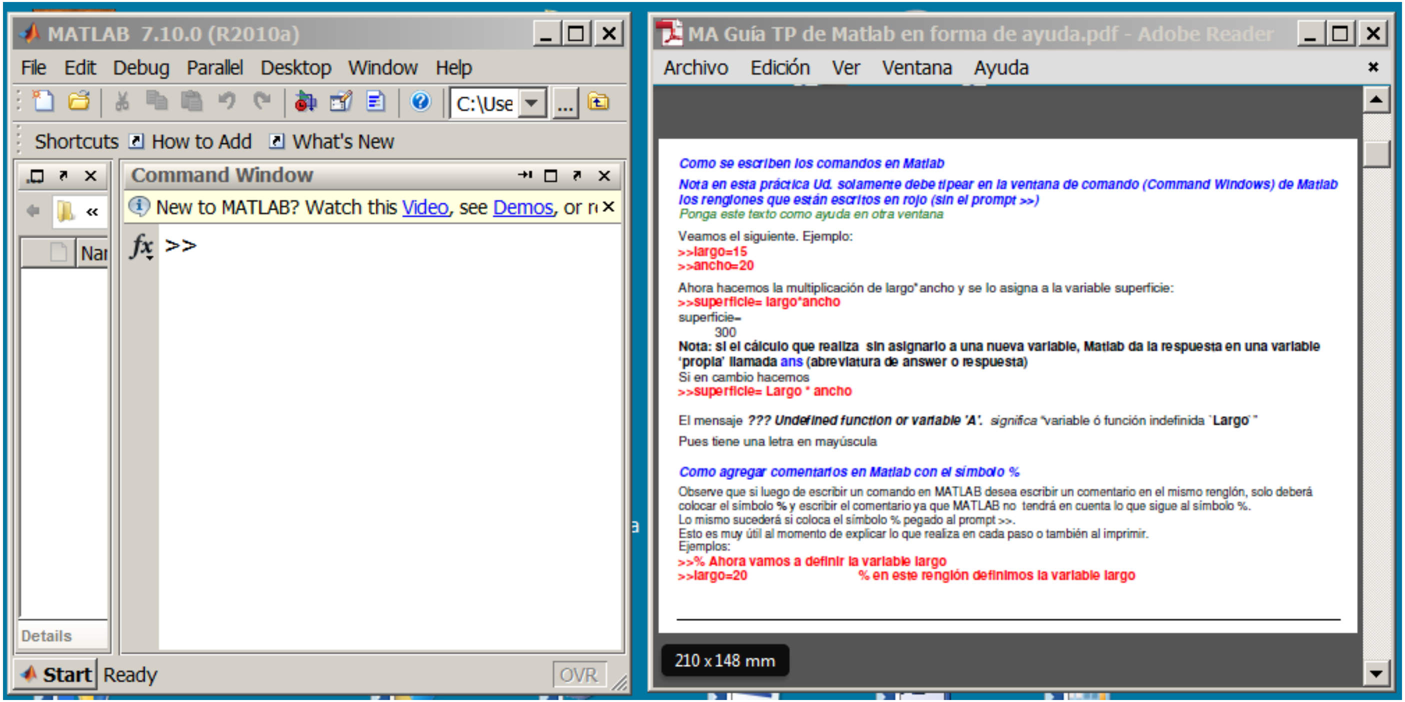
INTRODUCCION FreeMat / MATLAB / Octave

En esta CLASE 2 se desarrollan los siguientes contenidos:

- FUNCIONES NO POLINOMICAS
  - INTRINSECAS: SENO, COSENO, LOG
  - OTRAS
    - DEFINIRLAS
    - HALLAR CEROS O RAICES
  - GRAFICAR
- RESOLUCION SISTEMAS ECUACIONES LINEALES (SEL)
  - MATRICES Y VECTORES
  - RESOLUCION GRAFICA

Antes de comenzar a utilizar esta ayuda (archivo pdf) arranque el programa MATLAB o FreeMat.

Coloque las ventanas en simultáneo, en MOSAICO HORIZONTAL como observa en la figura, de modo de poder leer en la derecha y realizar los comandos (escribirlos) de esta lección.



MATLAB 7.10.0 (R2010a)
File Edit Debug Parallel Desktop Window Help
Shortcuts How to Add What's New
Command Window
New to MATLAB? Watch this Video, see Demos, or r
>>
Details
Start Ready
OVR
MA Guía TP de Matlab en forma de ayuda.pdf - Adobe Reader
Archivo Edición Ver Ventana Ayuda
Como se escriben los comandos en Matlab
Nota en esta práctica Ud. solamente debe tipear en la ventana de comando (Command Windows) de Matlab los renglones que están escritos en rojo (sin el prompt >>)
Ponga este texto como ayuda en otra ventana
Veamos el siguiente. Ejemplo:
>>largo=15
>>ancho=20
Ahora hacemos la multiplicación de largo*ancho y se lo asigna a la variable superficie:
>>superficie= largo*ancho
superficie=
300
Nota: si el cálculo que realiza sin asignarlo a una nueva variable, Matlab da la respuesta en una variable 'propia' llamada ans (abreviatura de answer o respuesta)
Si en cambio hacemos
>>superficie= Largo * ancho
El mensaje ??? Undefined function or variable 'A'. significa "variable ó función indefinida `Largo`"
Pues tiene una letra en mayúscula
Como agregar comentarios en Matlab con el símbolo %
Observe que si luego de escribir un comando en MATLAB desea escribir un comentario en el mismo renglón, solo deberá colocar el símbolo % y escribir el comentario ya que MATLAB no tendrá en cuenta lo que sigue al símbolo %.
Lo mismo sucederá si coloca el símbolo % pegado al prompt >>.
Esto es muy útil al momento de explicar lo que realiza en cada paso o también al imprimir.
Ejemplos:
>>% Ahora vamos a definir la variable largo
>>largo=20 % en este renglón definimos la variable largo
210 x 148 mm

FUNCIONES NO POLINOMICAS

FUNCIONES INTRINSECAS: SIN, COS, LOG

Como graficar otro tipo de funciones:

*LOG devuelve el logaritmo natural, para hallar el log decimal se debe usar LOG10

Se pueden graficar **dos funciones en el mismo eje**.

Primero se debe:

* generar un vector **alfa,** con valores desde 0 a 2π, o sea se definen 13 valores de ángulos, usando el operador dos puntos.

* calcular los valores del seno (variable y) y del coseno (variable z) para cada **alfa.**

* graficar las funciones usando: comando hold on y plot o bien un solo plot.

>>alfa=0:0.5:2*pi	% Este comando genera un vector de 13 valores. pi es un valor propio o intrínseco
>>y= sin(alfa)	% Se calcula el seno(alfa) y se almacena en el vector y.
>>z= cos(alfa)	% Se calcula el coseno(alfa) y se almacena en la variable o vector **z**.
>>plot(alfa,y, alfa, z)	% Se grafican ambas funciones.

Observar hay dos pares de valores, un par es (alfa, y) y el otro es (alfa, z).

Líneas más gruesas

Para dibujar con un trazo más grueso se puede agregar el parámetro LineWidth,2 al final de los pares de valores graficados

>>plot(alfa ,y, alfa , z , 'Linewidth' , 2)

Otra opción para el plot:

>>alfa=0:0.5:2*pi

>>y= sin(alfa)

>>z= cos(alfa)

>>plot(alfa,y)

>>hold on

>>plot(alfa, z)

FUNCIONES NO POLINOMICAS
Comando inline

Para definir y/o graficar una función **<u>que no es un polinomio</u> ni una función trigonométrica,** se puede definir como función 'en línea' y llamarla por ejemplo f(x). Luego puede ser utilizada, cuando la necesitemos, con sólo asignarla a una variable para un valor o varios valores de x, que son los argumentos que la función utiliza.

Conviene usar inline para definir funciones que se van a usar muchas veces en una sesión de trabajo.

>>nombredelafuncion=*inline('expresión de la función')*

<u>Ejemplo 11:</u> si se quiere definir una función potencial **(x elevado a un exponente no entero)**, x elevado a la 2.5: **$y = x^{2.5}$** y <u>se la quiere evaluar</u> en el intervalo [0, 2], con un salto o incremento **dx** de 0.2

Se debe ***definir con inline*** una función que llamaremos f1 (se puede llamar de distintas formas) pero No se pueden usar nombres de funciones propias de Matlab como: sin, cos, abs, log etc. Por eso generalmente se usa la letra f seguida de un número.

```
>> f1=inline('x.^2.5')      % se define la función potencial. Observar el punto luego de la x!!!!
>> x=0:0.2:2                %se definen todos los valores de x
>>ypot= f1(x)               % se asigna a la variable ypot todos los valores de f1, para todos los x
```

Nota: si NO se pone el punto antes del operador ^ la función f1 servirá solamente para calcular valores únicos o determinado de x. Por ejemplo >>f1(3) o bien f1(a), si la variable a=5, pero será sólo un valor.
SI NO SE PONE EL PUNTO no calculará para una serie de valores o vector x.

Ejemplos de uso del comando inline

Definir con Matlab o Freemat las siguientes funciones:

Ejemplo 12: **$y=f(x)=sen^2(x)$** se define:

```
>> f2=inline('sin(x).^2')     % ojo no se define como las trigonométricas seno y coseno!
f2 =
    Inline function:
f2(x) = sin(x).^2
```

Ejemplo 13: **$y= f(x)=sen^2(x) + cos^2(x)$** (recordar que siempre da 1, cualquiera sea x)

```
>> f3=inline('sin(x).^2+cos(x).^2')

>>f3(0)
ans=
        1
>>f3(1)          %  1 es un radián. Recordar que las funciones trigonométricas en Matlab
                    operan con ángulos expresados en radianes
ans=
        1
```

En estos ejemplos usamos **funciones propias de Matlab (sin , cos)** y las asignamos a una función definida por nosotros que llamamos f2 y f3

Comando fzero

fzero calcula la (o las) raíces de una ecuación NO POLINOMICA.

Funciona similar a ***roots*** (pero ***roots*** sirve **sólo para hallar raíces de polinomios**)

Se debe:

* definir la función

* dar un valor **próximo a la raíz,** para que a partir de ella Matlab "busque" la solución (es parecida a la Herramienta Buscar Objetivo de Excel). Para saber qué valor próximo tomar, convendrá primero graficar la función para así poder observar las cercanías de la/s raíz/ces.

Ejemplo 14: Hallar la raíz de la ecuación: $\sqrt{x} - 2 = 0$

En Matlab/freemat la **función raíz cuadrada es sqrt**

sqrt(x) – 2 = 0 (la solución es 4 pues la raíz cuadrada de 4 es 2)

Usamos aquí un ejemplo sencillo para visualizar bien la solución

PASOS EN MATLAB:

1) Se define en fecua la función con inline, podría tener otro nombre:

>>fecua = inline(‘sqrt(x) – 2’)

2) Para saber donde está la raíz aproximadamente, se **grafica la función** sqrt(x) – 2 entre x= 0 y x=5, y se observa **donde la curva corta el eje x (y=0).** En este ejemplo se observa perfectamente que es en el valor x=4

```
>> x=0: 0.2 : 5      % se define x entre 0 y 5
>> y= fecua(x)      % se evalúa la función para esos c y se almacenan en los valores de y
>>plot (x,y)        %  se grafica
>>grid on           %  se agrega una grilla y para observar que la raíz es cercana a 4
```

Entonces se usa fzero, dando 3 (podría ser 2.5 ó 3.5) como valor próximo a la raíz:

```
>>raíz=fzero(fecua,3)
ans=
4
```

Marcar la solución en el eje x (ya se explicó en clase 1)

Para marcar en la gráfica la raíz (x=4) se usa el comando PLOT. **Previo uso de hold on!!**

```
>> hold on
>>plot ( [4  4] , [-1 +1] ) % traza una línea vertical en x=4 desde y=-1 hasta y=+1
```

Matrices y Vectores con Matlab y Octave. Operaciones básicas con Matrices

Como escribir una matríz en Matlab

Recordar que la norma es ponerles nombres en mayúsculas.
Se escriben sus elementos por filas, separados por comas o espacios en blanco.
Para cambiar de fila se escribe **un punto y coma (;)**y todo se encierra todo entre corchetes.
Si tenemos la matriz A dada por:

$$\mathbf{A} = \begin{pmatrix} \mathbf{1} & \mathbf{2} \\ \mathbf{3} & \mathbf{4} \end{pmatrix}$$

Para escribirla en Matlab usamos el comando:

```
>>A = [ 1  2  ;  3  4]
A=
        1     2

        3     4
```

Transpuesta de una matriz, el operador apóstrofe

La matriz transpuesta de A se obtiene escribiendo A con un apóstrofe

```
>>B = A'      % B es la matriz transpuesta de A
B =
        1     3

        2     4
```

Como escribir un vector en Matlab

Se quiere escribir un vector FILA, por ejemplo, el vector **vf** = $\begin{bmatrix} 1 & 3 \end{bmatrix}$

>>vf = [1 3] ***% Es un vector fila pues se separan los elementos por espacios o comas***

Y si se quiere definir el vector columna **vc:**

vc = $\begin{bmatrix} 1 \\ 3 \end{bmatrix}$

Se escribe:

>>vc = [1 ; 3] ***% se separan por ; pues es un vector columna***

Otra forma es trasponer el vector fila vf usando el operador apóstrofe:

>>vc=vf ’

Solución de un Sistema de Ecuaciones Lineales (SEL) con MATLAB

Tenemos el sistema algebraico de ecuaciones de 2x2 que, como ya sabemos cada ecuación representa una recta:

$$\begin{cases} \mathbf{1\,x_1 - 1\,x_2 = -1} \\ \mathbf{1\,x_1 + 1\,x_2 = 7} \end{cases}$$

Esto en forma **matricial** se escribe como la operación entre la matríz A y el vector x para dar el vector b:

$$\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 \\ 7 \end{bmatrix}$$

En Algebra lineal verá que se expresa como: A * x = b donde A es la matríz de coeficientes, x y b son vectores de incógnitas y de términos independientes, respectivamente.

Para resolver este sistema con Matlab/Octave/FreeMat, hay que:

- ✓ definir a la matríz de coeficientes o matríz A
- ✓ definir el vector b (como vector columna)
- ✓ utilizar el operador \ (premultiplicar por la inversa)

Como se resuelve un SEL con Matlab

1º) Tomemos el sistema anterior.

Escribir el comando para definir la matriz de coeficientes (se puede llamar A):

>> A = [1-1 ; 1 1]

A=

1   -1
1   1

2º) Escribir el vector **columna b** (o términos independientes del sistema)

Observar que el vector b (columna) se debe escribir separando los elementos por punto y coma ;

>>b = [-1 ; 7]

b =
-1
7

3º) Utilizar el operador **** para resolver el problema. Se escribe entonces:

>>x =   A \ b   % Obtenemos así los valores de $x_1 = 3$ y $x_2 = 4$ que son la solución del sistema

x =
3.0000

4.0000

Los resultados pueden verificarse. Para ello se plantea calcular A * x, y ver si se obtiene b:

```
>> A * x          % Verificación

ans =
   -1
    7             % vemos que son los valores del vector b.
```