

Curso de MatLab

Nicolas Cardona Ramirez

23 de enero de 2020

1. Episodio 10: Formato de Salida

1.1. Notación Científica

Se utiliza para escribir número con un valor muy grande o muy pequeño. En MatLab se utiliza:

- 6.022e23

1.2. Cambiar formato de Números

- Para utilizar 14 cifras decimales se utiliza `format long`
- Para utiliza 2 cifras decimales se utiliza `format bank` y se aproxima la última cifra decimal
- El formato por defecto con 4 cifras decimales es `format short`
- El `format +` despliega solo los signos positivos y negativos dentro de una matriz
- El `format rat` despliega los números como números racionales, es decir, como fracciones
- Para cambiar la forma de la notación científica se utiliza los comandos anteriores más la letra e; `format short e`

2. Episodio 11: Funciones en MatLab

2.1. Funciones Internas

Las funciones internas más utilizadas son:

- Para usar raíz cuadrada se utiliza `sqrt()` y puede ser una matriz o un vector
- La función `rem(a,b)` calcula el residuo de dos números
- La función `[x,y] = size(d)` calcula el tamaño de una matriz o de un vector y tiene dos parámetros de salida donde `x` es el número de filas y `y`

2.2. Funciones anidadas

Para realizar una función anidada se utiliza `sqrt(sin(x))`

3. Episodio 12: Funciones Matemáticas

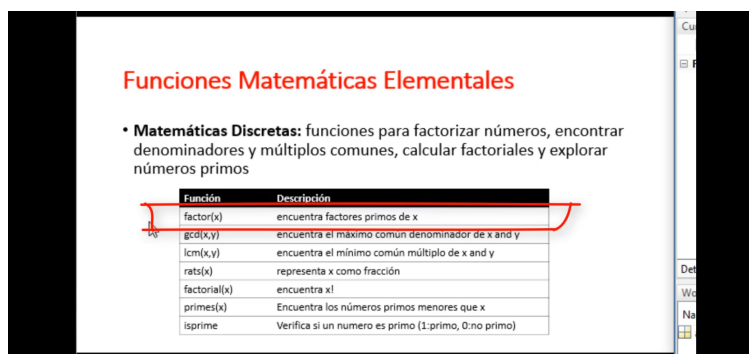
Las funciones matemáticas esenciales son:

- `abs()`: absoluto
- `sqrt()`: Raíz cuadrada
- `nthroot (x,n)`: Enésima raíz real de x
- `sign()`: Regresa el signo del número
- `rem(x,y)`: Residuo de x/y
- `log()`: Calcula el logaritmo natural de x
- `log10()`: Calcula el logaritmo base 10 de x

3.1. Funciones de Redondeo

Las funciones de redondeo son:

- `round(b)`: Redondea al número entero más cercano
- `fix(b)`: Redondea al entero más cercano a cero
- `floor(b)`: Redondea al entero mas cercano hacia el infinito negativo
- `ceil(b)`:Redondea hacia el entero más cercano hacia el infinito positivo



Funciones Matemáticas Elementales

- **Matemáticas Discretas:** funciones para factorizar números, encontrar denominadores y múltiplos comunes, calcular factoriales y explorar números primos

Función	Descripción
<code>factor(x)</code>	encuentra factores primos de x
<code>gcd(x,y)</code>	encuentra el máximo común denominador de x and y
<code>lcm(x,y)</code>	encuentra el mínimo común múltiplo de x and y
<code>rats(x)</code>	representa x como fracción
<code>factorial(x)</code>	encuentra x!
<code>primes(x)</code>	Encuentra los números primos menores que x
<code>isprime</code>	Verifica si un numero es primo (1:primo, 0:no primo)

Figura 1: Funciones matemáticas discretas

4. Episodio 13: Funciones Trigonómicas

Los ángulos deben de estar en radianes

$$grados = radianes \left(\frac{180}{\pi} \right) \quad (1)$$

$$radianes = grados \left(\frac{\pi}{180} \right) \quad (2)$$

Las funciones trigonométricas estandares son:

- | | |
|---------------------------------|--------------------------------------|
| ■ $\sin(x)$ con x en radianes | ■ $\text{sind}(x)$ con x en grados |
| ■ $\cos(x)$ con x en radianes | ■ $\text{cosd}(x)$ con x en grados |
| ■ $\tan(x)$ con x en radianes | ■ $\text{tand}(x)$ con x en grados |
| ■ $\csc(x)$ con x en radianes | ■ $\text{cscd}(x)$ con x en grados |
| ■ $\sec(x)$ con x en radianes | ■ $\text{secd}(x)$ con x en grados |
| ■ $\cot(x)$ con x en radianes | ■ $\text{cotd}(x)$ con x en grados |

4.1. Funciones Trigonómicas Inversas

Las funciones trigonométricas inversas son:

- | | |
|--|---------------------------------------|
| ■ $\text{asin}(x)$ con x en radianes | ■ $\text{asind}(x)$ con x en grados |
| ■ $\text{acos}(x)$ con x en radianes | ■ $\text{acosd}(x)$ con x en grados |
| ■ $\text{atan}(x)$ con x en radianes | ■ $\text{atand}(x)$ con x en grados |
| ■ $\text{acsc}(x)$ con x en radianes | ■ $\text{acscd}(x)$ con x en grados |
| ■ $\text{asec}(x)$ con x en radianes | ■ $\text{asecd}(x)$ con x en grados |
| ■ $\text{acot}(x)$ con x en radianes | ■ $\text{acotd}(x)$ con x en grados |

4.2. Funciones Trigonómicas Hiperbólicas

- | | |
|----------------------------------|--|
| ■ $\sinh(x)$ con x en radianes | ■ $\text{csch}(x)$ con x en radianes |
| ■ $\cosh(x)$ con x en radianes | ■ $\text{sech}(x)$ con x en radianes |
| ■ $\tanh(x)$ con x en radianes | ■ $\text{coth}(x)$ con x en radianes |

5. Episodio 14: Análisis de Datos en MatLab

- Para encontrar el **máximo** de una matriz entrega el resultado del máximo por cada columna o un vector se utiliza la función `max()` y para guardar su posición se utiliza `[a,b] = max()` donde **a** es el máximo y **b** es su posición.
- Para encontrar el **mínimo** se utiliza la función `min()`
- Para **trasponer** una matriz se utiliza el nombre de la matriz mas comilla; `max(y')`
- Para hacer una **sumatoria** de los elementos de un vector o matriz se utiliza el comando `sum()`
- Para hacer una suma acumulada se tiene que `cumsum()` que acumula la suma por filas
- De la misma manera funciona el **productorio** con los comandos `prod()` y `cumprod()`
- Para ordenar los datos de forma ascendente o descendente se utiliza el comando `sort()` y de manera descendente `sort(x, 'descend')`
- Para ordenar una columna determinada se utiliza el comando `sortrows(y,n)`
- El comando `size` determina el tamaño de la matriz y `length` arroja el valor mayor del tamaño de la matriz

6. Episodio 15: Números Complejos

MatLab incluye varias funciones principales con números complejos. Se presentan como:

- $a = 12 + 7i$ ó $a = 12 + 7j$ o también se puede utilizar el comando `complex(a,i)`; para que lo anterior funciones, las variables *i* y *j* no deben de estar siendo empleadas. Lo anterior también puede ser usado con vectores.
- Se puede llamar la parte real de un número complejo y la parte imaginaria con los comandos `real()` y `imag()`
- El comando `isreal()` se emplea para conocer si el número es real o no con indicadores lógicos. El conjugado de un número complejo se obtiene con `conj()`
- Para escribir un número complejo en coordenadas polares se utiliza los comandos `abs()` para encontrar el radio y el comando `angle` para determinar el ángulo

7. Episodio 16: Graficar Vectores en 2D

- Se definen los datos de vectores x y y . Luego se utiliza el comando `plot`. Para agregar un título a la gráfica se utiliza el comando `title("")`. Para nombrar los ejes se pone `xlable("")` y `ylabel("")`. Para poner una grilla se utiliza `grid on` y el comando `legend` pone nombres en la gráfica.
- El comando `hold` y luego el comando `plot` para sobreponer las gráficas. Para agregar las leyendas a las gráficas se utiliza el comando `legend('Primera función', 'Segunda función')`

7.1. Creación de Gráficas Múltiples

- Con `figure ()` se crean las figuras en ventanas diferentes

7.2. Subplot

- Para crear varias figuras en una misma ventana se utiliza `figure` y `subplot(2,1,1)` donde el primer número es el número de filas, el segundo el número de columnas y el tercero el eje actual

7.3. Línea, Color y Estilo de Marca

Los diferentes estilos de gráficas son:

Tipo de línea	Indicador	Tipo de punto	Indicador	Color	Indicador
sólida	-	punto	.	azul	b
punteada	:	círculo	o	verde	g
raya-punto	-.	marca x	x	rojo	r
rayada	--	más	+	cian	c
		estrella	*	magenta	m
		cuadrado	s	amarillo	y
		diamante	d	negro	k
		triángulo abajo	v		
		triángulo arriba	^		
		triángulo izquierda	<		
		triángulo derecha	>		
		pentagrama	p		
		hexagrama	h		

Figura 2: Estilos y colores en gráficas

- Para ello se pone `plot(a,b,'atributos')`

- Para cambiar el tamaño de la línea se utiliza el comando `linewidth` y para el tamaño de la fuente `FontSize`. Para cambiar la fuente de los Axes se usa el comando `gca`.
- `plot(a,b,'atributos','linewidth',6)`
- `title('Función de Seno','FontSize',15)`
- `set(gca,'fontsize',14)`

7.4. Escalamiento de Ejes

- Se utilizan los comandos `axis([xmin xmax ymin ymax])`
- Para crear anotaciones dentro de la gráfica se utiliza el comando `text(Posición en x, Posición en y, 'Leyenda')`
- Para controlar más sobre los parámetros de plot, utilizar el comando `help plot`

8. Episodio 17: Animaciones en Gráficas MatLab

Para ello se utiliza la interfaz interna del programa GUIDE. Los pasos para ello son los siguientes:

- Crear una figura: Este es el espacio de trabajo del programa; Para ello se persigue así:

```
fig(1)=figure('name','Monitor','menubar','none','position',[200 200 800 700],'color',[0.9 0.6 0.3])
```
- Para crear los Axes:

```
axe(1)=axes('parent',fig(1),'units','pixels','position',[60 80 600 550],'xlim',[0 40]. 'ylim',[-3 3], 'xgrid','on','ygrid','on')
```
- Para el nombre de los ejes:

```
set(get(axe(1),'Xlabel'),'String','Tiempo (Seg)')
set(get(axe(1),'Ylabel'),'String','Función')
```
- Para la creación de los atributos de las líneas:

```
lin(1)=Line('parent','axe(1)','xdata',[ ],'ydata',[ ],'Color','r','LineWidth',2.5)
lin(2)=Line('parent','axe(1)','xdata',[ ],'ydata',[ ],'Color','k','LineWidth',2)
```

VER EPISODIO COMPLETO

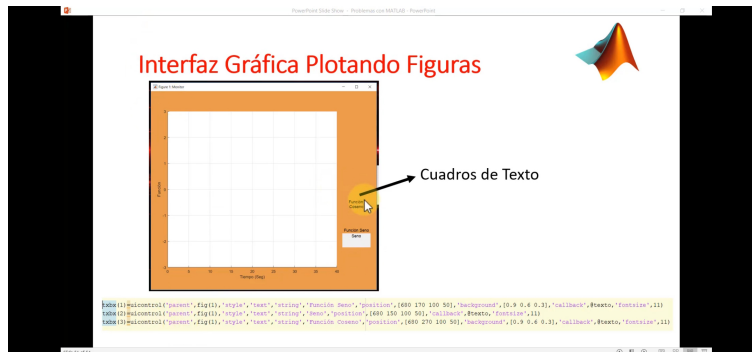


Figura 3: Cuadro de texto



Figura 4: Botones

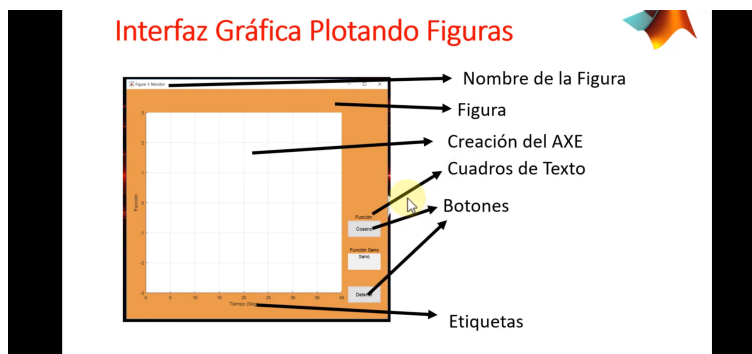


Figura 5: Interfaz final

9. Episodio 18: Gráficas Polares en MatLab

Las otras formas de representar los datos son:

- La función `polar(theta,r)` con el ángulo `theta` en radianes
Ejemplo 1:
`figure`
`x=0:pi/100:pi;`
`y = sin(x);`
`polar(x,y)`
- Flor:
`figure`
`theta = 0:0.01*pi:2*pi;`
`r = 5*cos(4*theta);`
`polar(theta,r)`
- Estrella
`figure`
`theta = pi/2:4/5*pi:4.8*pi;`
`r=ones(1,6);`
`polar(theta,r)`

10. Episodio 19: Arreglo de Gráficas

Para emplear las gráficas logarítmicas se utiliza de la siguiente manera:

Gráficas rectangular y logarítmica	
<code>plot(x,y)</code>	Genera una gráfica lineal de los vectores x y y
<code>semilogx(x,y)</code>	Grafica los vectores x y y con escala logarítmica para x y lineal para y
<code>semilogy(x,y)</code>	Grafica los vectores x y y con escala logarítmica para y y lineal para x
<code>loglog(x,y)</code>	Grafica los vectores x y y con escala logarítmica para x y y

Figura 6: Arreglo logarítmico para gráficas

- Ejemplo: Graficar $y = 5x^2$ con los cuatro enfoques de escalamiento
`x = 0 : 0,5 : 50;` [Vector de `x`]
`y = 5 * x.^2;` [Función de `y` (El punto es importante para que eleve cada elemento del vector al cuadrado)]
`subplot(2,2,1)`
`plot(x,y)`
`title('Polinomial-Lineal-lineal')`
`ylabel('y'), grid`


```

subplot(2,2,2)
semilogx(x,y)
title('Polinomial-Logaritmica-Lineal')
ylabel('y'), grid
subplot(2,2,3)
semilogy(x,y)
title('Polinomial-Lineal-Logaritmica')
xlabel('x'), grid
subplot(2,2,4)
loglog(x,y)
title('Polinomial-Logaritmica- Logaritmica')
xlabel('x'), grid


```

11. Episodio 20: Gráfica de Barras y Pastar en MatLab

Los diferentes tipos de gráficos disponibles son:

Gráficas de barras y de pastel

- Las gráficas de barra, histograma y de pastel son formas populares para reportar datos.



Gráficas de barra y de pastel	
bar(x)	Gráfica de Barras del vector x. Si x es matriz se agrupa los datos por fila
barh(x)	Gráfica de barras horizontal del vector x
bar3(x)	Gráfica de barras tridimensional
bar3h(x)	Gráfica de barras horizontal tridimensional
pie(x)	Gráfica de pastel, cada elemento de la matriz es una porción.
pie3(x)	Gráfica de pastel tridimensional
hist(x)	Genera un Histograma.

Figura 7: Tipos de gráficas

- Ejemplo 1:**
 $x = [1 \ 2 \ 5 \ 4 \ 8]$; $y = [x; 1:5]$

```

figure
subplot(2,2,1)
bar(x)
title('Gráfica de Barras del vector x')
subplot(2,2,2)
bar(y)
title('Gráfica de la matriz y')

```

```
subplot(2,2,3)
bar3(x)
title('Gráfica tridimensional de Barras del vector  $x$ ')
subplot(2,2,4)
bar3h(y)
title('Gráfica tridimensional de Barras Horizontales de la matriz  $y$ ')
```

12. Episodio 21: Histograma

Para hacer un histograma, el defecto son 10 depósitos o categorías.

- Ejemplo 1: Calificaciones de alumnos

El comando a utilizar es `hist(x,n)` donde n es el número de depósitos. También se puede almacenar el histograma en una variable `A = hist(x,n)`

`hist help` recomienda usar en vez `HISTOGRAM`

13. Episodio 22: Graficar en MatLab en 3D

MatLab utiliza los siguientes comandos para graficar:

Gráficas Tridimensionales

- MATLAB ofrece una variedad de comandos para gráficas tridimensionales, como por ejemplo:

Gráficas Tridimensionales	
<code>plot3(x,y,z)</code>	Crea una grafica lineal tridimensional
<code>comet3(x,y,z)</code>	Genera una versión animada de <code>plot3</code>
<code>mesh(z)</code>	Crea una superficie de malla
<code>mesh(x,y,z)</code>	
<code>surf(z)</code>	Crea una grafica de superficie similar a la función mesh
<code>surf(x,y,z)</code>	
<code>shading interp</code>	Interpola entre los colores usados para ilustrar gráficas de superficie
<code>shading flat</code>	Colorea cada sección de retícula con un color sólido
<code>colormap(map_name)</code>	Permite al usuario seleccionar el patron de color a usar en las gráficas de superficie
<code>contour(z)</code>	Genera una gráfica de contorno
<code>contour(x,y,z)</code>	
<code>surf(z)</code>	Crea una gráfica de superficie combinada con una gráfica de contorno
<code>surf(x,y,z)</code>	
<code>pcolor(z)</code>	crea una gráfica en pseudocolor
<code>pcolor(x,y,z)</code>	

Bibliografía usada para hacer el video: MATLAB® para ingenieros

Figura 8: Tipos de gráficas en 3D

14. Episodio 23: Superficie en Gráficas 3D

- Las gráficas Mesh son muy utilizadas en procesos de optimización de ingeniería

- La función Surf crea una superficie tridimensional en vez de un mallado, para controlar las sombras se utiliza el comando **shading + opción**

15. Episodio 24: Gráficas de Contorno y Superficie

- El comando **[X,Y] = meshgrid(x,y)** se utiliza para crear una grilla
- La función **surf** combina la gráfica de superficie con la gráfica de contorno

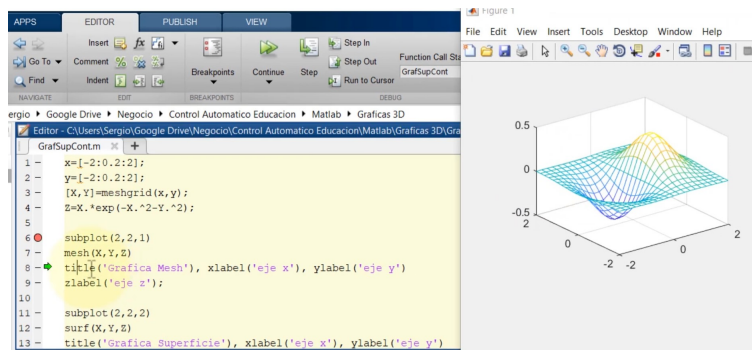


Figura 9: Ejemplo del comando meshgrid

- Para utilizar el colormap
 - El esquema de color usado en las gráficas de superficie se puede controlar con la función **colormap**. Por ejemplo,
 - **colormap(gray)**



Figura 10: Ejemplos del colormap

- En el comando **pcolor** para quitar la cuadrícula se utiliza el comando **shading interp**