

2020

Manual MATLAB



Duván Mejía

UNIVERSIDAD SURCOLOMBIANA

25-11-2020

Tabla de contenido

Clase semana 1:	2
Clase semana 2	8
Clase semana 3 & 4	13
Clase Semana 5	21
GRAFICOS MATLAB	26
Clase Semana 8	38
Manual 3 de febrero 2021	46

Clase semana 1:

who: Aquí podemos observar las variables que tengo.

```
a =  
  
    4  
  
>> who  
  
Your variables are:  
  
a
```

whos: aquí vemos todas las características de las variables.

```
>> whos  
Name      Size      Bytes  Class  Attributes  
  
a         1x1         8  double  
b         1x1         8  double  
c         1x1         8  double
```

Clear + Variable: Borramos la variable seleccionada.

```
>> clear b  
>> who  
  
Your variables are:  
  
a  c
```

Clear all: borramos todas las variables.

```
>> clear all  
>> who  
>> |
```

disp: Método para Imprimir una variable.

```
>> disp(x)
Duvan
>> x = 'Duvan';
>> disp(x)
Duvan
```

ans: Variable temp Operación del dato anterior.

```
>> 3 + 5

ans =

     8

>> suma = 4 + 1

suma =

     5

>> ans

ans =

     8
```

Save: guardar archivo.mat y muestra la ruta

```
>> save

Saving to: C:\Users\Mr Mejia\Documents\MATLAB\Examples\R2020a\matlab\CallAFunctionThatReturnsOutputExample\matlab.mat
```

cls : limpia la pantalla.

```
>> a = 4;
>> b = 3

b =

     3

>> x = 'Duvan';
>> cls
.
```

```
f3 >> |
```

Load: cargar el archivo Matlab

```
>> load

Loading from: C:\Users\Mr Mejia\Documents\MATLAB\Examples\R2020a\matlab\CallAFunctionThatReturnsOutputExample\matlab.mat
```

format –long : Formato de decimal largo

```
>> format long
>> pi

ans =

    3.141592653589793
```

format -shortEng: Formato de Ingeniería

```
>> format shortEng
>> pi

ans =

    3.1416e+000
```

format -short : Formato de decimales corto

```
>> format short
>> pi
ans =
    3.1416
```

format- shortG: Formato más pequeño de decimales

```
>> format shortG
>> pi
ans =
    3.1416
```

Eps: Es el valor mas pequeño y equivalente a `eps(1.0)` y `eps('doble')`.

```
>> eps
ans =
    2.2204e-16
```

Pi: función matematica

```
>> pi
ans =
    3.1416
```

Inf: Variable infinita resultante de la división entre Zero

```
>> 6 / (4-4)
ans =
    Inf
```

NaN: Cuando la variable no tiene ningún dato o indeterminado.

```
>> nan(5)

ans =

    NaN    NaN    NaN    NaN    NaN
    NaN    NaN    NaN    NaN    NaN
    NaN    NaN    NaN    NaN    NaN
    NaN    NaN    NaN    NaN    NaN
    NaN    NaN    NaN    NaN    NaN
```

Compleja(i+j): utilizamos con valores complejos.

```
>> sqrt(-1)

ans =

    0.0000 + 1.0000i
```

Realmin: más pequeño a utilizar.

```
>> format long e
>> f = realmin

f =

    2.225073858507201e-308
```

Realmax: valor más grande a utilizar.

```
>> format long e
>> f = realmax

f =

1.797693134862316e+308
```

Clock: formato de fecha.

```
>> format shortg
c = clock

c =

2020      11      26      21      2      38.18
```

Date: formato de fecha

```
>> z = date

z =

'26-Nov-2020'
```

Calendar: variable del calendario

```
>> calendar(1998,01)

      Jan 1998
  S    M    Tu    W    Th    F    S
  0     0     0     0     1     2     3
  4     5     6     7     8     9    10
 11    12    13    14    15    16    17
 18    19    20    21    22    23    24
 25    26    27    28    29    30    31
 0     0     0     0     0     0     0
```

Funciones de Aproximación

fix: Redondea el valor más cercano a cero.

```
>> x

x =

    3.2

>> y = fix(x)

y =

    3
```

Floor: Valor negativo hacia infinito.

```
>> X = [-1.9 -0.2 3.4; 5.6 7.0 2.4+3.6i];
>> Y = floor(X)

Y =

    -2 + 0i    -1 + 0i     3 + 0i
     5 + 0i     7 + 0i     2 + 3i
```

Round: Redondea al valor decimal o entero infinito más cercano.

```
>> y = round(pi,3)

y =

    3.142

>> y = round(pi)

y =

    3
```


Clase semana 2

Trigonometría

Sin: Seno dado en radianes.

```
>> sin(pi)

ans =

    1.2246e-16
```

Sind: Seno dado en grados.

```
>> sind(180)

ans =

    0
```

asin: seno inverso en radianes.

```
>> asin(pi)

ans =

    1.5708 - 1.8115i
```

Asind: seno inverso en grados.

```
>> asind(pi)
```

```
ans =
```

```
9.0000e+01 - 1.0379e+02i
```

Sinh: seno hiperbólico.

```
>> sinh(pi)
```

```
ans =
```

```
11.5487
```

Cos: Coseno en radianes.

```
>> cos(pi)
```

```
ans =
```

```
-1
```

Cosd: coseno en grados.

```
>> cosd(pi)
```

```
ans =
```

```
0.9985
```

Acos: inverso de coseno en radianes.

```
>> acos(pi)
```

```
ans =
```

```
0.0000 + 1.8115i
```

Acosd: inverso de coseno en grados.

```
>> acosd(pi)
```

```
ans =
```

```
0.0000e+00 + 1.0379e+02i
```

Cosh: coseno hiperbólico en radianes.

```
>> cosh(pi)
```

```
ans =
```

```
11.5920
```

Acosh: coseno hiperbólico inverso en radianes.

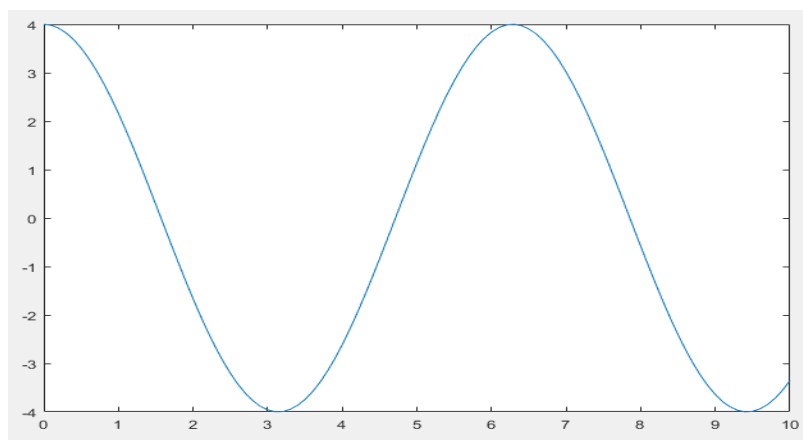
```
>> acosh(pi)
```

```
ans =
```

```
1.8115
```

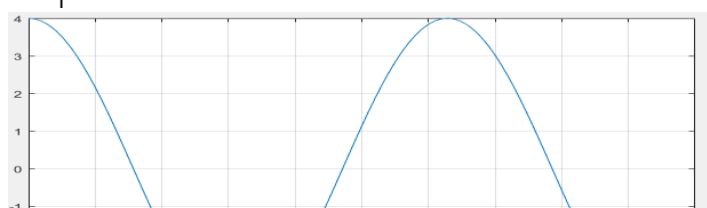
Plot: Graficar funciones.

```
t = 0:0.1:10;  
y = 4 * cos(t);  
plot(t,y);  
% Grafico de la funcion coseno en  
% radianes con amplitud de 4 |
```



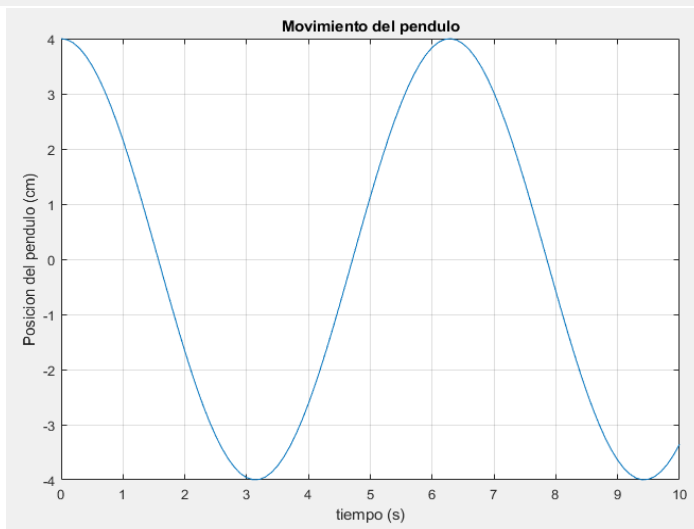
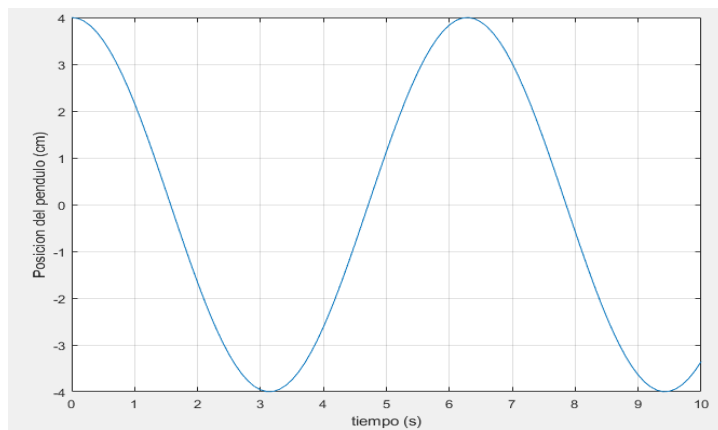
Grid – on - off: Activar cuadrícula o desactivar cuadrícula

```
>> grid
```



Label : Etiqueta de clase.

```
>> xlabel("tiempo (s)")  
>> ylabel("Posicion del pendulo (cm)")
```



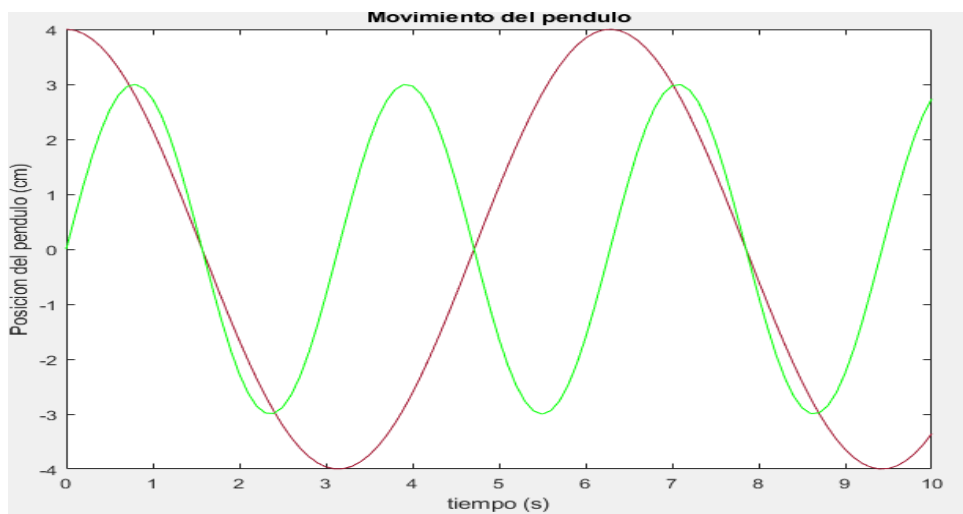
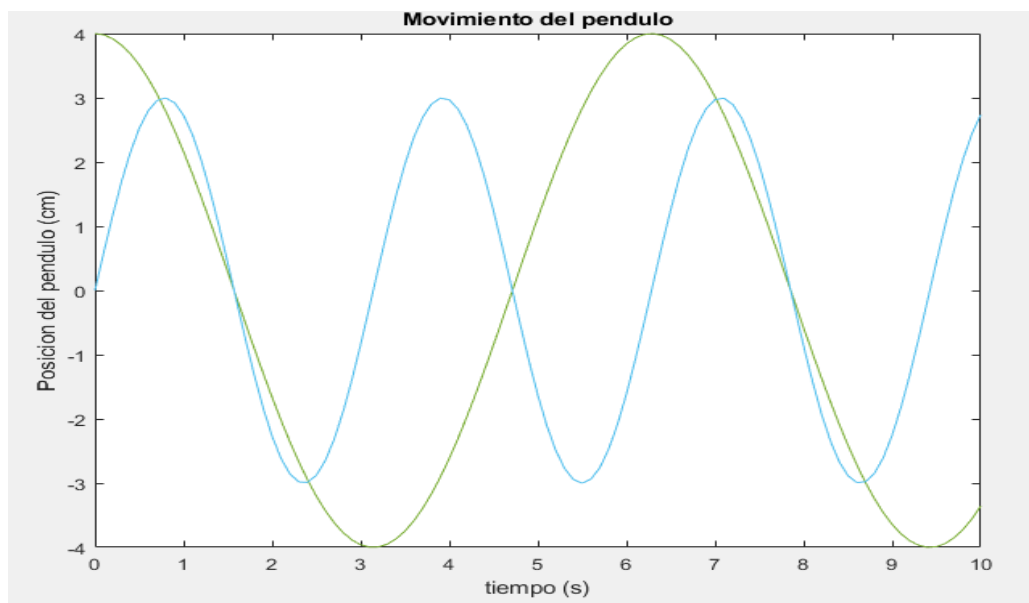
Workspace: Muestra la venta de trabajo donde están las variables

```
>> workspace
```

Workspace	
Name	Value
ans	1.8115
t	1x101 double
x	4
y	1x101 double

Hold on : Muestro dos graficas distintas en una misma figura.

```
hold on
y2=3*sin(2*t);
plot(t,y2);
```



Subplot: Graficar subfunciones.

```
subplot(2,3,4)
plot(t,y1)|
```

Clase semana 3 & 4

Cross: Producto cruz entre dos vectores.

```
>> x = [3 2 0];
y = [2 -1 0];
>> cross(x,y)
```

```
ans =

     0     0    -7
```

dot: Producto escalar entre dos vectores.

```
>> dot(x,y)
```

```
ans =

     4
```

det: Hallar la determinante.

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

A =

     1     2     3
     4     5     6
     7     8     9

determinande_A =

-9.5162e-16
```

inv: Hallar la matriz inversa

```
>> invA=inv(A)
Warning: Matrix is close to singular or badly
scaled. Results may be inaccurate. RCOND =
2.202823e-18.

invA =

1.0e+16 *

     0.3153    -0.6305     0.3153
    -0.6305     1.2610    -0.6305
     0.3153    -0.6305     0.3153
```

fliplr: devuelve A con sus columnas volteadas en la dirección izquierda-derecha

```
>> fliplr(A)
```

```
ans =
```

3	2	1
6	5	4
9	8	7

flipud: Devuelve las filas en dirección arriba a abajo, alrededor del eje horizontal.

```
>> flipud(A)
```

```
ans =
```

7	8	9
4	5	6
1	2	3

reshape: reformula A utilizando el vector de tamaño, sz, para definir size(B)

```
>> A = 1:10;
```

```
B= reshape(A,[5,2])
```

```
B =
```

1	6
2	7
3	8
4	9
5	10

rot90: gira la matriz A en sentido contrario a las agujas del reloj en 90 grados.


```
>> rot90(A)
```

```
ans =
```

```
10  
9  
8  
7  
6  
5  
4  
3  
2  
1
```

Rot90 + n: gira la matriz A n veces en el sentido contrario al reloj.

```
>> rot90(A,2)
```

```
ans =
```

```
10    9    8    7    6    5    4    3    2    1
```

sqrtm: devuelve la raíz cuadrada principal de la matriz A.

```

A =

     5     -4      1      0      0
    -4      6     -4      1      0
     1     -4      6     -4      1
     0      1     -4      6     -4
     0      0      1     -4      6

>> X = sqrtm(A)

X =

    2.0015   -0.9971    0.0042    0.0046    0.0032
   -0.9971    2.0062   -0.9904    0.0118    0.0094
    0.0042   -0.9904    2.0171   -0.9746    0.0263
    0.0046    0.0118   -0.9746    2.0503   -0.9200
    0.0032    0.0094    0.0263   -0.9200    2.2700

```

sqrt: devuelve la raíz cuadrada de cada elemento de la matriz x . Para los elementos de x que son negativos o complejos, `sqrt(x)` produce resultados complejos.

```

>> sqrt(A)

ans =

Columns 1 through 4

    2.2361 + 0.0000i    0.0000 + 2.0000i    1.0000 + 0.0000i    0.0000 + 0.0000i
    0.0000 + 2.0000i    2.4495 + 0.0000i    0.0000 + 2.0000i    1.0000 + 0.0000i
    1.0000 + 0.0000i    0.0000 + 2.0000i    2.4495 + 0.0000i    0.0000 + 2.0000i
    0.0000 + 0.0000i    1.0000 + 0.0000i    0.0000 + 2.0000i    2.4495 + 0.0000i
    0.0000 + 0.0000i    0.0000 + 0.0000i    1.0000 + 0.0000i    0.0000 + 2.0000i

Column 5

    0.0000 + 0.0000i
    0.0000 + 0.0000i
    1.0000 + 0.0000i
    0.0000 + 2.0000i
    2.4495 + 0.0000i

```

rand: Números aleatorios distribuidos uniformemente.

```
>> A = rand(2,3,4)
```

```
A(:,:,1) =
```

0.9293	0.1966	0.6160
0.3500	0.2511	0.4733

```
A(:,:,2) =
```

0.3517	0.5853	0.9172
0.8308	0.5497	0.2858

```
A(:,:,3) =
```

0.7572	0.3804	0.0759
0.7537	0.5678	0.0540

```
A(:,:,4) =
```

0.5308	0.9340	0.5688
0.7792	0.1299	0.4694

```
...
```

permute: Me cambia el orden de las dimensiones en este caso filas, columnas y cantidad.

```
>> B = permute(A,[3 2 1]);
```

```
>> size(B)
```

```
ans =
```

4	3	2
---	---	---

diag: Traza Diagonal y el resto 0 en la matriz.

```
>> v = [1 2 3 4]
M2 = diag(v)
```

v =

1	2	3	4
---	---	---	---

M2 =

1	0	0	0
0	2	0	0
0	0	3	0
0	0	0	4

tril: La parte triangular inferior de la matriz.

```
>> M = [1 2 3 4; 5 6 7 8; 9 10 11 12]
tril(M)
```

M =

1	2	3	4
5	6	7	8
9	10	11	12

ans =

1	0	0	0
5	6	0	0
9	10	11	0

triu: Matriz triangular superior.

```
triu(M)
```

ans =

1	2	3	4
0	6	7	8
0	0	11	12

Guardar un polinomio:

```
p = [1 -1 -26 -24]

p =

     1     -1    -26    -24
```

roots: Calculamos las raíces del polinomio.

```
>> roots(p)

ans =

     6.0000
    -4.0000
    -1.0000
```

polyval: Evaluar el polinomio en un punto dado.

```
>> polyval(p,1)

ans =

    -50
```

conv: devuelve la convolución Circunvolución de los vectores u y v. Si u y v son vectores de coeficientes polinómicos, convolución equivale a multiplicar los dos polinomios.

```
>> u = [1 0 1];
v = [2 7];
>> w = conv(u,v)

w =

     2     7     2     7
```

deconv: División de un polinomio.

```
>> w = conv(u,v)

w =

     2     7     2     7
```

Clase Semana 5

polyder : Derivada de un polinomio ejemplo: $p(x) = 3x^5 - 2x^3 + x + 5$. Y la derivada es : $q(x) = 15x^4 - 6x^2 + 1$.

```
>> p = [3 0 -2 0 1 5];  
q = polyder(p)  
  
q =  
  
15.00      0      -6.00      0      1.00
```

fieldnames : Conozco los campos de una estructura.

```
>> fieldnames(cliente)
```

```
ans =  
  
5x1 cell array  
  
{'id'      }  
{'name'    }  
{'lastname'}  
{'city'    }  
{'email'   }
```

isfield: Devuelve un valor lógico 1 si la clave esta en la estructura, de lo contrario retorna un 0.

```
>> isfield(cliente, 'city')
```

```
ans =  
  
logical  
  
1
```

isstruct: Devuelve un valor lógico 1 si el valor es una estructura de lo contrario retorna 0.

```
>> isstruct(cliente)
```

```
ans =  
  
logical  
  
1
```

rmfield: Elimina un campo de la estructura.

```
>> rmfield(cliente, 'city')

ans =

    struct with fields:

        id: 1
        name: 'mafe'
        lastname: 'rodriguez'
        email: 'mafe@outlook.com'
```

celda o cell : devuelve una Matriz de celda

```
>> celda(1) = {'Duvan Mejia'}
celda(2) = {eye(4)}

celda =

    1×1 cell array

    {'Duvan Mejia'}

celda =

    1×2 cell array

    {'Duvan Mejia'}    {4×4 double}
```

celldisp: Me retorna las celdas

```
>> celldisp(celda)

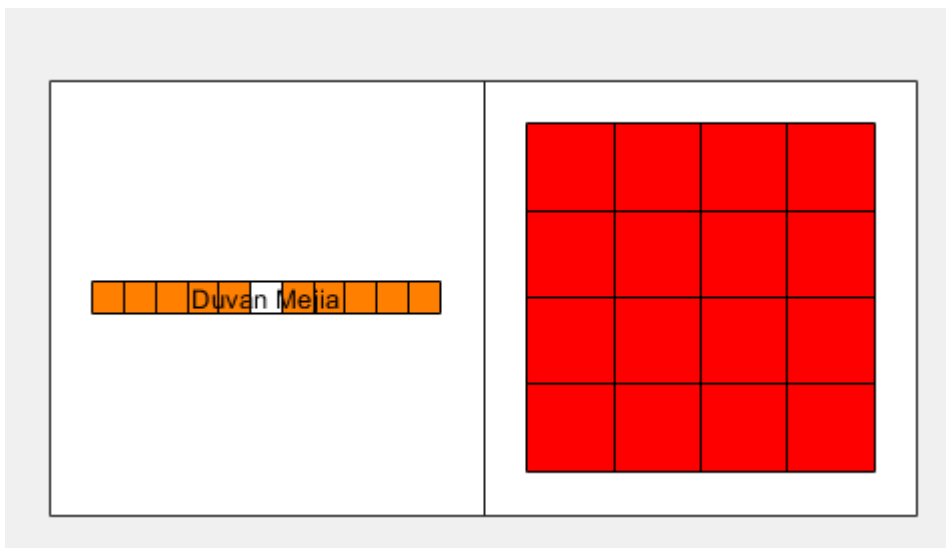
celda{1} =

Duvan Mejia

celda{2} =

     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

cellplot: me grafica la celda.



iscell : Me retorna un valor lógico de 1 si celda es una cell de lo contrario me retorna un 0

```
>> iscell(celda)
```

```
ans =
```

```
logical
```

```
1
```

and: devuelve un valor lógico entre el vector A y el vector B.

```
>> and(A,B)
```

```
ans =
```

```
1×4 logical array
```

```
0 0 0 1
```

or: devuelve valor lógico si cumple la condición de or

```
>> or(A,B)
```

```
ans =
```

```
1×4 logical array
```

```
0 1 1 1
```


not: devuelve un valor lógico 1 si el valor del vector es false y 0 si el vector es true.

```
>> not(A)

ans =

1×4 logical array

1 1 0 0
```

Xor: Siempre que las entradas de los vectores sean distinta la salida es 1

```
>> xor(A,B)

ans =

1×4 logical array

0 1 1 0
```

for: repetición número de veces

```
>> for v = 1.0:-0.2:0.0
    t = 2 * v;
    disp(v);
end

1

0.8000

0.6000

0.4000

0.2000

0
```

while: repite la declaración cuantas veces sea necesario si la condición es verdadera.

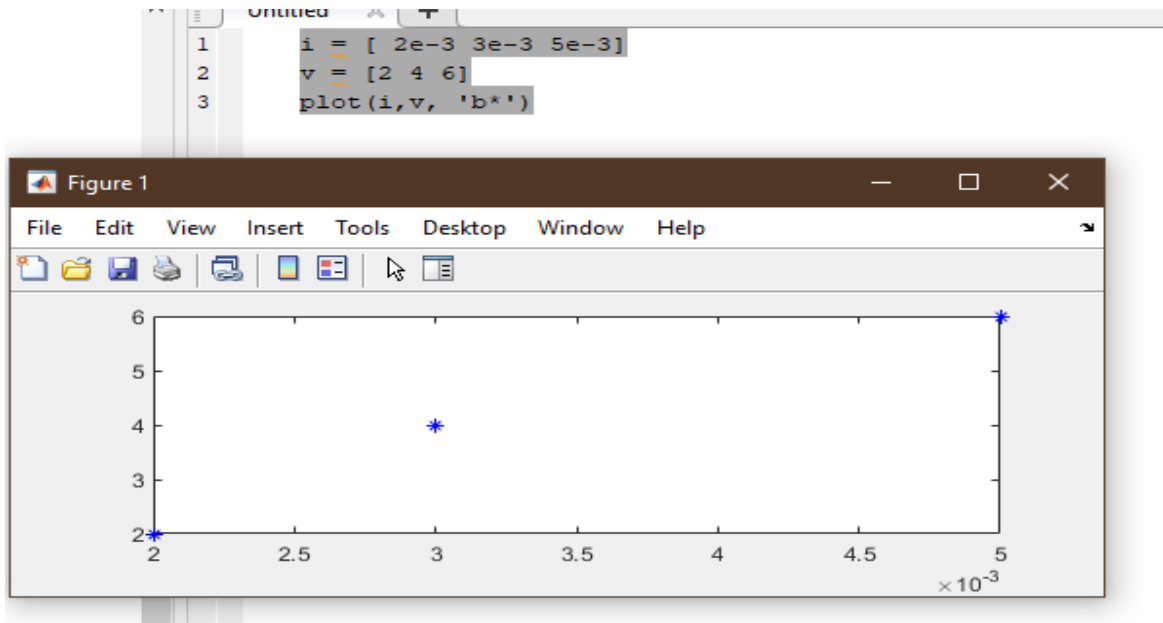
```
>> n = 10;
f = n;
while n > 1
    n = n-1;
    f = f*n;
end
disp(['n! = ' num2str(f)])
n! = 3628800
\\
```

if: ejecuta la instrucción si es verdadera

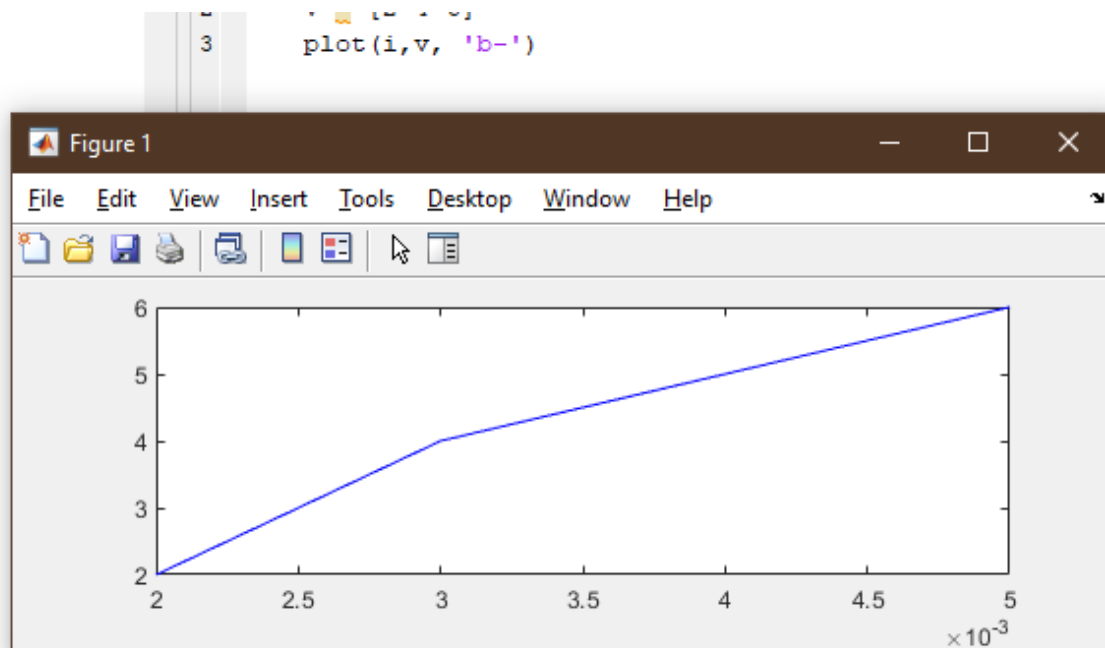
```
>> A = ones(2,3);
B = rand(3,4,5);
if isequal(size(A),size(B))
    C = [A; B];
else
    disp('A and B are not the same size.')
    C = [];
end
A and B are not the same size.
\\ |
```

GRAFICOS MATLAB

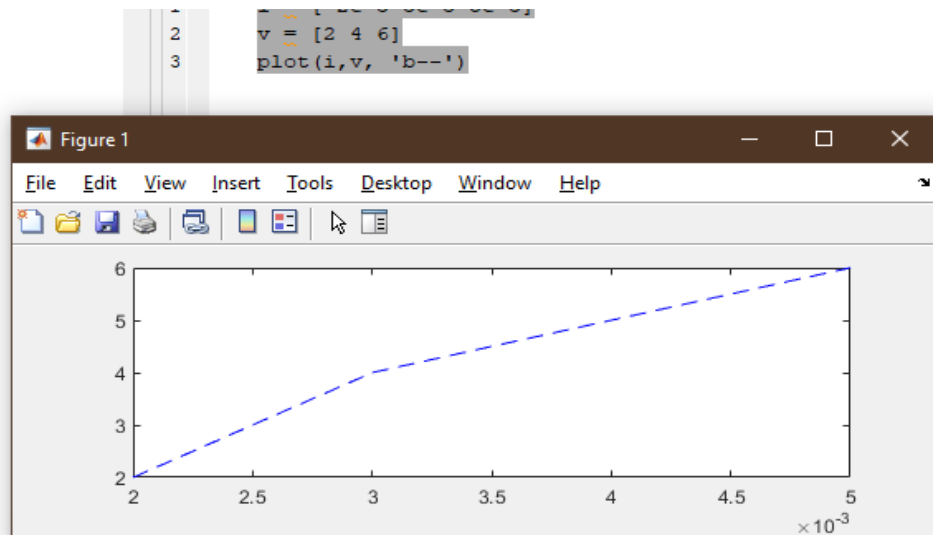
Plot: Función punto a punto. Contiene muchas características para graficar información, como colores, representaciones de valores.



Plot 'letra': Podemos cambiar el color de la línea y la forma en que esta se muestre.

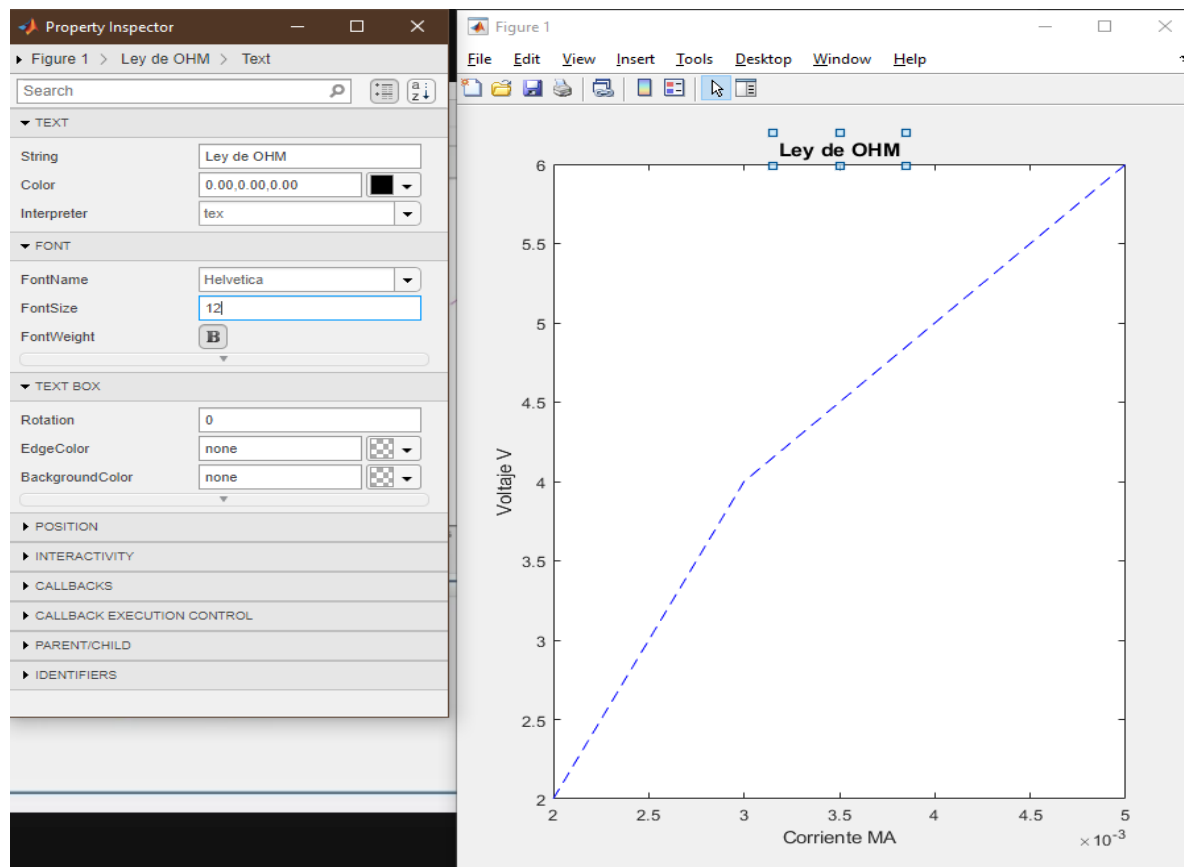


Plot + "--,*,-": Forma en que se muestra la línea

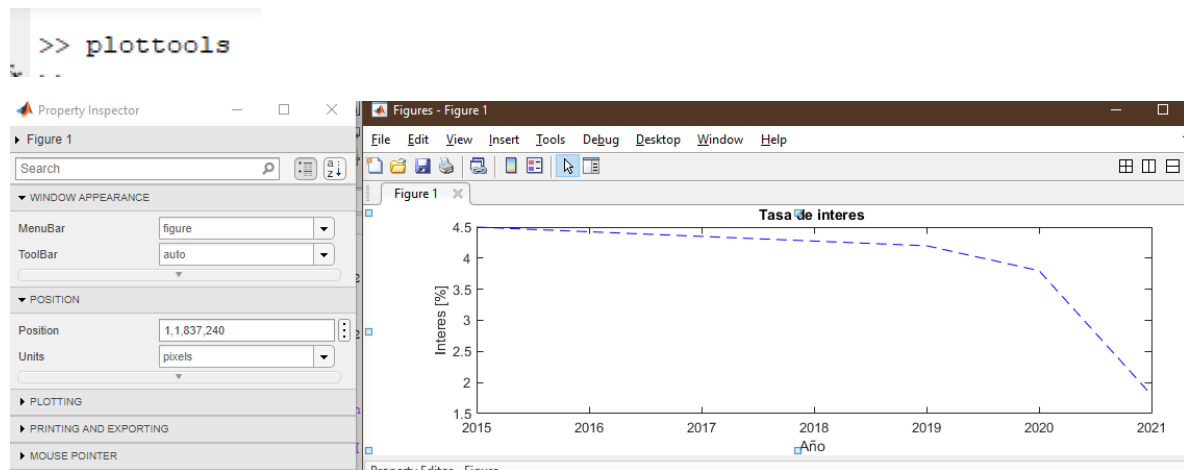


7

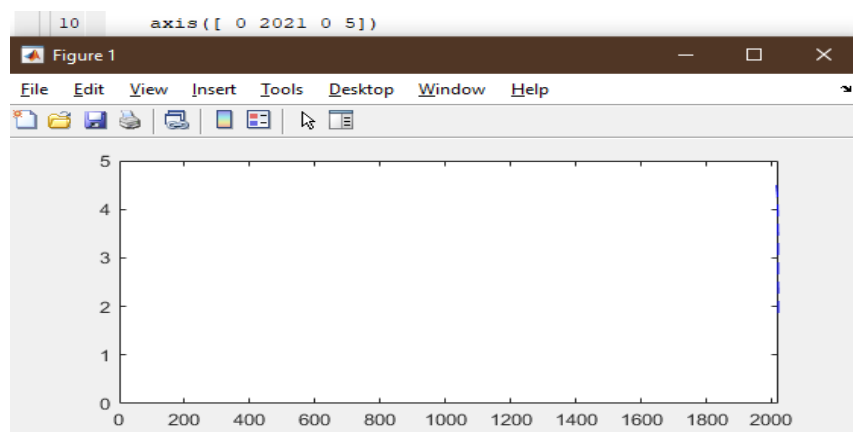
Edición: Se puede agregar el editor de gráficos de dos formas manual o por comando.



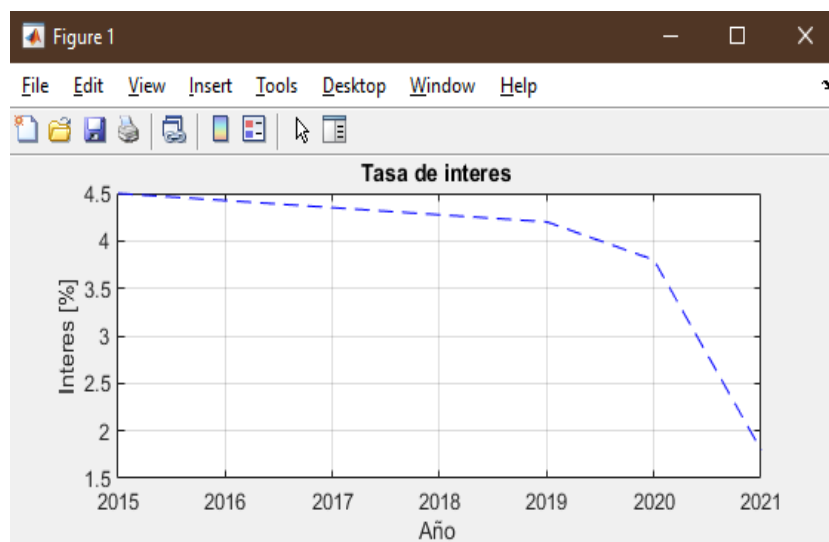
Plottools: Rápida forma para mostrar el editor de gráficos.



Axis: Establece los parámetros de los ejes X & Y



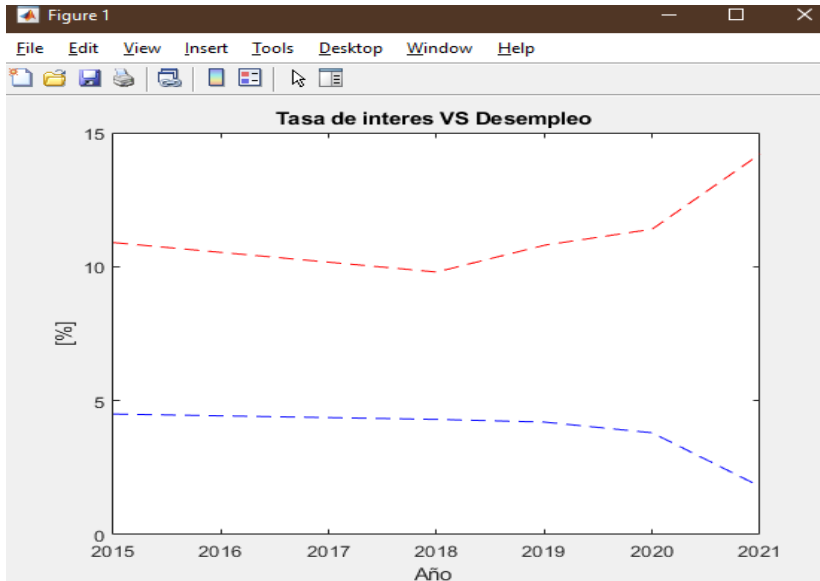
Grid: Aparece la cuadrícula en la gráfica.



Hold on: Grafique en la misma figura

```
% Desempleo  
a = [ 2019-4 2018 2019 2020 2021]  
d = [10.9 9.8 10.8 11.4 14.2]
```

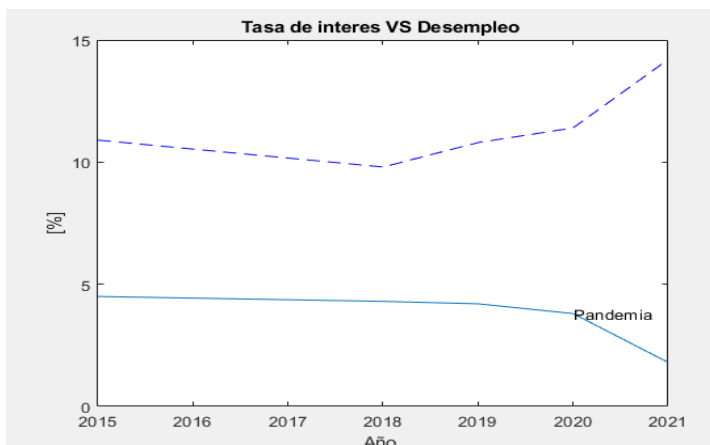
```
>> hold on  
>> plot(a,d, 'r--')  
>> ylabel('%')
```



Hold off: Quitamos de la gráfica la línea que anteriormente agregamos.

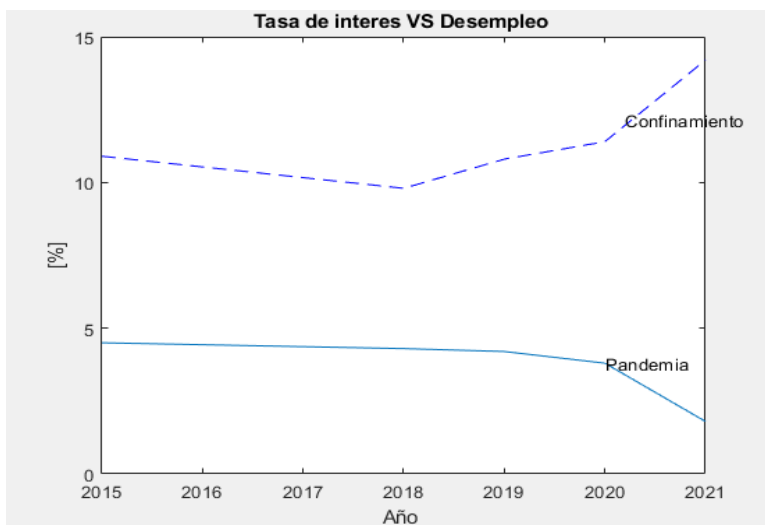
Text: Agrega datos de descripción para los puntos.

```
>> text(2020,3.8, 'Pandemia')  
^^
```



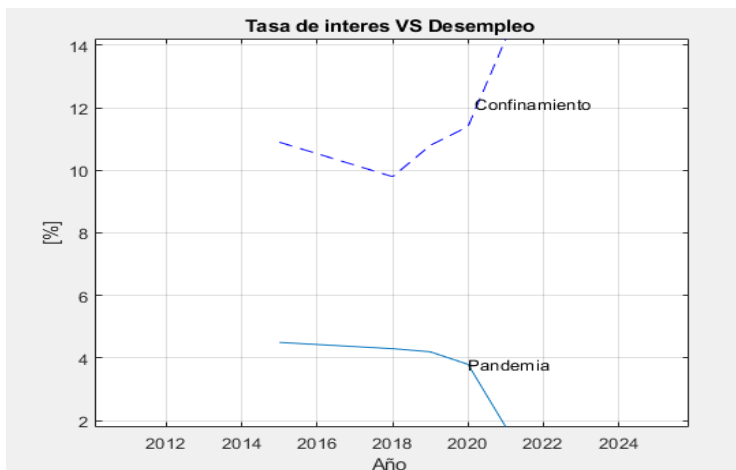
Gtext: Agrego la descripción del punto y luego la coloco en la gráfica.

```
>> gtext('Confinamiento')
```



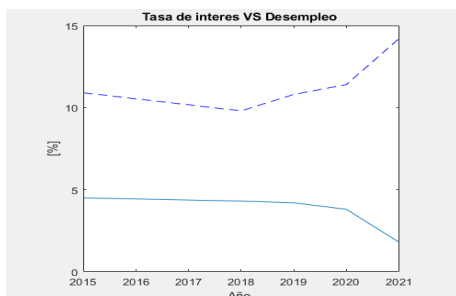
Axis equal: Muestra las escalas en los ejes que sea igual.

```
>> axis equal
```



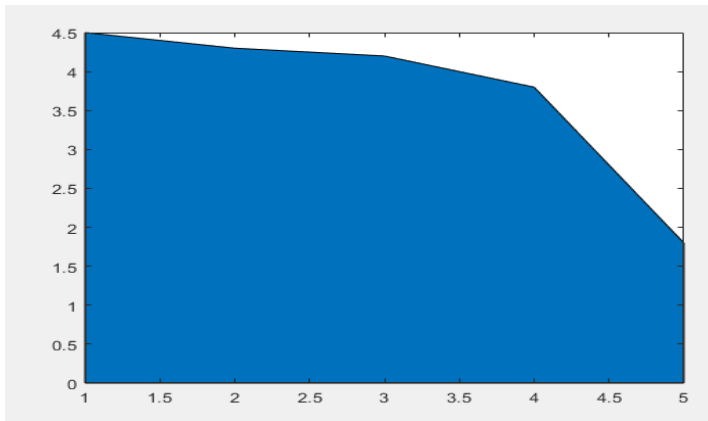
Axis square: Muestra el grafico como un cuadrado.

```
>> axis square
```



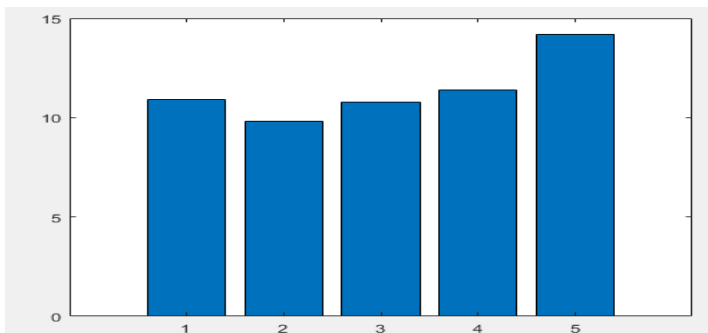
Area(Y) : Sombrea el área por debajo de la figura igualando a 0

```
>> area(v)
```

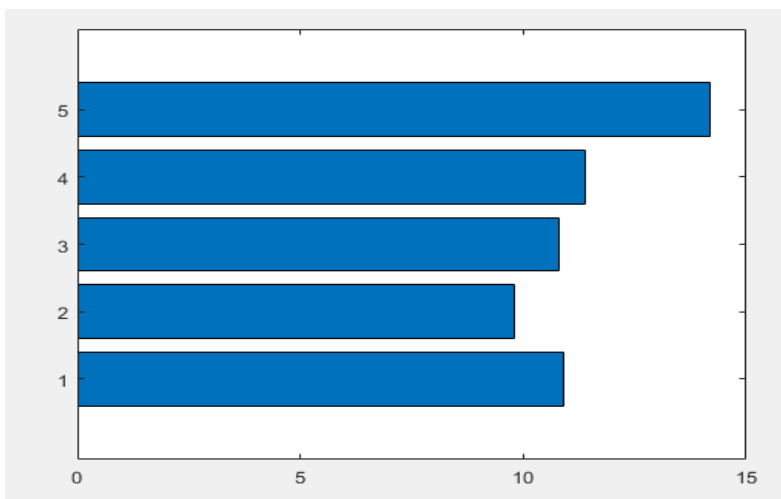


Bar: Grafico de Barras con la función.

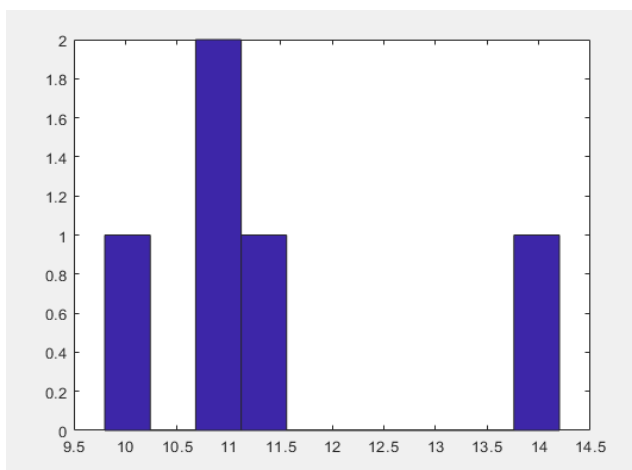
```
>> bar(d)
```



Barh: Grafico horizontal

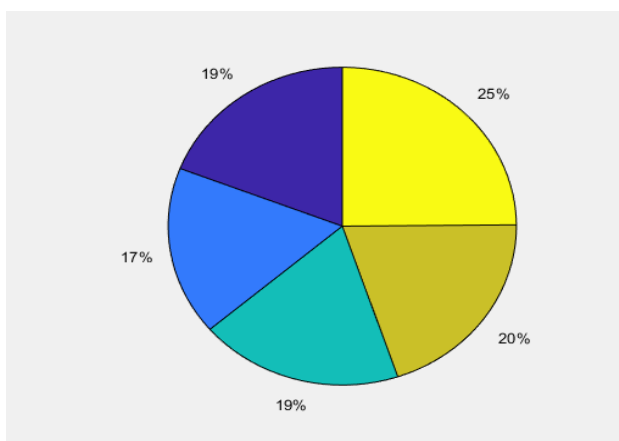


Hist: Histograma

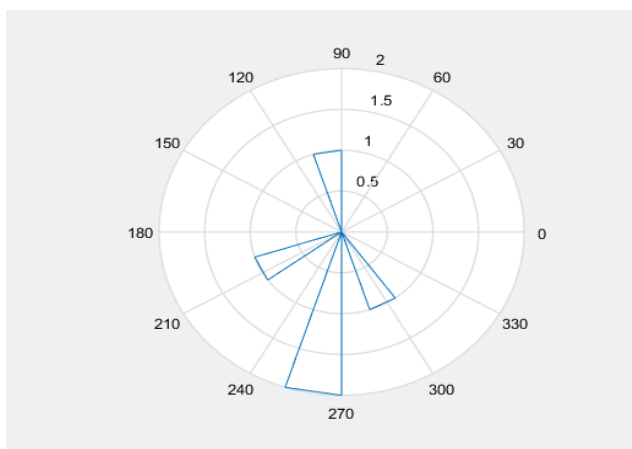


Pie: Grafico tipo Torta

```
>> pie(d)
```

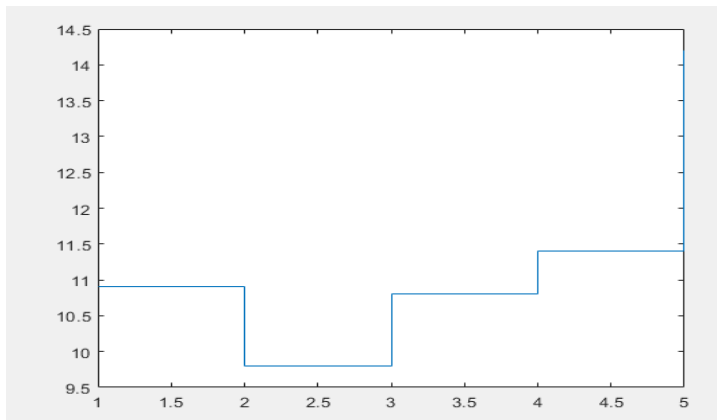


Rose: Tipo Polar.



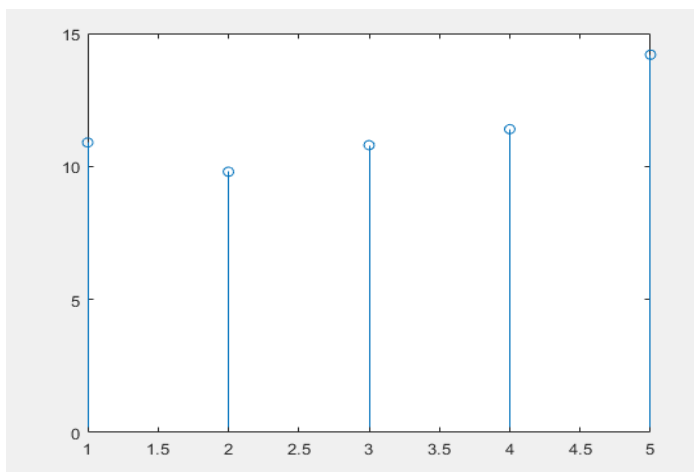
Stairs: Grafico de pasos.

```
>> stairs(d)
```



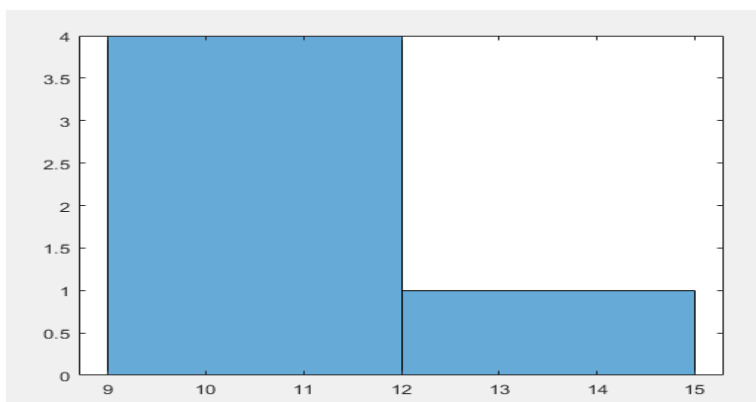
Stem: Valores Discretos.

```
>> stem(d)
```



Histogram: Histograma

```
>> histogram(d)
```



Semilogx(x): Funciones Exponenciales o logarítmicas.

Plot3: Imprimir una gráfica en 3D.

```
>> x = 112:680;  
y = sind(x);  
z = cosd(x);  
>> x = 112:680;  
y = sind(x);  
z = cosd(x);  
>> x(1)
```

```
ans =
```

```
112
```

```
>> y(1)
```

```
ans =
```

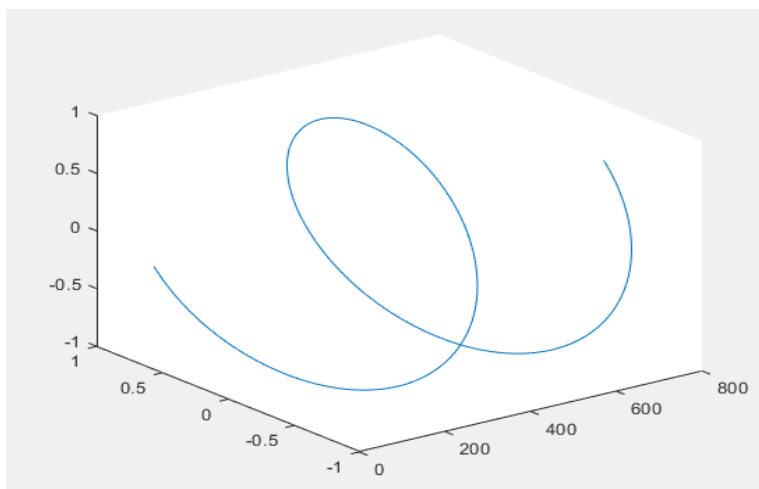
```
0.9272
```

```
>> z(1)
```

```
ans =
```

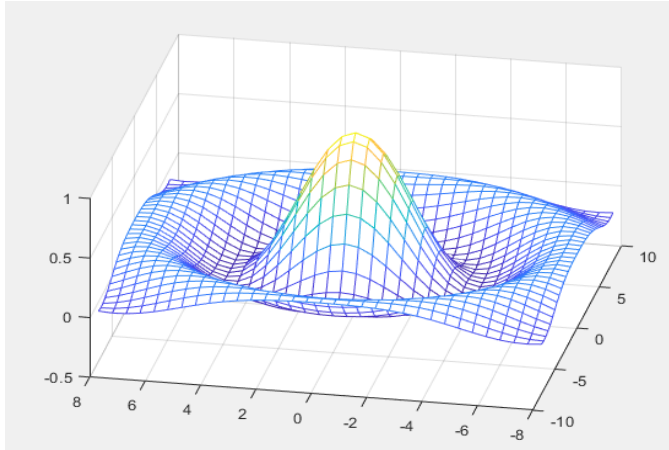
```
-0.3746
```

```
>> plot3(x,y,z)
```



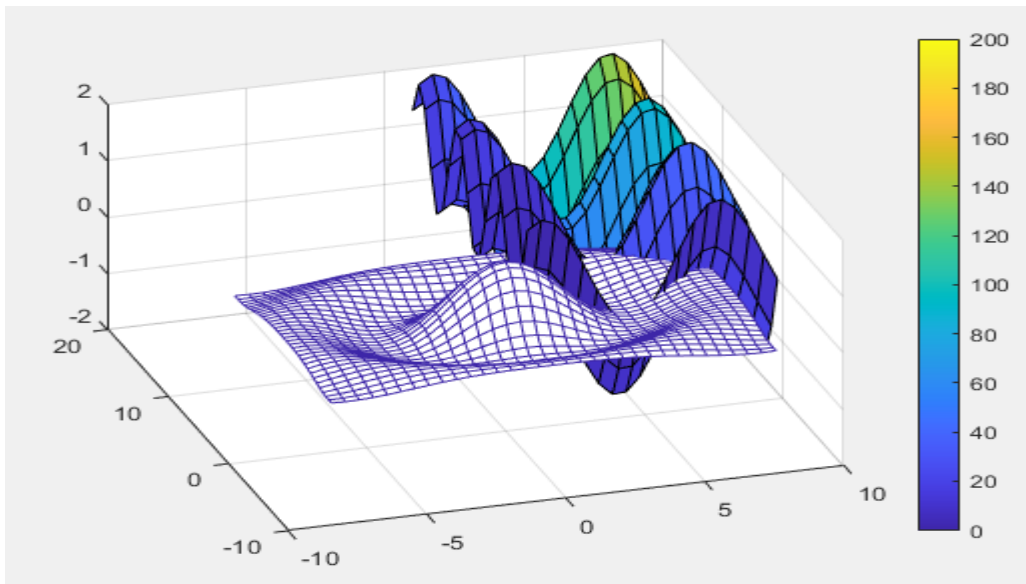
Meshgrid: Crea una matriz gráfica malla en 3D.

```
>> [X,Y] = meshgrid(-8:.5:8);  
R = sqrt(X.^2 + Y.^2) + eps;  
Z = sin(R) ./ R;  
mesh(X,Y,Z)
```



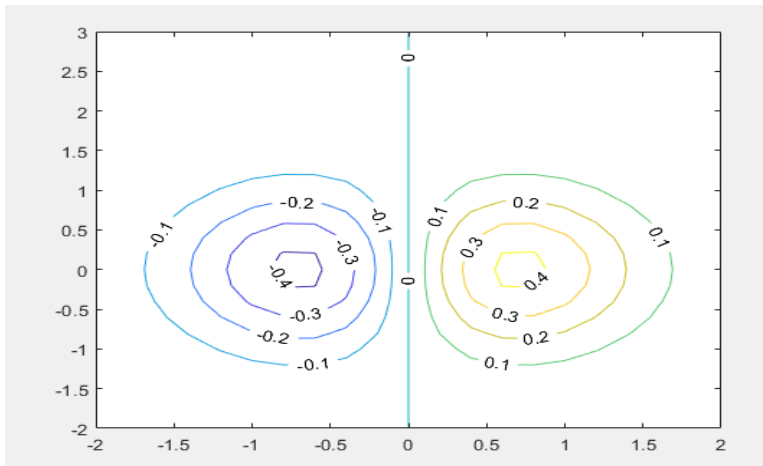
Surface: La función traza los valores de la matriz Z como alturas sobre una cuadrícula en el plano x-y definido por X e Y. El color de la superficie varía según las alturas especificadas por Z.

```
>> [X,Y] = meshgrid(1:0.5:10,1:20);  
Z = sin(X) + cos(Y);  
C = X.*Y;  
surface(X,Y,Z,C)  
colorbar  
view(3)
```



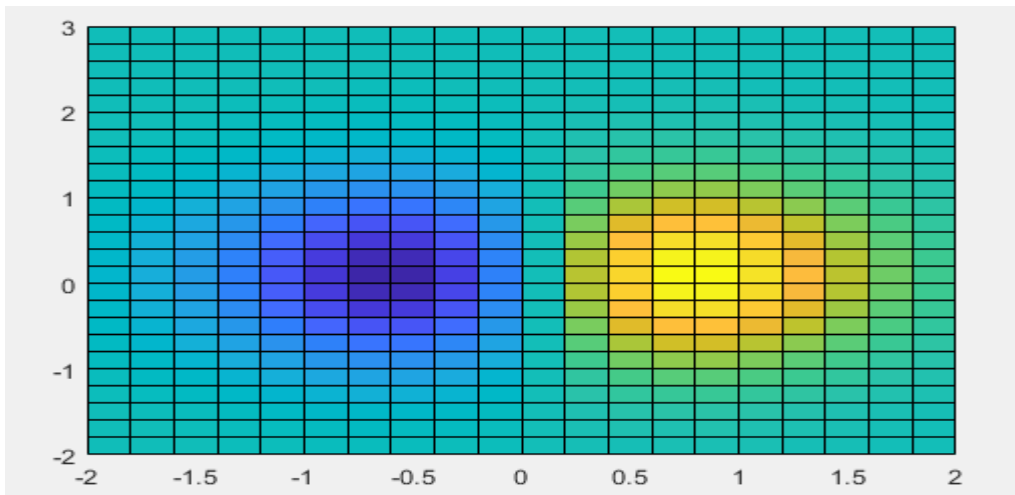
Contour: dibuja una trama de contorno de la matriz Z, donde Z se interpreta como alturas con respecto al plano x-y .

```
>> x = -2:0.2:2;  
y = -2:0.2:3;  
[X,Y] = meshgrid(x,y);  
Z = X.*exp(-X.^2-Y.^2);  
  
figure  
contour(X,Y,Z, 'ShowText', 'on')
```

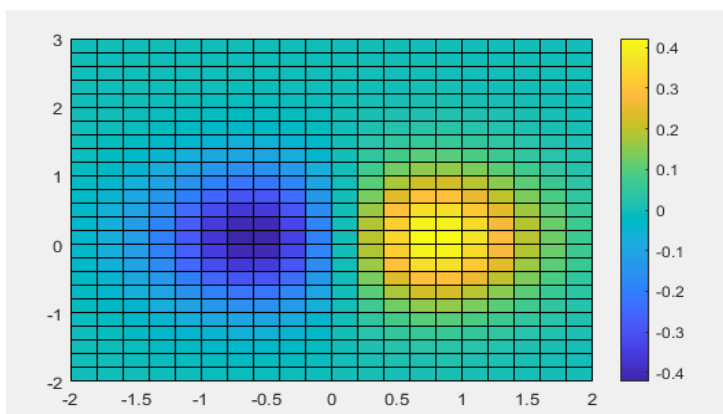


pcolor: Crea una figura de dos dimensiones con datos de tres dimensiones.

```
>> pcolor(X,Y,Z)
```

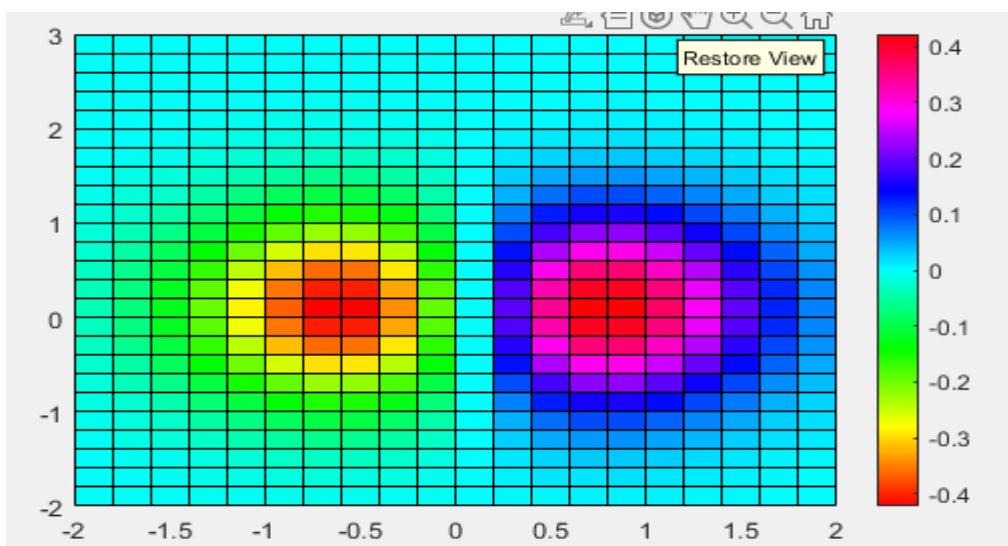


Colorbar: Muestra la barra de colores

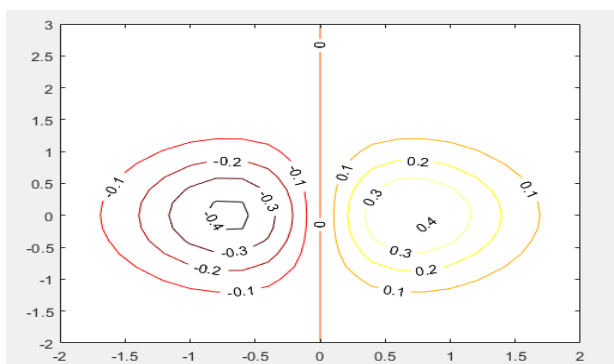


Colormap: Cambia los colores que estoy utilizando.

```
>> colormap(hsv)
```

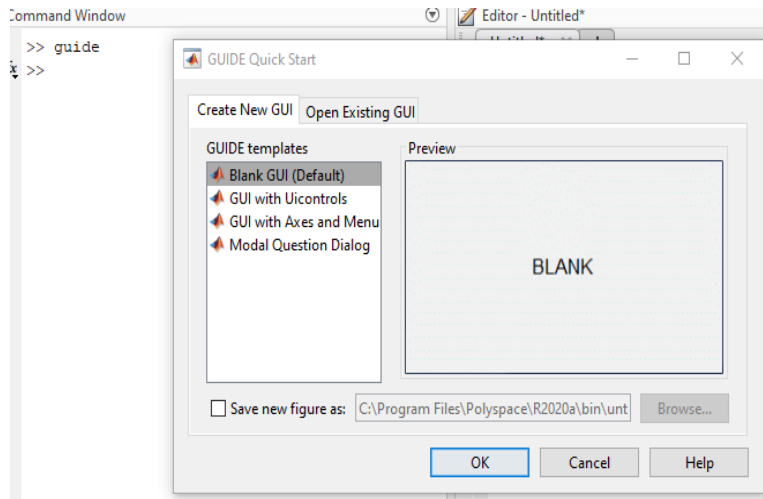


Colormap: Cambia el color a rojo amarillo

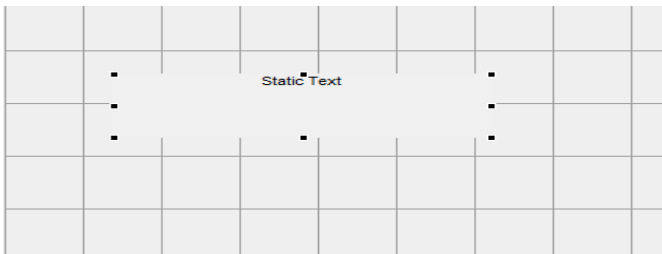


Clase Semana 8

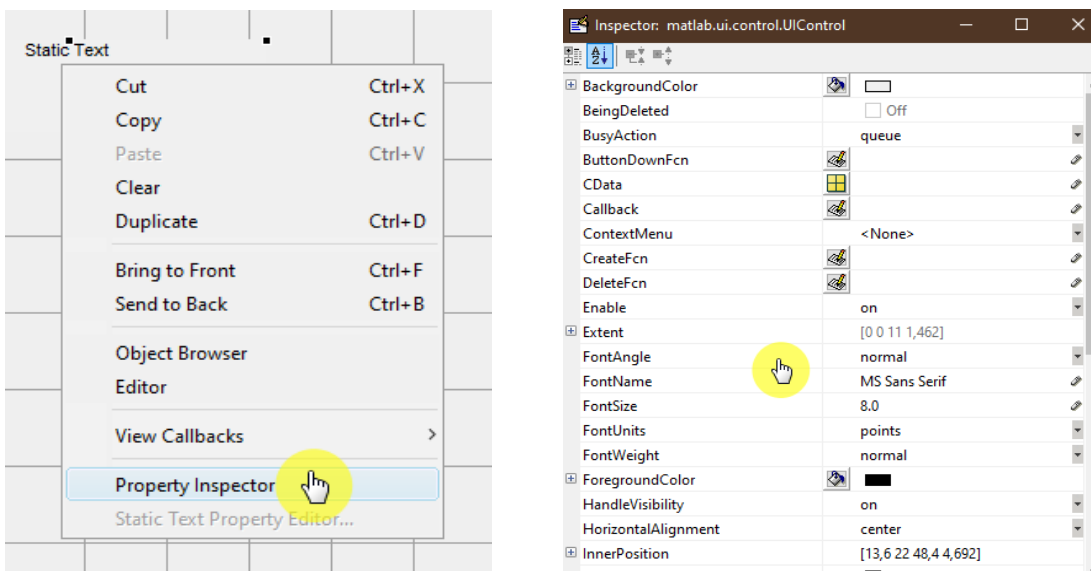
guide: Entorno de interfaz gráfica de Matlab. (GUI)



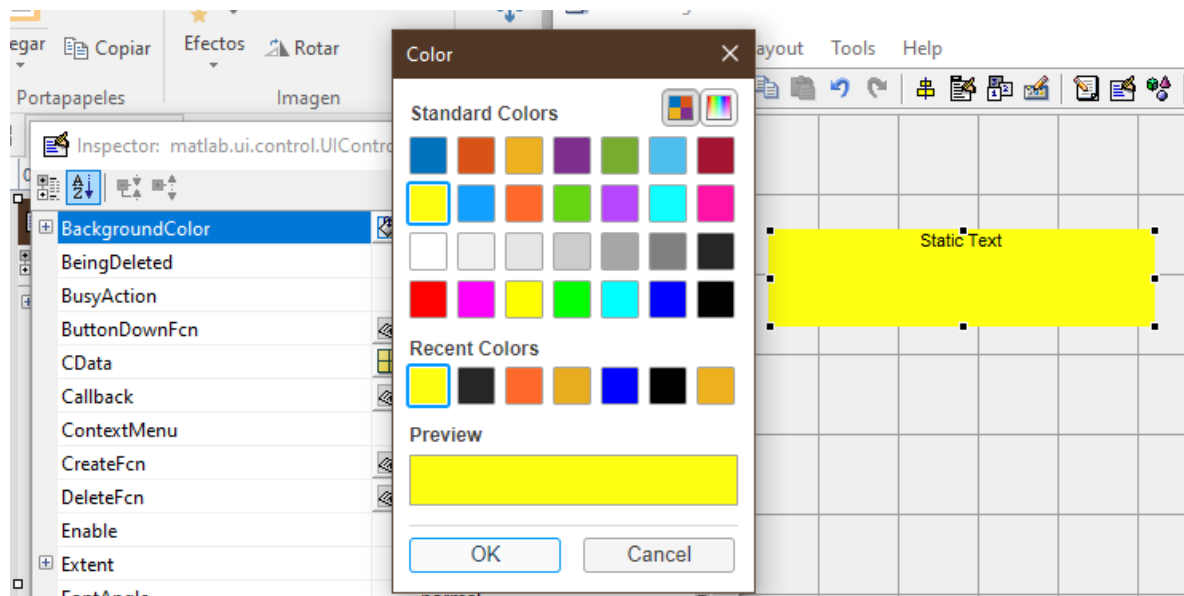
Static text: Caja de texto.



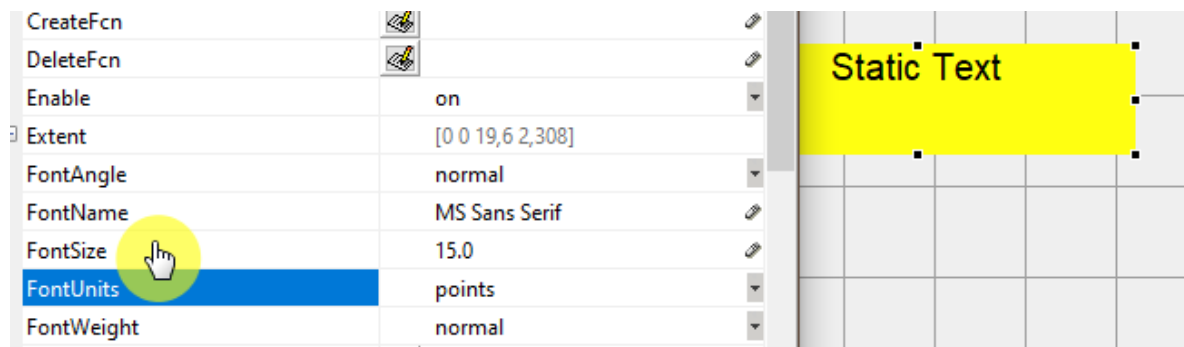
Property inspector: Vemos las propiedades de cualquier objeto.



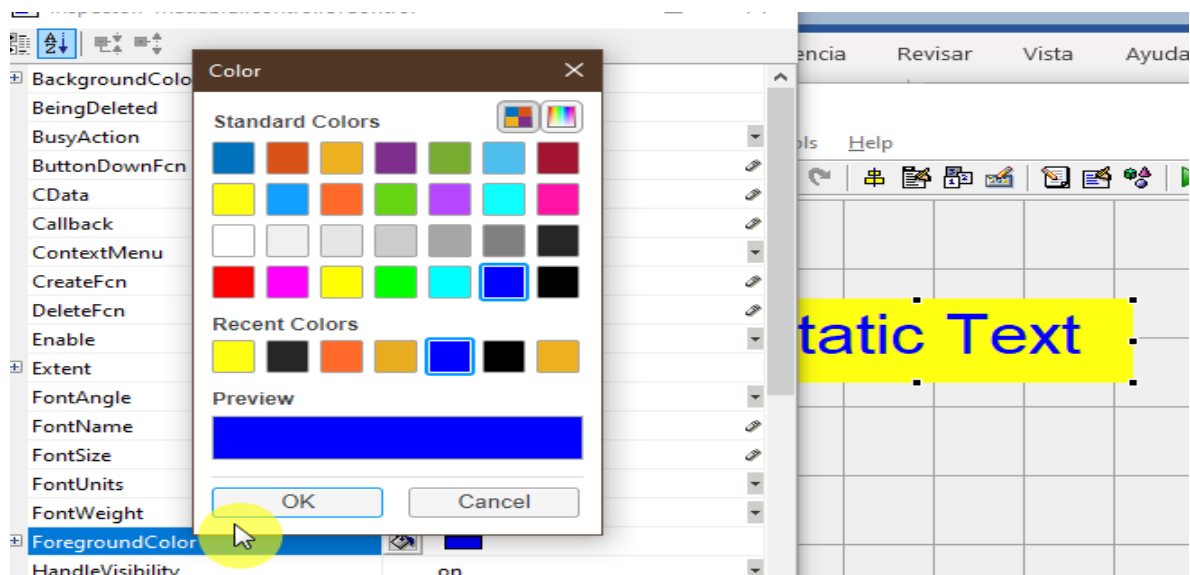
BackgroundColor: Cambia el color del fondo



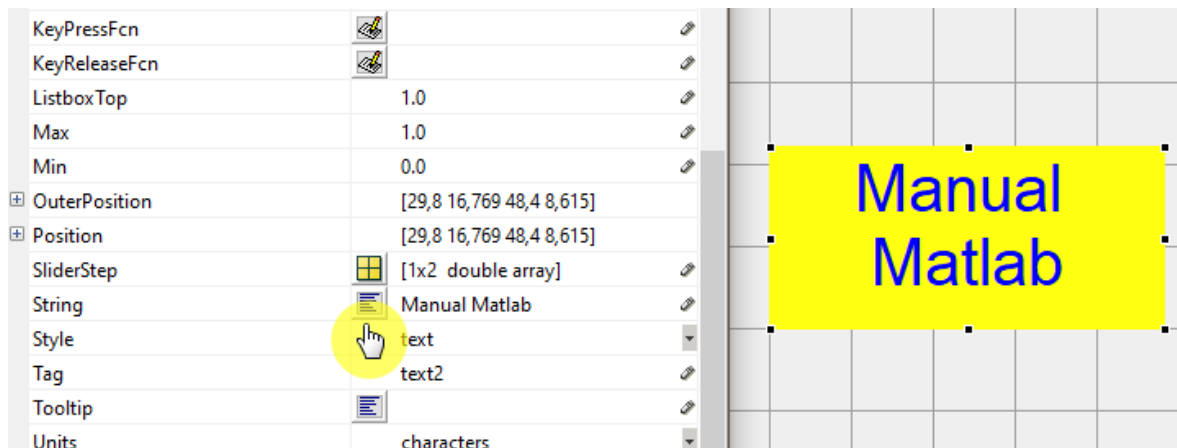
FontSize: Tamaño de letra.



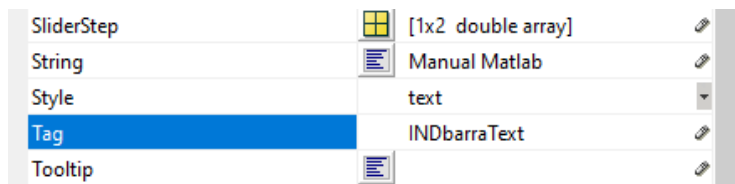
ForegroundColor: Color de la letra



String: Edito el texto



Tag: Nombre de la barra del texto.



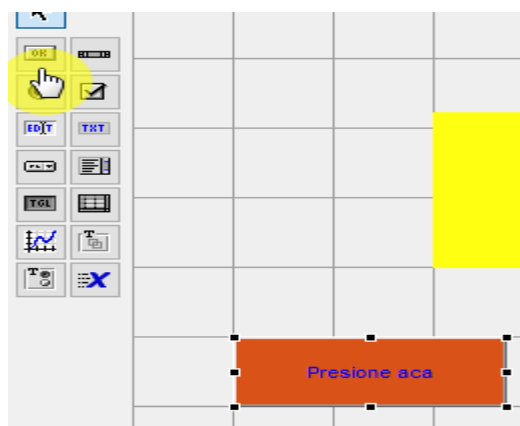
Max & Min: Propiedades de Tamaño.



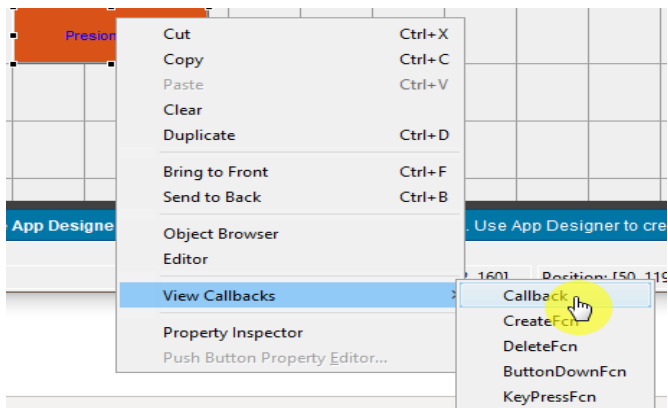
Position: Posición del recuadro de texto en la ventana de interfaz.



PushButton: Botón que cambia de evento al dar click.



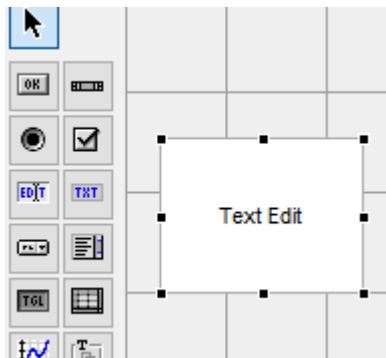
Callback: Programación para el push botón



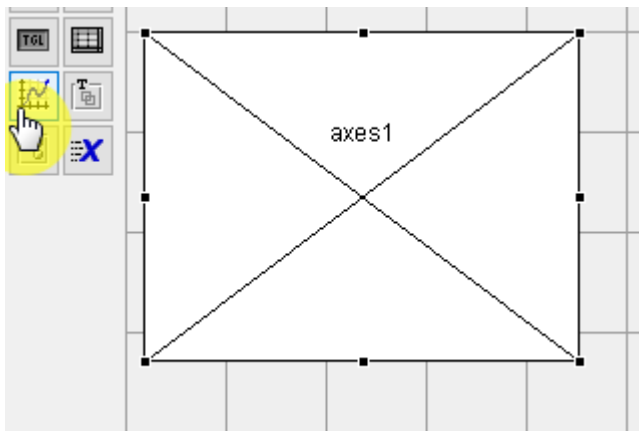
Set(): Paso por parámetros las variables que quiero que se muestren en los eventos.

```
% --- Executes on button press in pushbutton1.  
function pushbutton1_Callback(hObject, eventdata, handles)  
    mensaje = "Duvan"  
    set(handles.INDbarraText, 'String',mensaje);
```

TextEdit: Caja de texto editable



Axes : Comado para graficas.

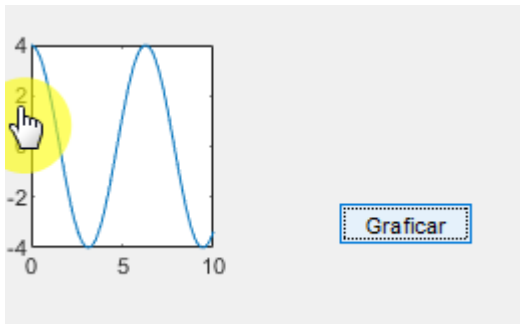


Plot handles: Graficar una función.

```
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject      handle to pushbutton1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

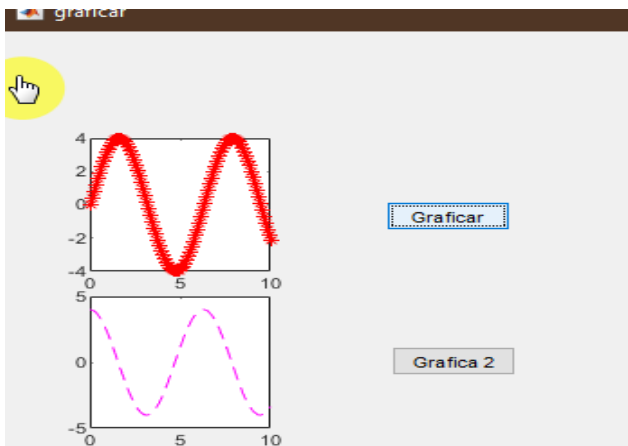
x = 0:0.1:10;
y = 4*sin(x);
z = 4*cos(x);

plot(handles.axes1,x,y)
plot(handles.axes1,x,z)
```

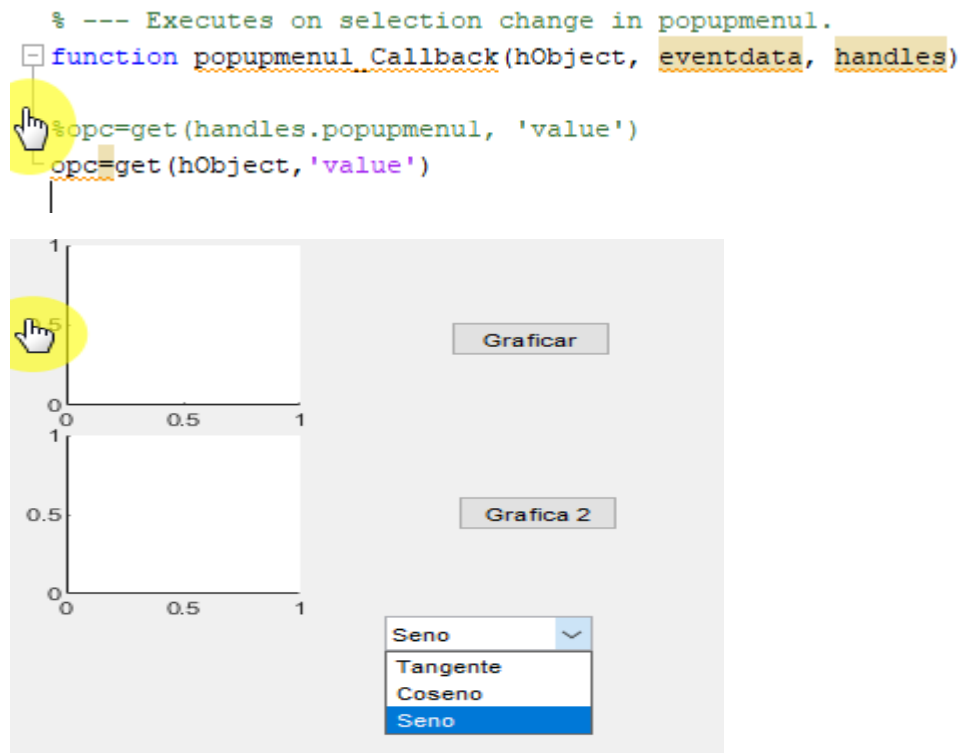


Set handles axes: Modificación rejilla de la gráfica.

```
% --- Executes on button press in Graficar2.
function Graficar2_Callback(hObject, eventdata, handles)
% hObject      handle to Graficar2 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
set(handles.axes1,'XGrid','On');
set(handles.axes1,'YGrid','On');
set(handles.axes2,'XGrid','On');
set(handles.axes2,'YGrid','On');
```



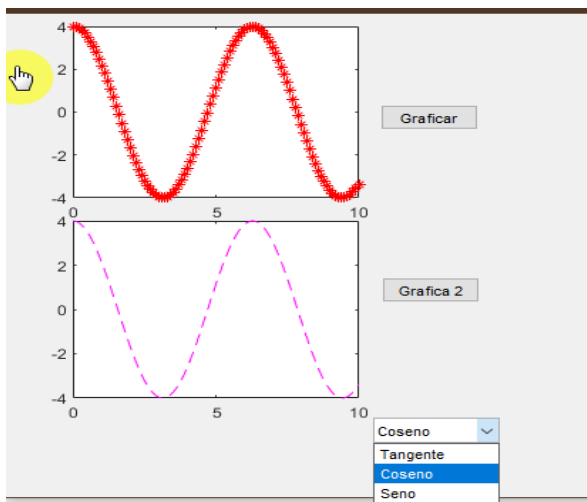
Pop-Up Menú: Es un objeto donde seleccionamos los que hay adentro.



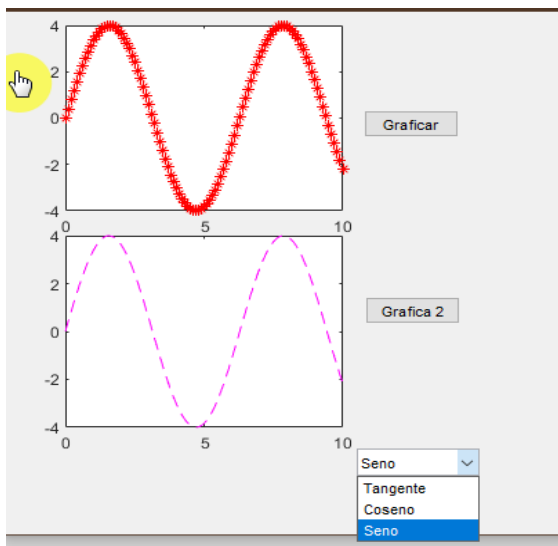
Get: Obtenemos un valor de una condición.

```
% --- Executes on selection change in popupmenu1.  
function popupmenu1_Callback(hObject, eventdata, handles)  
    %opc=get(handles.popupmenu1, 'value')  
    opc=get(hObject, 'value')  
    x = 0:0.1:10;  
    if opc == 1  
        w = 8*tan(x);  
    elseif opc == 2  
        w = 4*cos(x);  
    else  
        w = 4*sin(x);  
    end  
    plot(handles.axes1,x,w,'*r');  
    plot(handles.axes2,x,w,'--m');
```

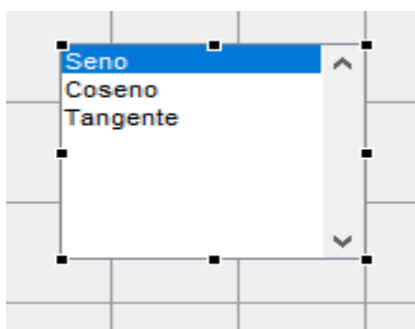
Función coseno:



Función Seno:

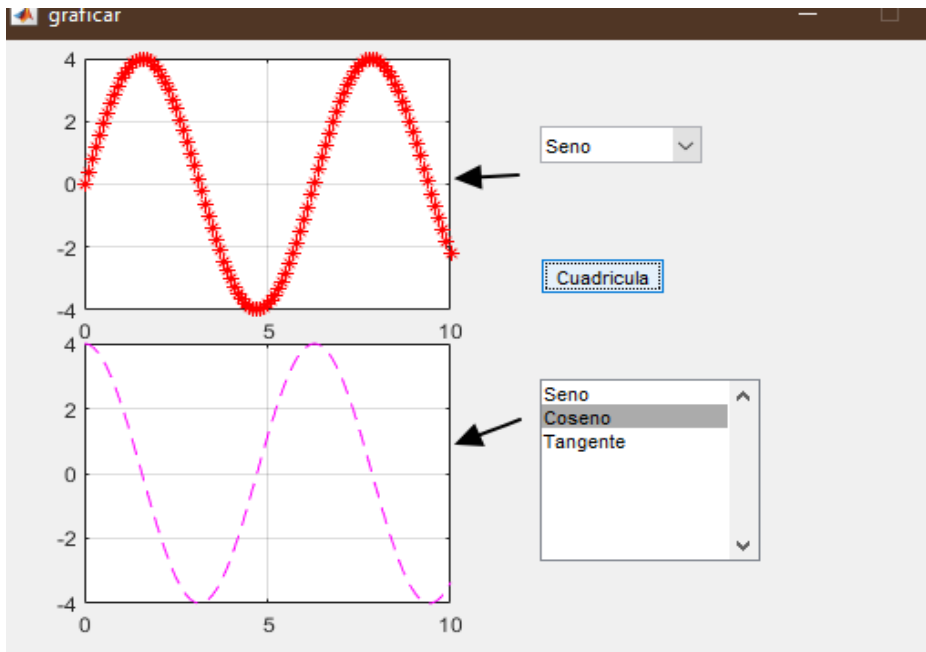


ListBox: Caja de lista



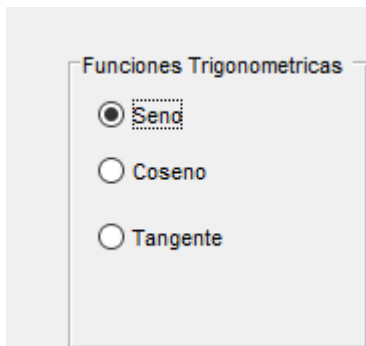
ListBox_callback: Función para graficar con caja de lista.

```
% --- Executes on selection change in listbox1.  
function listbox1_Callback(hObject, eventdata, handles)  
  
opc = get(hObject, 'Value')  
  
x = 0:0.1:10;  
if opc == 1  
    w = 4*sin(x);  
elseif opc == 2  
    w = 4*cos(x);  
else  
    w = 8*tan(x);  
end  
plot(handles.axes2,x,w, '--m');
```

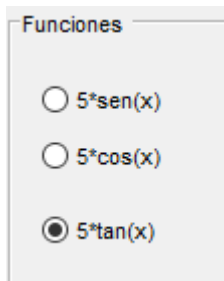


Manual 3 de febrero 2021

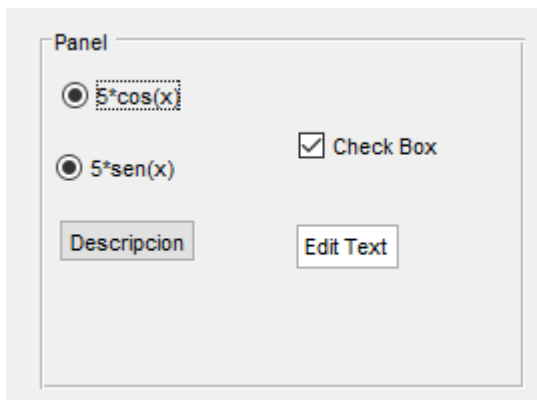
Button Group: Caja donde inserto varios componentes como radio Buttons , y solo puedo seleccionar uno solo al mismo tiempo.



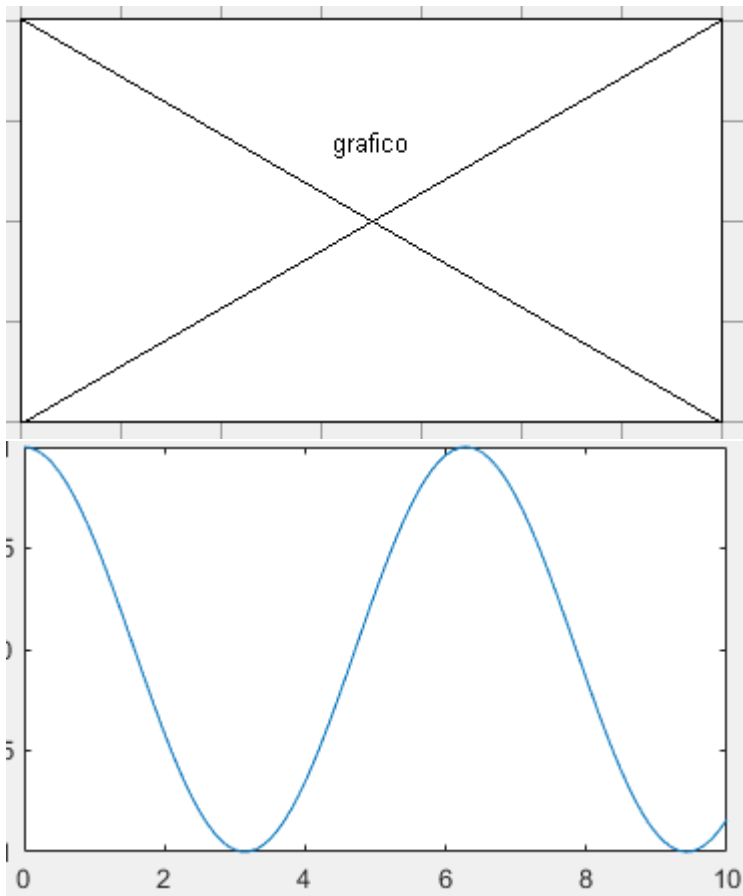
RadioButton: crea un botón de radio dentro de un grupo de botones y devuelve el objeto RadioButton.



Panel: Agrupar varias cosas al mismo tiempo



Axes: Grafico En Matlab



Function: Declarar nombre de función, entradas y salidas

```
function RBseno_Callback(hObject, eventdata, handles)
% hObject    handle to RBseno (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

hObject : Referecia al objeto donde estoy llamando a la función.

```
% Hint: get(hObject,'Value') returns toggle state of RBseno

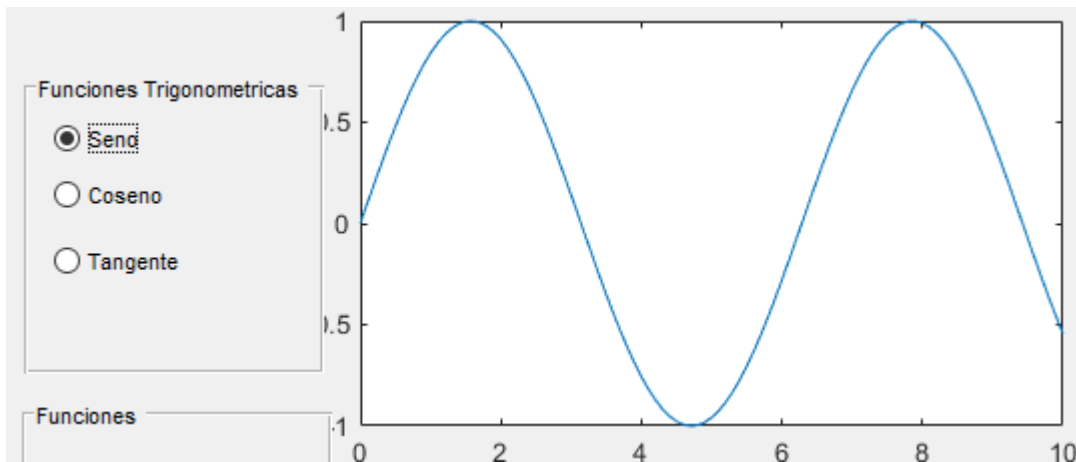
v =get(hObject, 'Value');
```

If else: Ejecutar instrucciones si la condición es verdadera

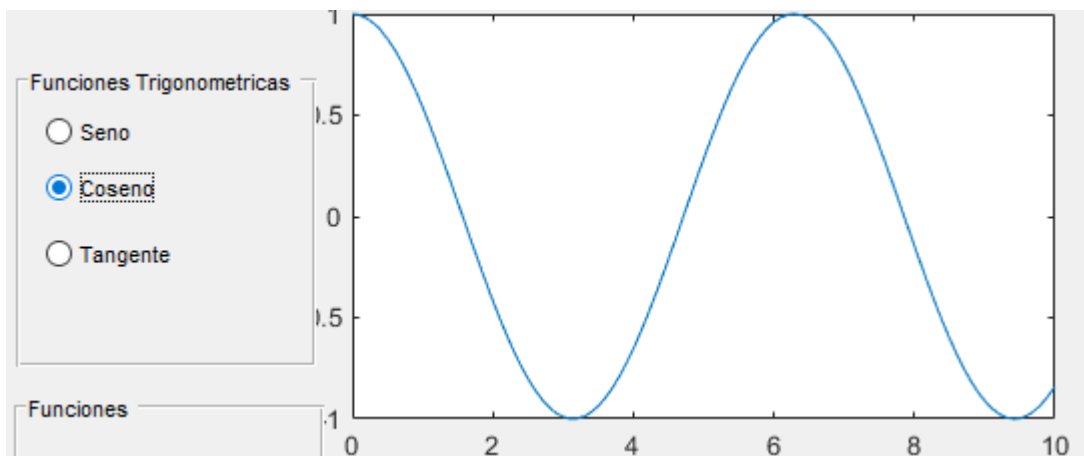
```
if v ==1
    plot(handles.grafico,x,y)
else
    cla(handles.grafico,'reset')
end
```


Reset: Función para limpiar el grafico al seleccionar otra función.

Función seno



Click función coseno

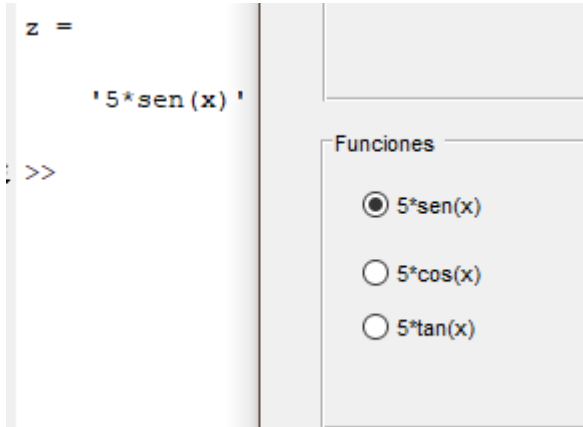


Handles: es un tipo de datos de MATLAB que representa una función: El objetivo es pasar de una función a otra

```
v = get(hObject, 'Value');  
x = 0:0.1:10;  
y = sin(x)  
if v == 1  
    plot(handles.grafico,x,y)  
else  
    cla(handles.grafico,'reset')  
end
```

Get String : Capturo el string nombre de la variable

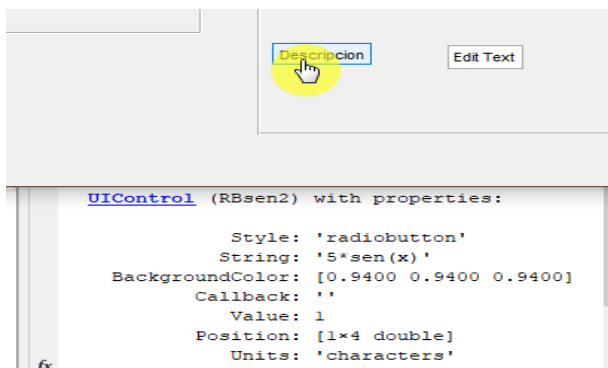
```
z = get(hObject, 'String')  
x=0:0.1:10;
```



Switch : evalúa una expresión y elige ejecutar uno de varios grupos de instrucciones. Cada elección es un caso.

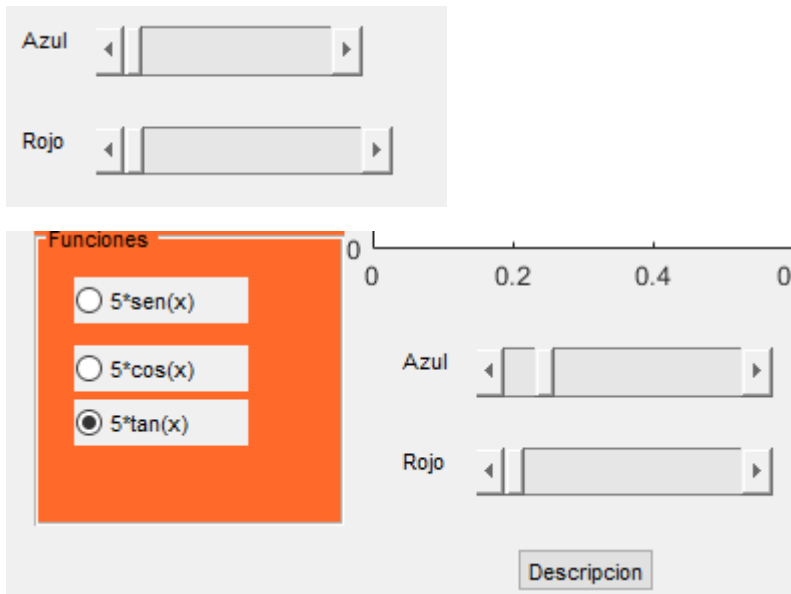
```
switch z  
case '5*sen(x)'  
    y=5*sin(x);  
case '5*cos(x)'  
    y=5*cos(x);  
otherwise '5*tan(x)'  
    y=5*tan(x);  
end  
plot(handles.grafico,x,y)
```

SelectObject : Trae toda las propiedades del grupo seleccionado. Funciona llamando desde un objeto externo. En este caso desde un botón .



```
z = get(handles.uibuttongroup2, 'SelectedObject')
```

Slider : ajusta el valor del parámetro del bloque conectado durante la simulación



```
% --- Executes on button press in PBgraficar.
function PBgraficar_Callback(hObject, eventdata, handles)
% hObject    handle to PBgraficar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
R = get(handles.Azul, 'Value')
G = get(handles.Rojo, 'Value')
set(handles.uibuttongroup2, 'BackgroundColor', [R G])
```