ANÁLISIS DE SEÑALES

CURSO 2022 – PROF. JORGE RUNCO

SCRIPTS Y FUNCIONES

- Los programas en Matlab/Octave se escriben como scripts o funciones.
- Scripts es un archivo .m que tiene una secuencia de sentencias en Matlab.
- Una función también es un archivo .m, pero es llamado desde un scripts (o también >>).
- Las variables definidas en la función, tienen alcance local (solo conocidas en esa función).

FUNCIONES

- El usuario de Matlab/Octave puede definir sus propias funciones o subrutinas y asignarle el nombre que quiera con la misma limitación que se tiene para nombrar un archivo.
- Esto es así porque de hecho definir una función propia consiste sencillamente en la creación de un archivo m que tiene por nombre el mismo nombre que el de la función.

FUNCIONES

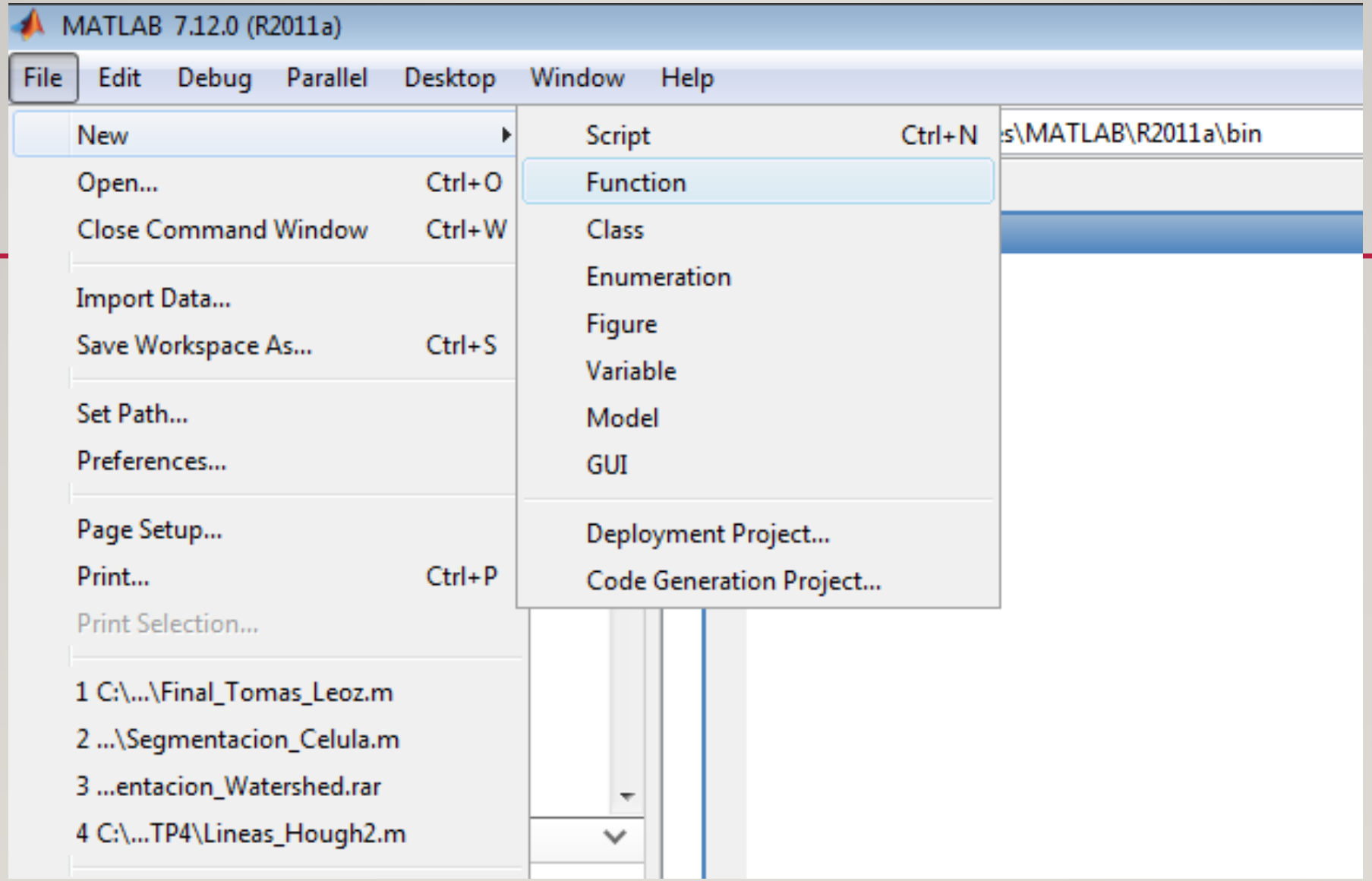
- Una función(habitualmente denominadas M-funciones en Matlab/Octave), es un programa con una "interfase" de comunicación con el exterior mediante argumentos de entrada y de salida.
- Las funciones Matlab/Octave responden al siguiente formato de escritura. La cláusula end del final no es obligatoria, excepto en el caso de funciones anidadas.

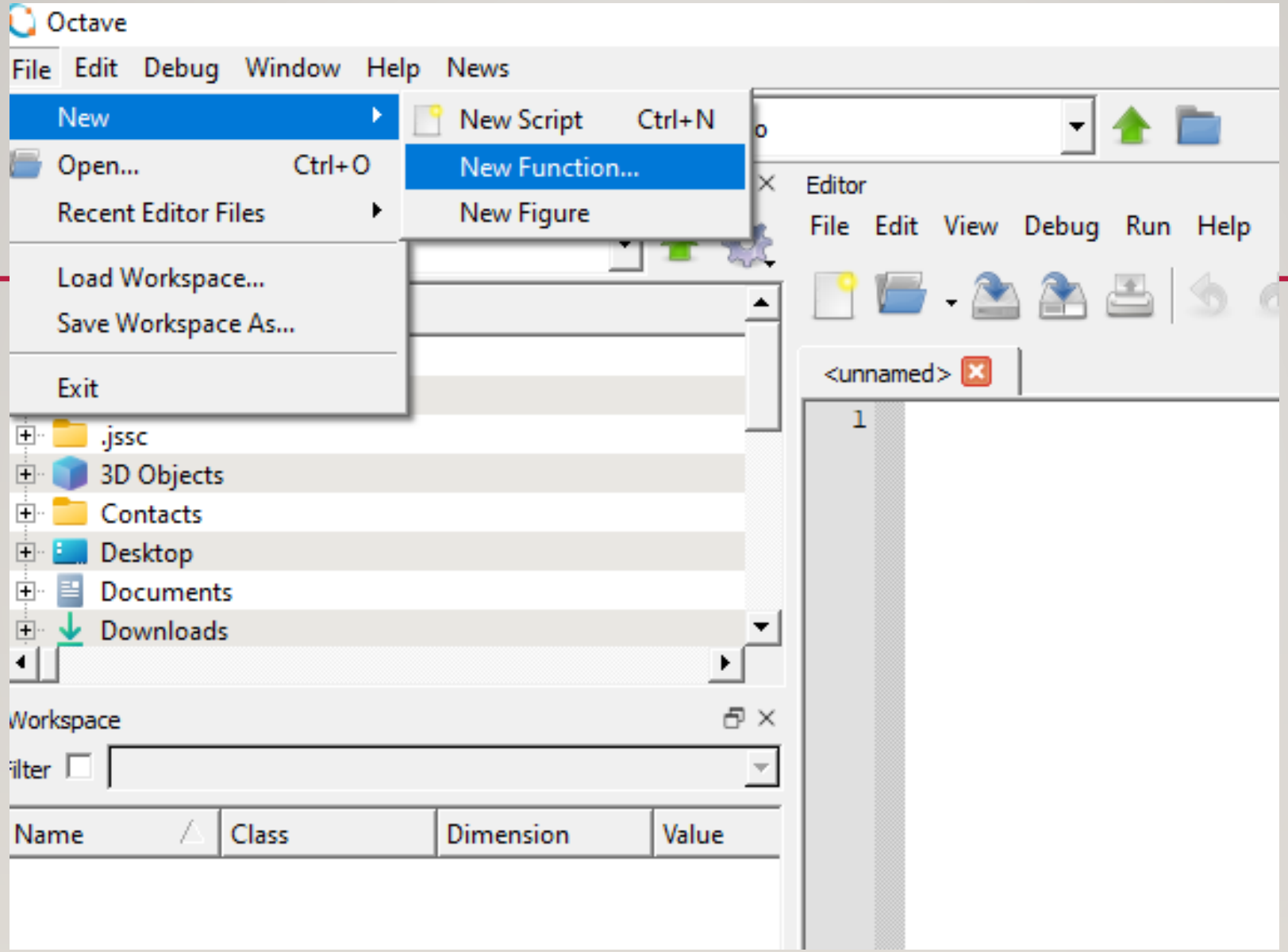
FUNCIONES DEFINIDAS POR EL PROGRAMADOR

- La primera línea empieza con la palabra function y tiene la forma

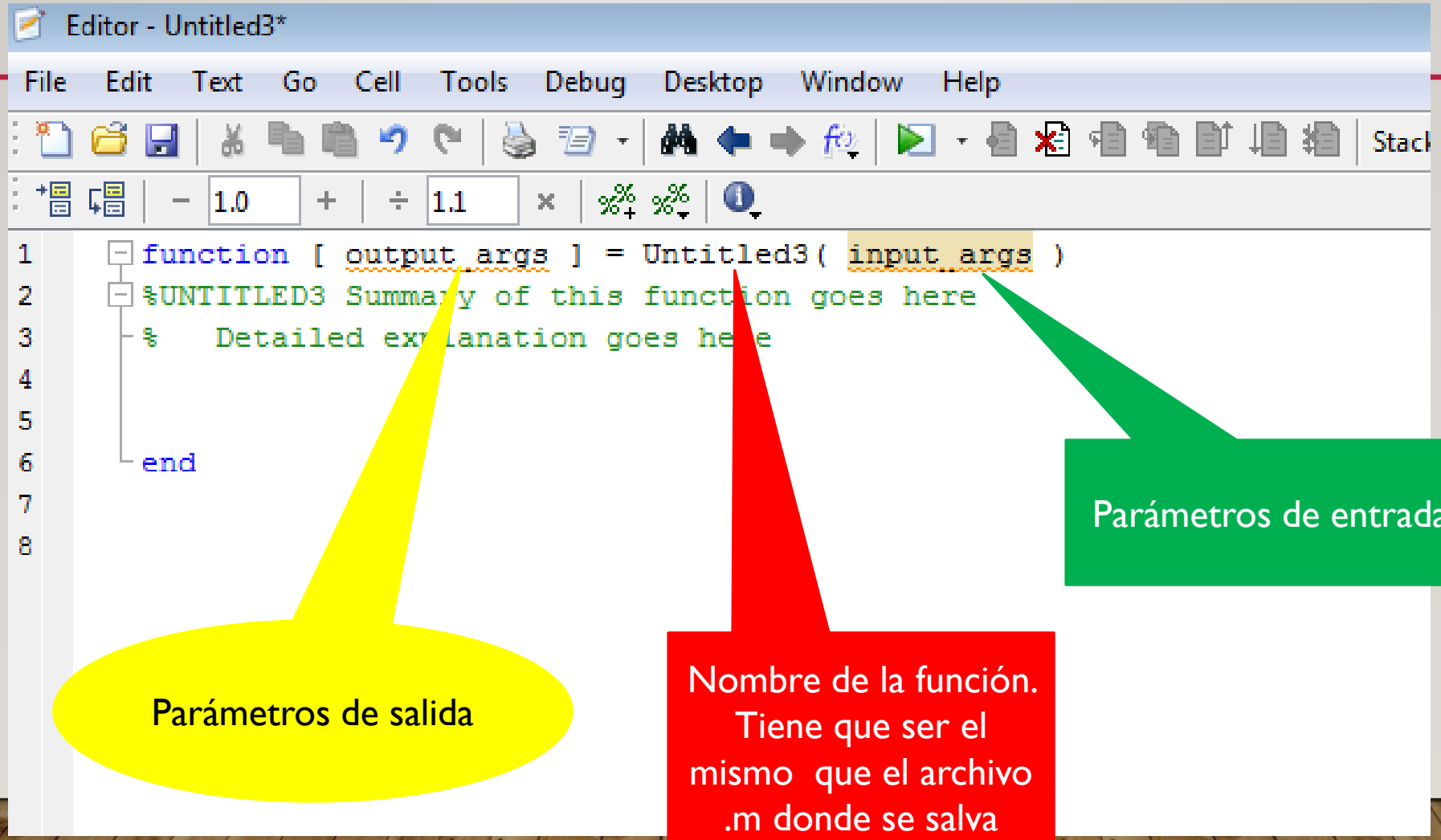
```
function [arg_salida]=nombre_funcion(arg_entrada)
function [out1, out2,…]=mifun(in1, in2, ….)
                sentencias
end
```

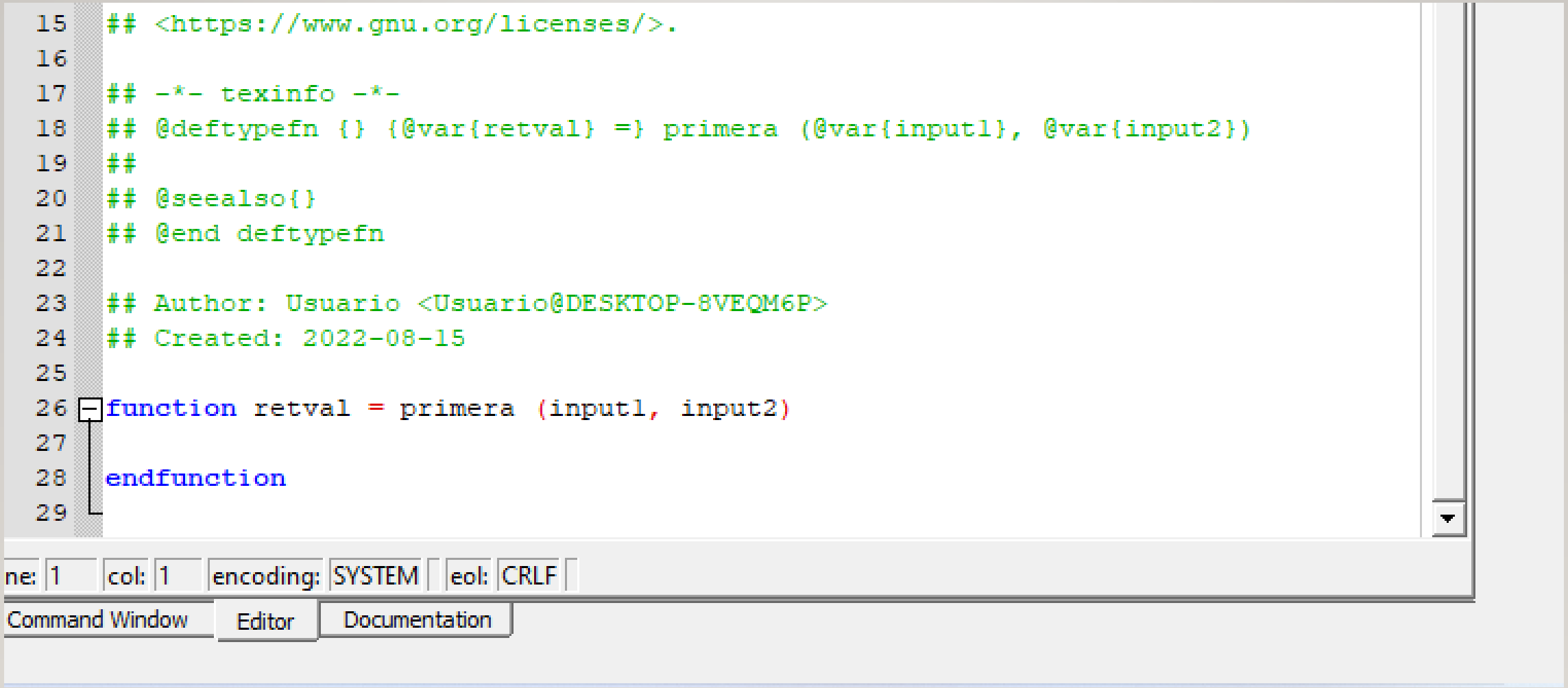
- El archivo se debe guardar con nombre mifun.m


MATLAB 7.12.0 (R2011a)
File
Edit
Debug
Parallel
Desktop
Window
Help
New
Open... Ctrl+O
Close Command Window Ctrl+W
Import Data...
Save Workspace As... Ctrl+S
Set Path...
Preferences...
Page Setup...
Print... Ctrl+P
Print Selection...
1 C:\...\Final_Tomas_Leoz.m
2 ...\Segmentacion_Celula.m
3 ...entacion_Watershed.rar
4 C:\...TP4\Lineas_Hough2.m
Script Ctrl+N
Function
Class
Enumeration
Figure
Variable
Model
GUI
Deployment Project...
Code Generation Project...
s\MATLAB\R2011a\bin


Octave
File Edit Debug Window Help News
New
Open... Ctrl+O
Recent Editor Files
Load Workspace...
Save Workspace As...
Exit
New Script Ctrl+N
New Function...
New Figure
Editor
File Edit View Debug Run Help
<unnamed>
1
.jssc
3D Objects
Contacts
Desktop
Documents
Downloads
Workspace
Filter
Name
Class
Dimension
Value

Editor para generar una función. Genera el esqueleto, hay que completar con las sentencias, argumentos de entrada y salida, nombre de la función.




15 ## <https://www.gnu.org/licenses/>.
16
17 ## -*- texinfo -*-
18 ## @deftypefn {} {@var{retval} =} primera (@var{input1}, @var{input2})
19 ##
20 ## @seealso{}
21 ## @end deftypefn
22
23 ## Author: Usuario <Usuario@DESKTOP-8VEQM6P>
24 ## Created: 2022-08-15
25
26 function retval = primera (input1, input2)
27
28 endfunction
29
ne: 1 col: 1 encoding: SYSTEM eol: CRLF
Command Window
Editor
Documentation

FUNCIONES ELEMENTALES DEFINIDAS

sqrt(x)	raiz cuadrada	sin(x)	seno (radianes)
abs(x)	módulo	cos(x)	coseno (radianes)
conj(z)	complejo conjugado	tan(z)	tangente (radianes)
real(z)	parte real	cotg(x)	cotangente (radianes)
imag(z)	parte imaginaria	asin(x)	arcoseno
exp(x)	exponencial	acos(x)	arcocoseno
log(x)	logaritmo natural	atan(x)	arcotangente
log10(x)	logaritmo decimal	cosh(x)	cos. hiperbólico
rat(x)	aprox. racional	sinh(x)	seno hiperbólico
mod(x,y) rem(x,y)	resto de dividir x por y . Iguales si x,y>0 . Ver help para definición exacta	tanh(x)	tangente hiperbólica
fix(x)	Redondeo hacia 0	acosh(x)	arcocoseno hiperb.
ceil(x)	Redondeo hacia + infinito	asinh(x)	arcoseno hiperb.
floor(x)	Redondeo hacia - infinito	atanh(x)	arcotangente hiperb.
round(x)	Redondeo al entero más próximo		

FUNCIONES PARA GRAFICAR 2D

- plot() crea un gráfico a partir de vectores y/o columnas de matrices, con escalas lineales sobre ambos ejes.
- loglog() ídem con escala logarítmica en ambos ejes.
- semilogx() ídem con escala lineal en el eje de ordenadas y logarítmica en el eje de abscisas.
- semilogy() ídem con escala lineal en el eje de abscisas y logarítmica en el eje de ordenadas.

- Existen funciones orientadas a añadir títulos al gráfico, a los ejes, a dibujar una cuadrícula auxiliar, a introducir texto, etc.
- title('título') añade un título al dibujo
- xlabel('tal') añade una etiqueta al eje de abscisas. Con xlabel off desaparece
- ylabel('cual') idem al eje de ordenadas. Con ylabel off desaparece
- text(x,y,'texto') introduce 'texto' en el lugar especificado por las coordenadas x e y. Si x e y son vectores, el texto se repite por cada par de elementos.
- grid activa una cuadrícula en el dibujo. Con grid off desaparece la cuadrícula

plot(X,Y,'opción') (opción: permite elegir color y trazo de la curva)

y	yellow	.	point	-	solid
m	magenta	o	circle	:	dotted
c	cyan	x	x-mark	-.	dashdot
r	red	+	plus	--	dashed
g	green	*	star		
b	blue	s	square		
w	white	d	diamond		
k	black	v	triangle (down)		
		^	triangle (up)		
		<	triangle (left)		
		>	triangle (right)		

EJEMPLO

Sea la función:

$$f(x) = \begin{cases} x^2 & si\ x < 0 \\ 1 & si\ 0 \le x < 1 \\ -x + 2 & si\ x \ge 1 \end{cases}$$

Vamos a representarla en el intervalo (-2, +3)

```
x = linspace(-2, 3, 3000);
y = (x.^2).*(x<0)+1.*((0<=x)&(x<1))+(-x+2).*(x>=1);
plot(x,y,'.'), grid on
```

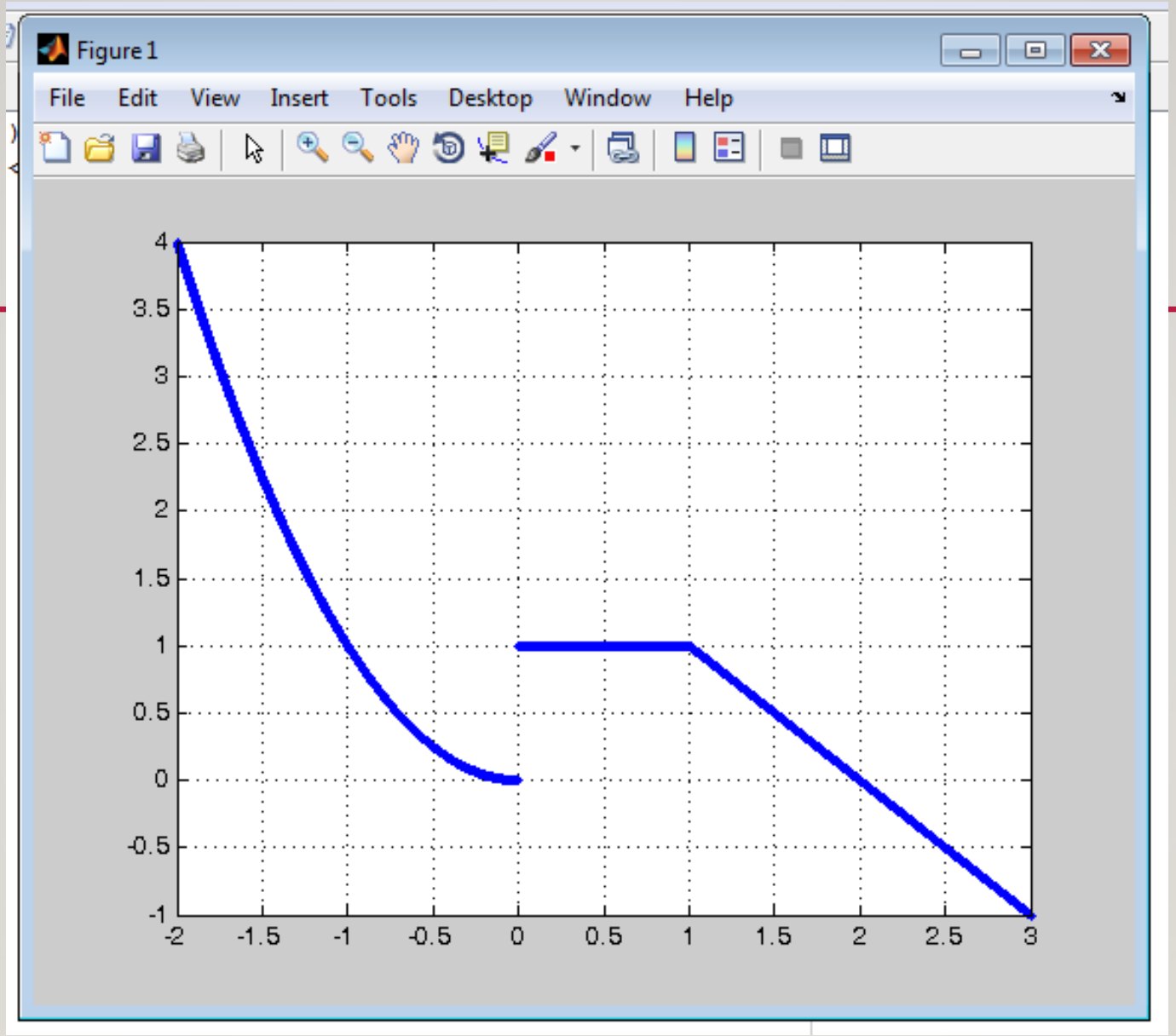
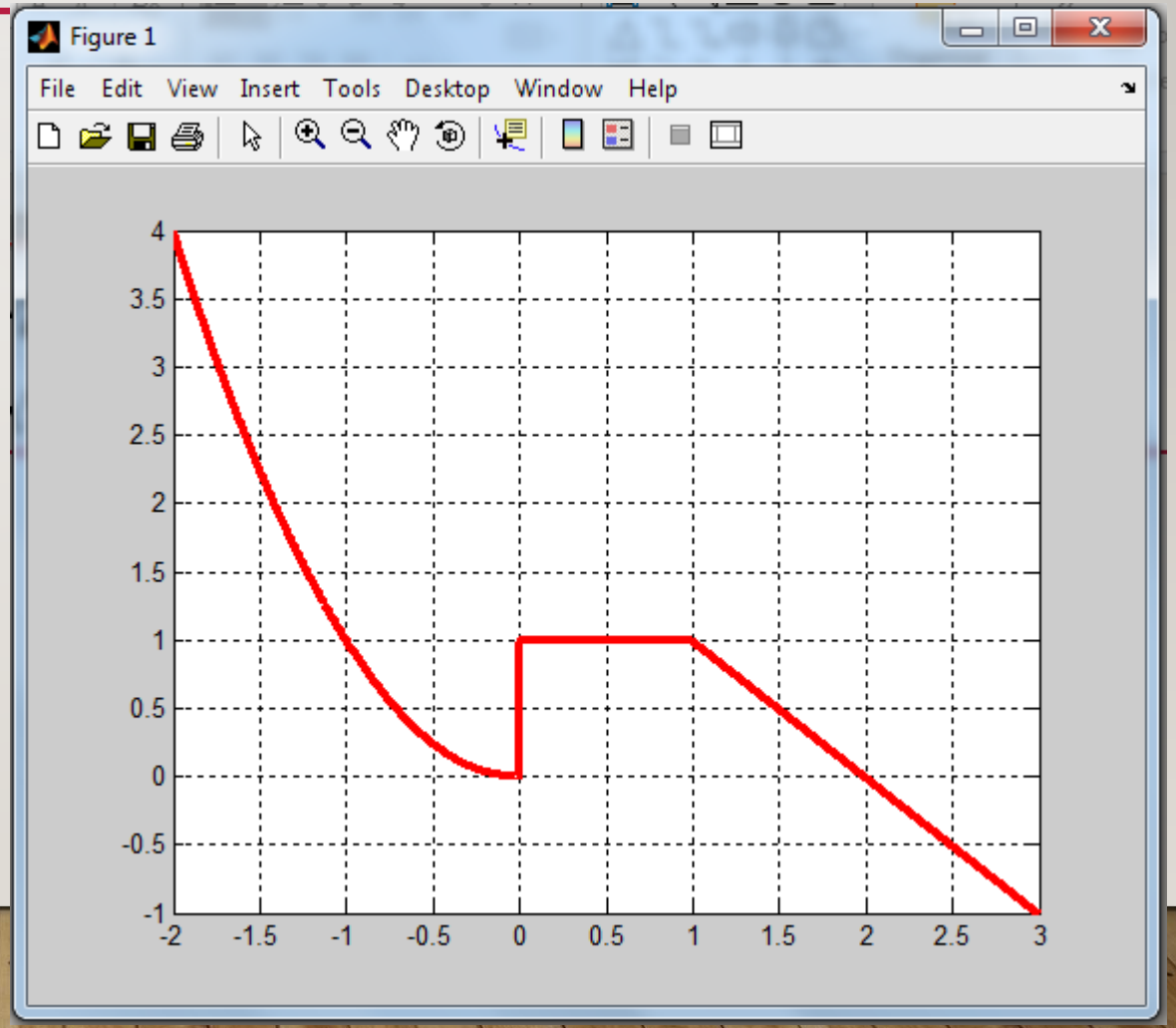


Figure 1
File Edit View Insert Tools Desktop Window Help
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5 3

- x = linspace(-2, 3, 3000);
- y = (x.^2).*(x<0)+1.*((0<=x)&(x<1))+(-x+2).*(x>=1);
- plot(x,y,'r','LineWidth',3), grid on



SUBPLOT

- Subplot(m,n,p) divide al dibujo en m x n y dibuja en la posición indicada por p.
- El primer subplot es la primera columna de la primera fila, el segundo esta en la segunda columna de la primer fila y así siguiendo.

FUNCIÓN ESCALÓN (U.M)

Parámetro de salida. En este caso sólo el valor de la función y.

Parámetros de entrada. Único parámetro tiempo t.

```
function y=u(t)
y=0.*(t<0)+1.*(t>=0);
end
```

Ecuación del escalón

FUNCIÓN: ECUACIÓN DE LA RECTA.

Parámetro de salida. En este caso sólo el valor de la función y.

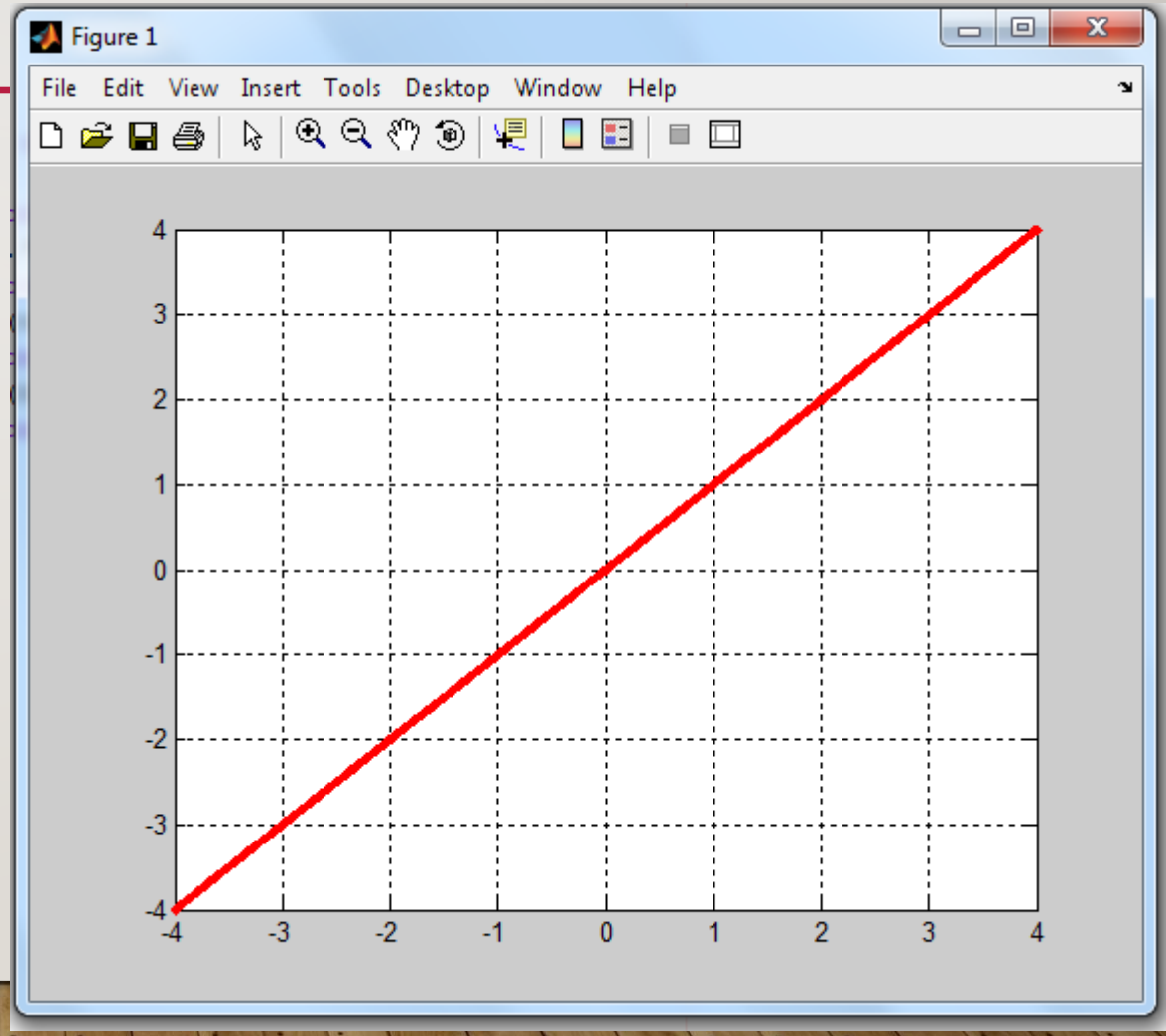
Parámetros de entrada. El tiempo t.

```
function y=recta(t)
y=t;
end
```

Ecuación de la recta

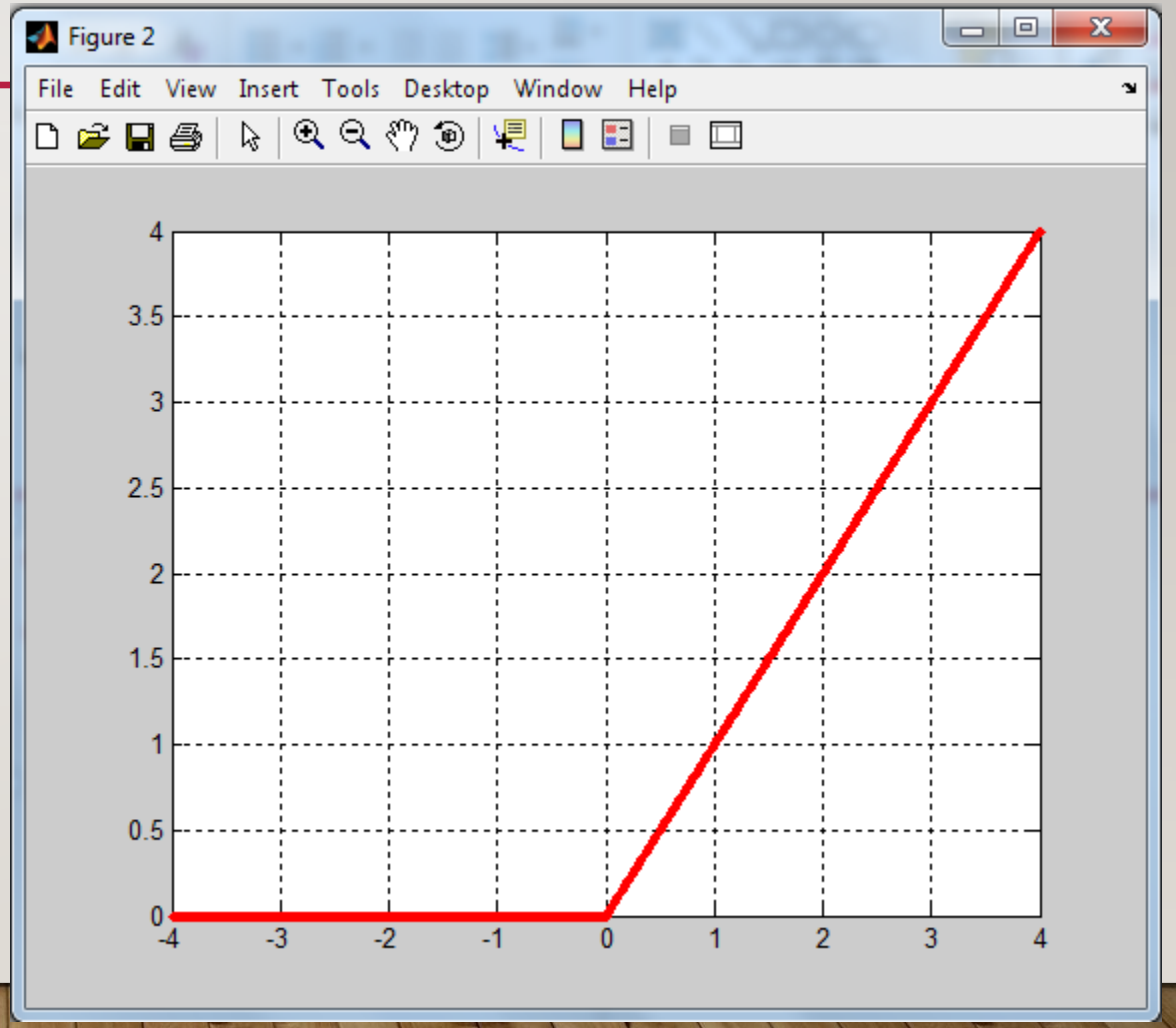
TRIÁNGULO: PASO A PASO(1)

- t=-4:0.01:4;
- yy= recta(t);
- plot(t,yy,'.r');grid on;



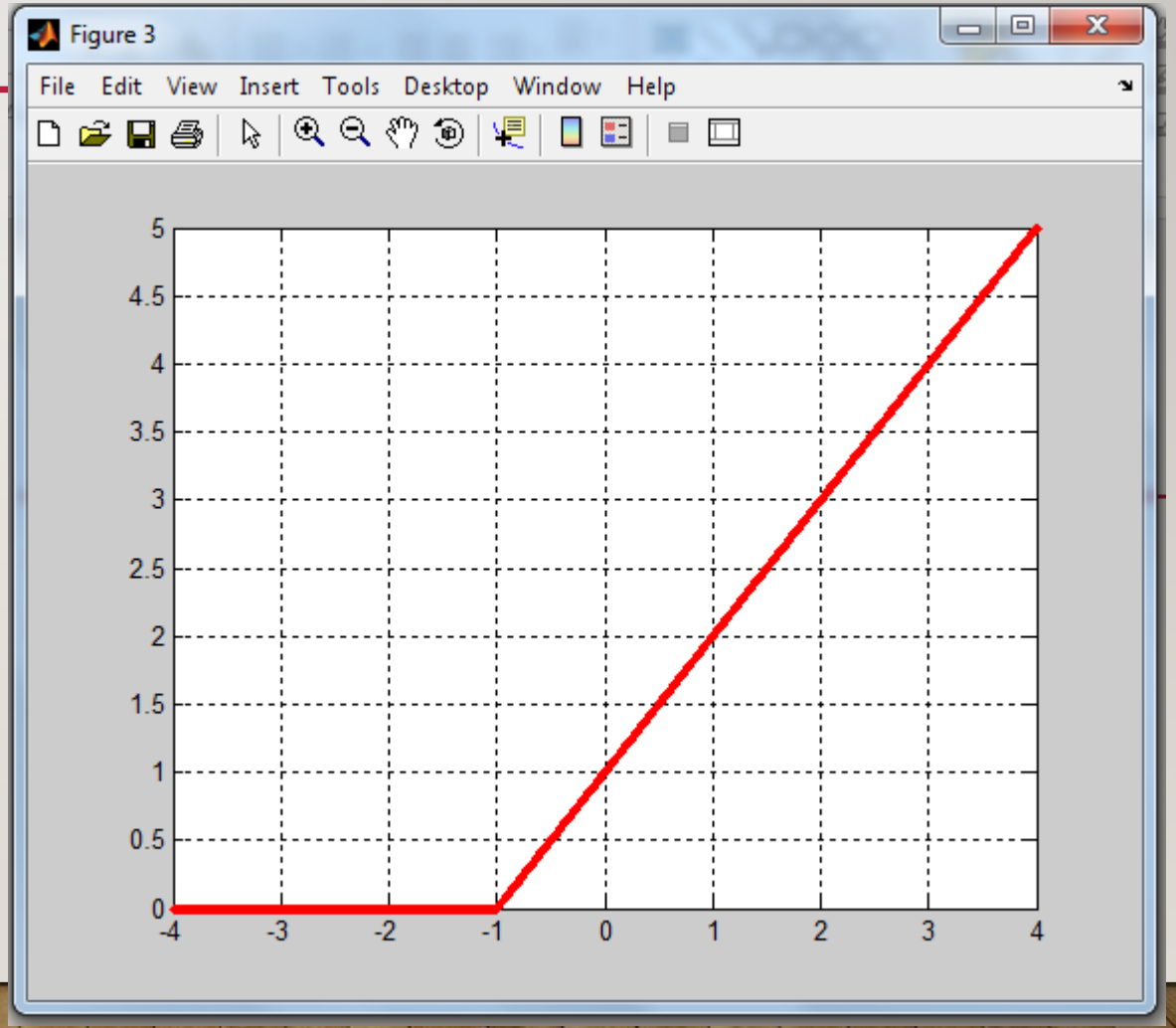
TRIÁNGULO: PASO A PASO(2)

- rampa=recta(t).*u(t);
- figure, plot(t,rampa, '.r');grid on;



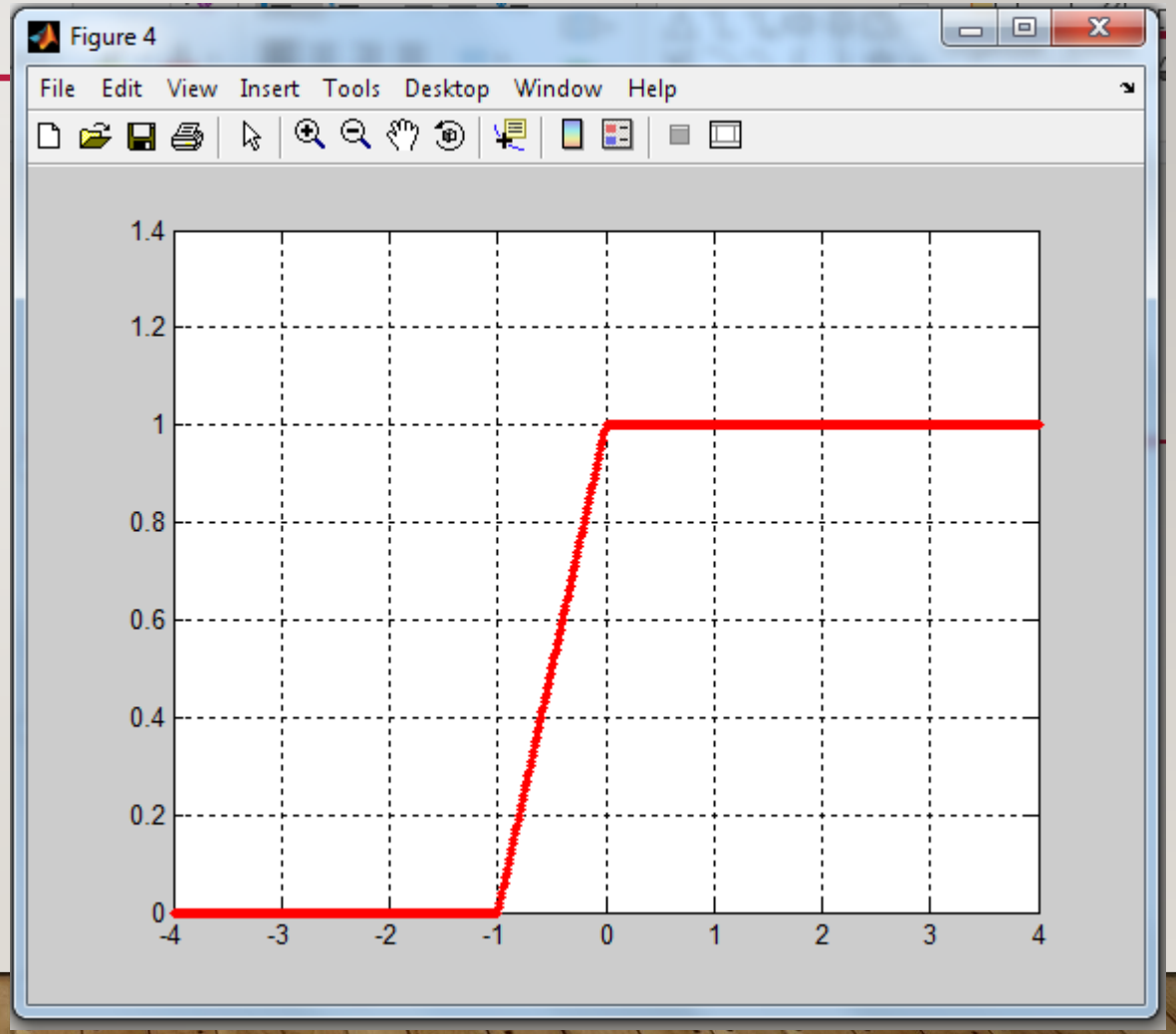
TRIÁNGULO: PASO A PASO(3)

- tri1=recta(t+1).*u(t+1);
- figure, plot(t,tri1,'.r');grid on;



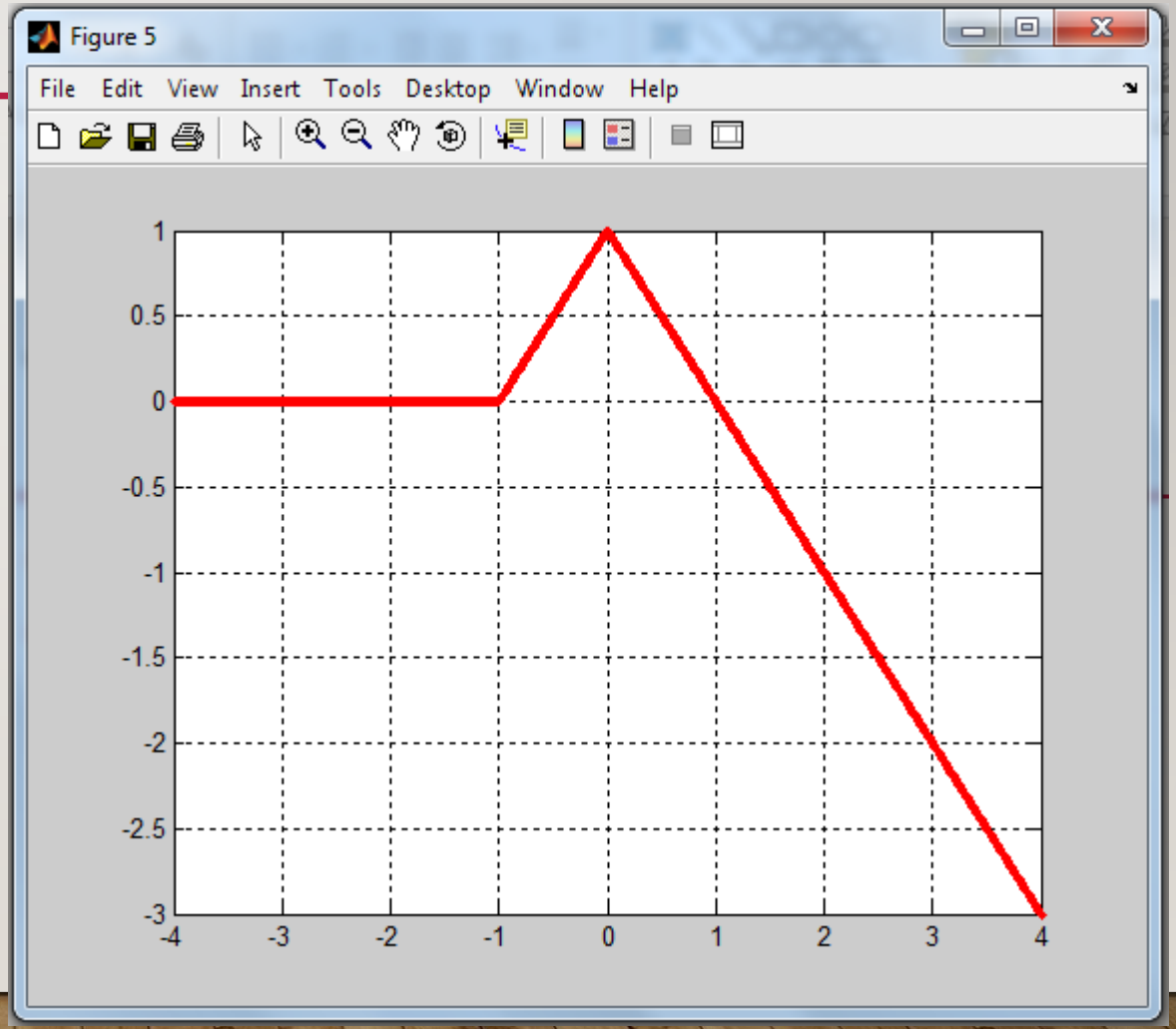
TRIÁNGULO: PASO A PASO(4)

- tri2=recta(t+1).*u(t+1)-recta(t).*u(t);
- figure, plot(t,tri2, '.r');grid on;



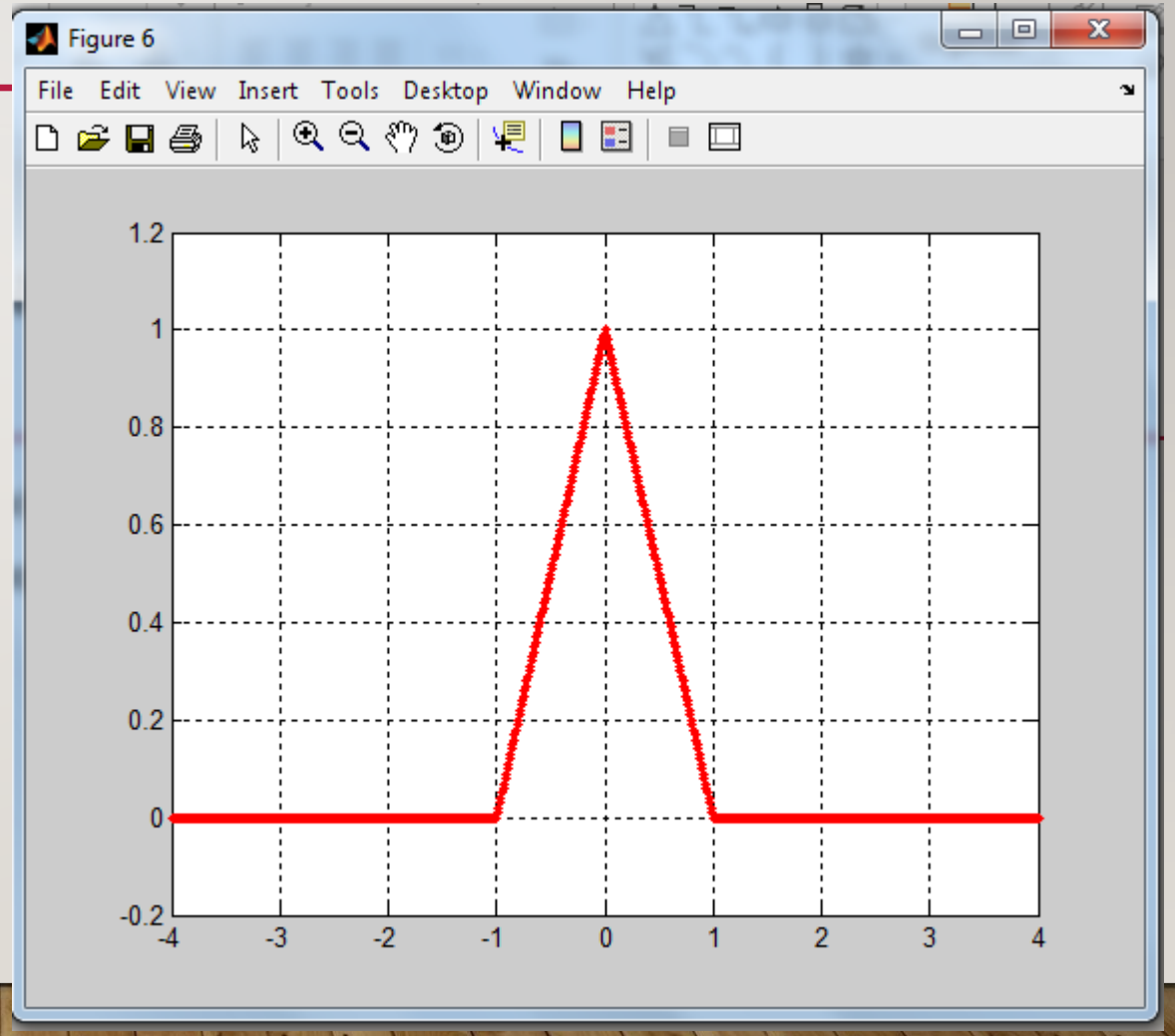
TRIÁNGULO: PASO A PASO(5)

- tri3=recta(t+1).*u(t+1)-2.*recta(t).*u(t);
- figure, plot(t,tri3, '.r');grid on;



TRIÁNGULO: PASO A PASO(6)

- tri4=recta(t+1).*u(t+1)-2.*recta(t).*u(t)+recta(t-1).*u(t-1);
- figure, plot(t,tri4, '.r');grid on;



Todo junto

```
t=-4:0.01:4;
yy= recta(t);
plot(t,yy, '.r');grid on;
rampa=recta(t).*u(t);
figure, plot(t,rampa, '.r');grid on;
tri1=recta(t+1).*u(t+1);
figure, plot(t,tri1, '.r');grid on;
tri2=recta(t+1).*u(t+1)-recta(t).*u(t);
figure, plot(t,tri2, '.r');grid on;
tri3=recta(t+1).*u(t+1)-2.*recta(t).*u(t);
figure, plot(t,tri3, '.r');grid on;
tri4=recta(t+1).*u(t+1)-2.*recta(t).*u(t)+recta(t-1).*u(t-1);
figure, plot(t,tri4, '.r');grid on;
```

FUNCIÓN ESCALÓN: OTRA FORMA

Parámetro de salida. En este caso sólo el valor de la función y.

Parámetros de entrada. El tiempo t y el retardo t0. Para un corrimiento hacia la derecha el valor de t0 es negativo. Hacia la izquierda es positivo.

```
function y=u(t, t0)
y=0.*(t<-t0)+1.*(t>=-t0);
end
```

Ecuación del escalón. El signo menos hace que un t0 negativo, el desplazamiento sea a derecha.

FUNCIÓN: ECUACIÓN DE LA RECTA.

Parámetro de salida. En este caso sólo el valor de la función y.

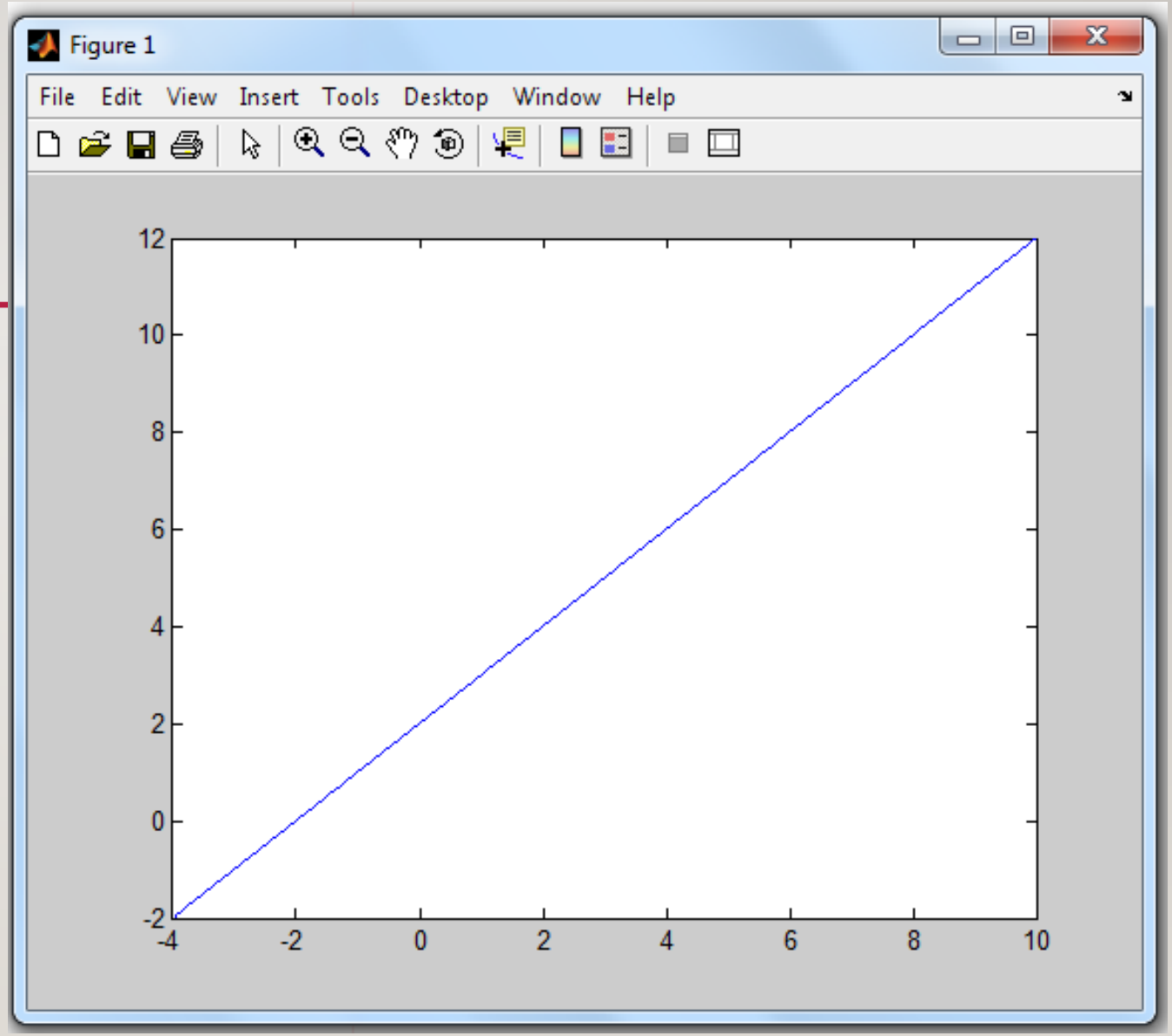
Parámetros de entrada. El tiempo t, la pendiente m, la ordenada al origen c y el retardo t0.

```
function y=recta(t,m,c,t0)
y=m.*(t+t0)+c;
end
```

Ecuación de la recta

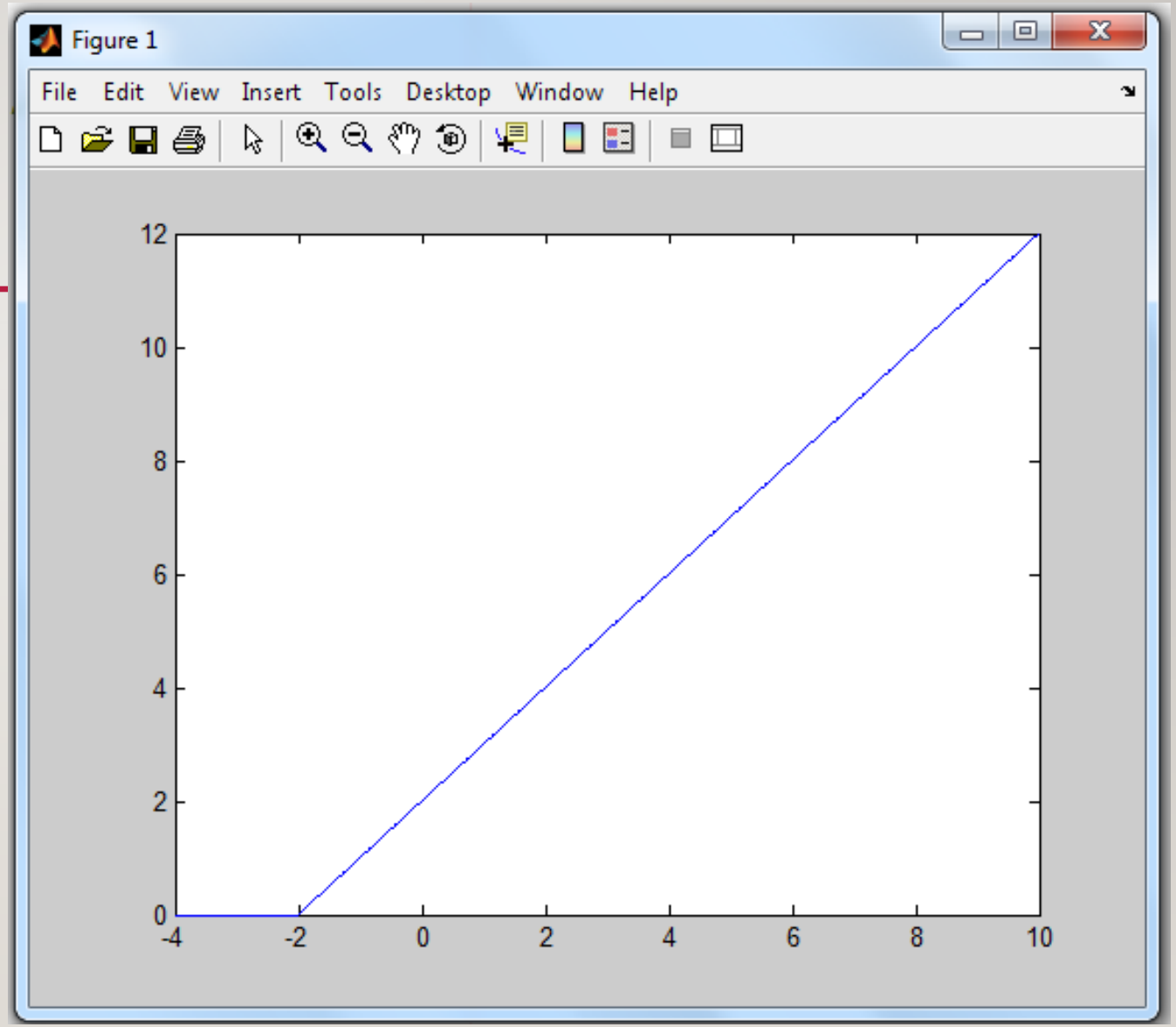
TRAPECIO

```
t=-4:0.01:10;
y=recta(t,1,0,2);
plot(t,y);
```

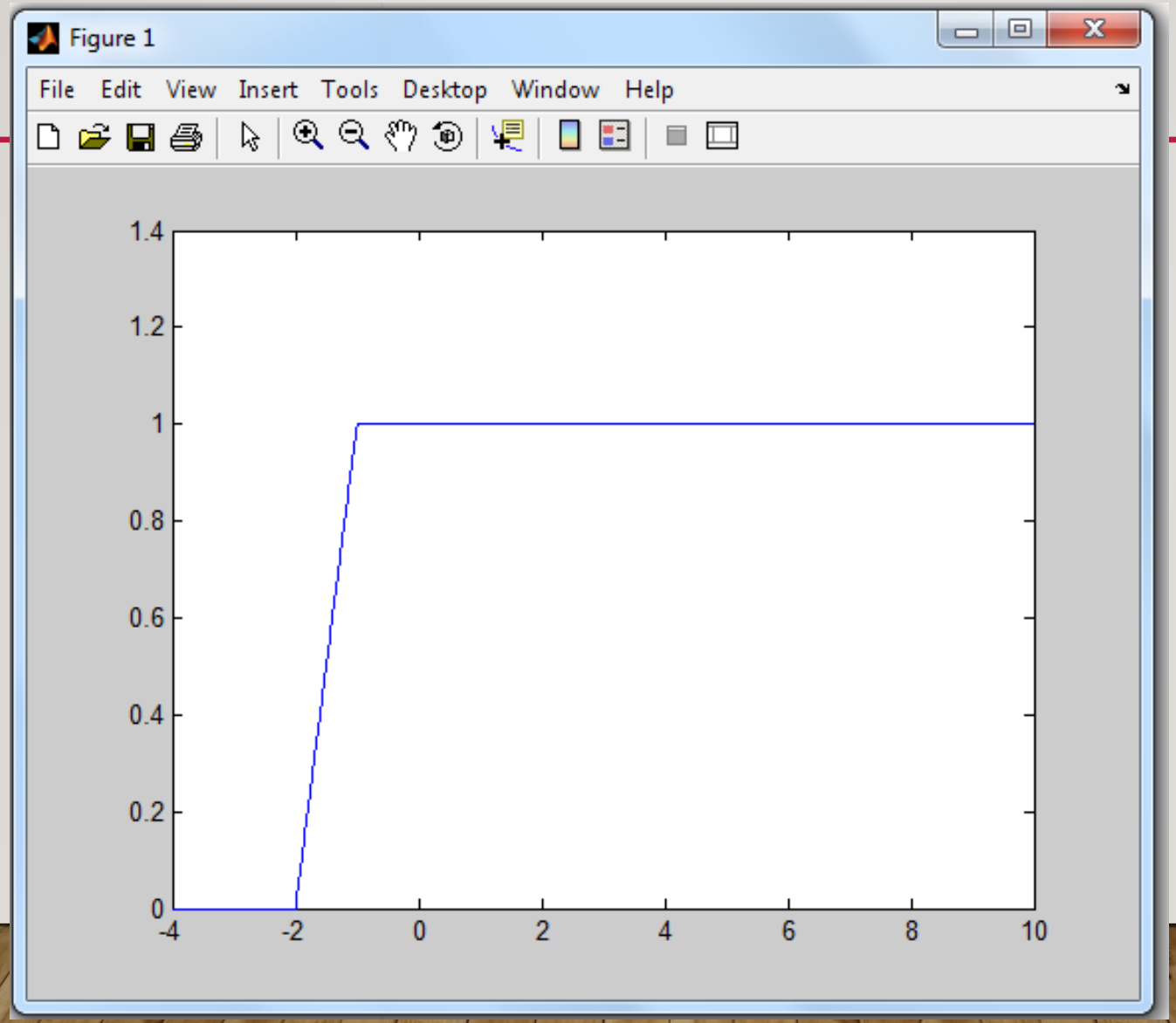


TRAPECIO

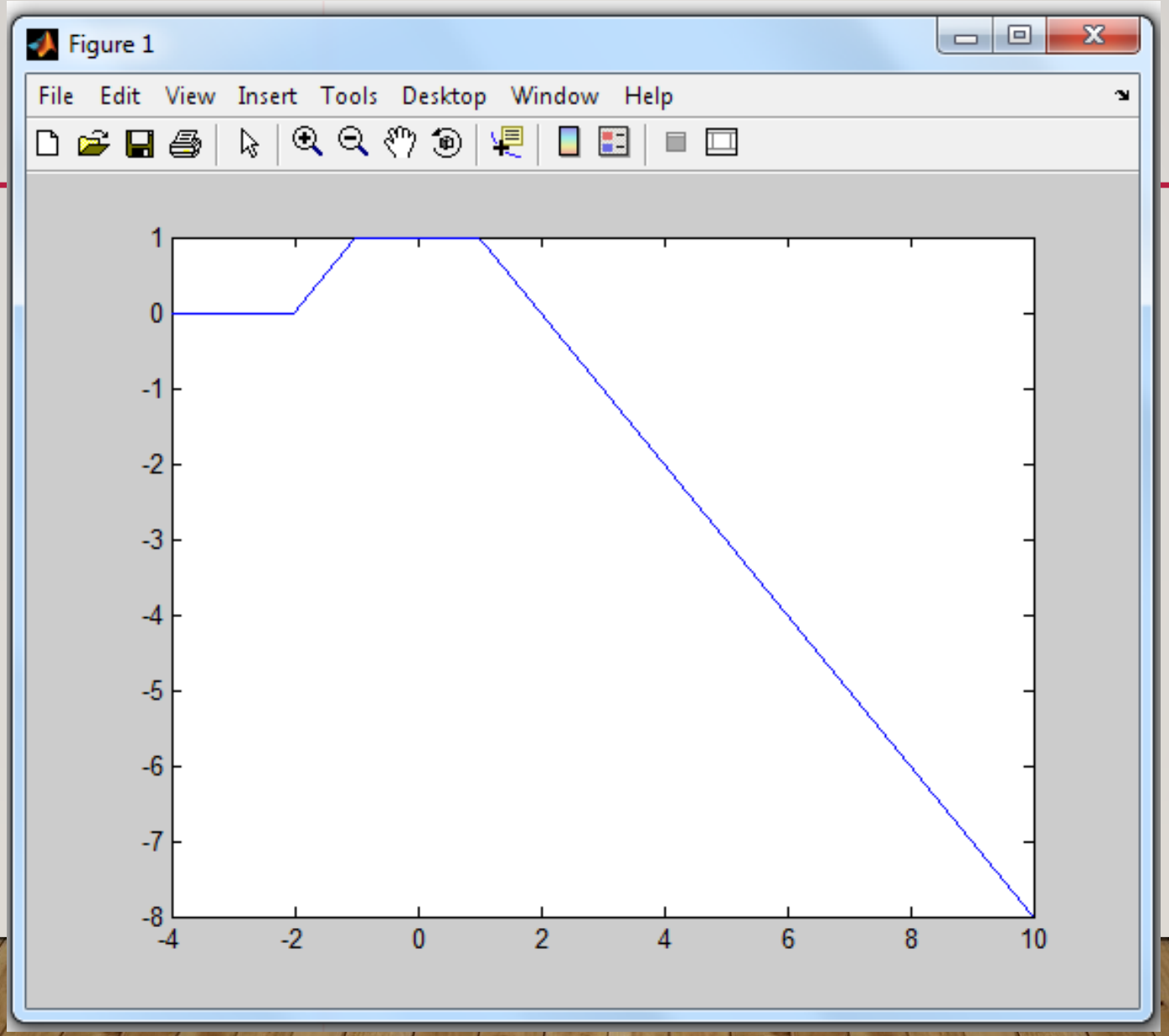
```
t=-4:0.01:10;
y=recta(t,1,0,2).*u(t,2);
plot(t,y);
```



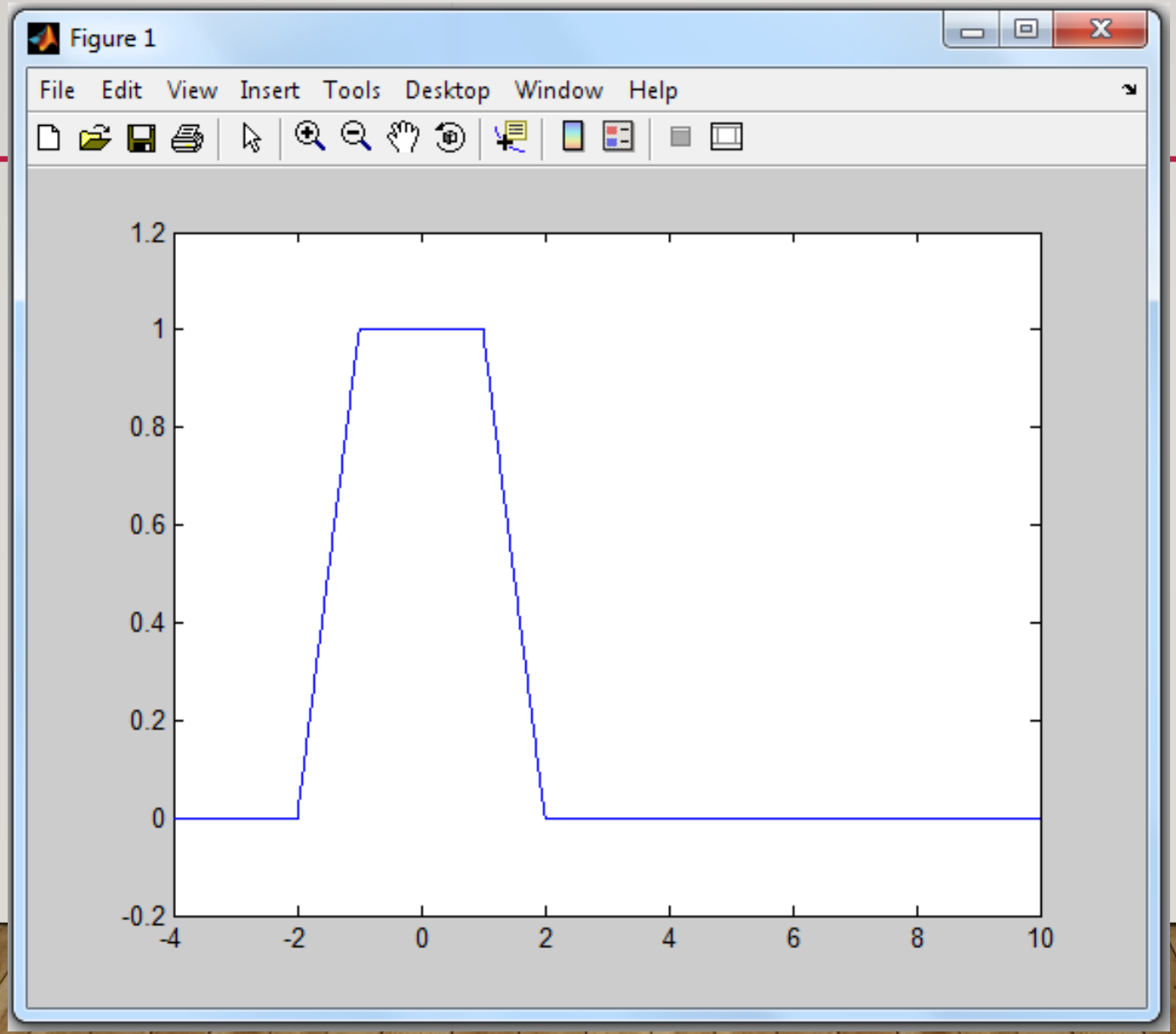
y=recta(t,1,0,2).*u(t,2)- recta(t,1,0,1).*u(t,1);



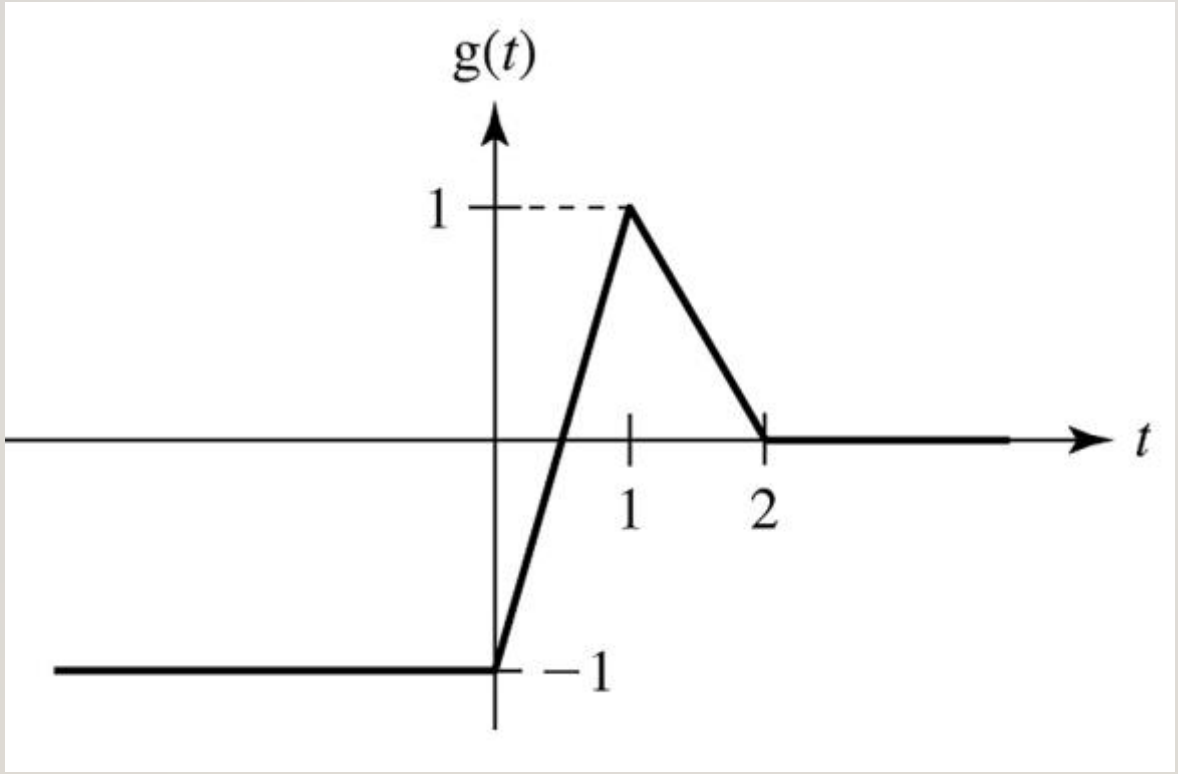
```
y=recta(t,1,0,2).*u(t,2)- recta(t,1,0,1).*u(t,1)- recta(t,1,0,-1).*u(t,-1);
```



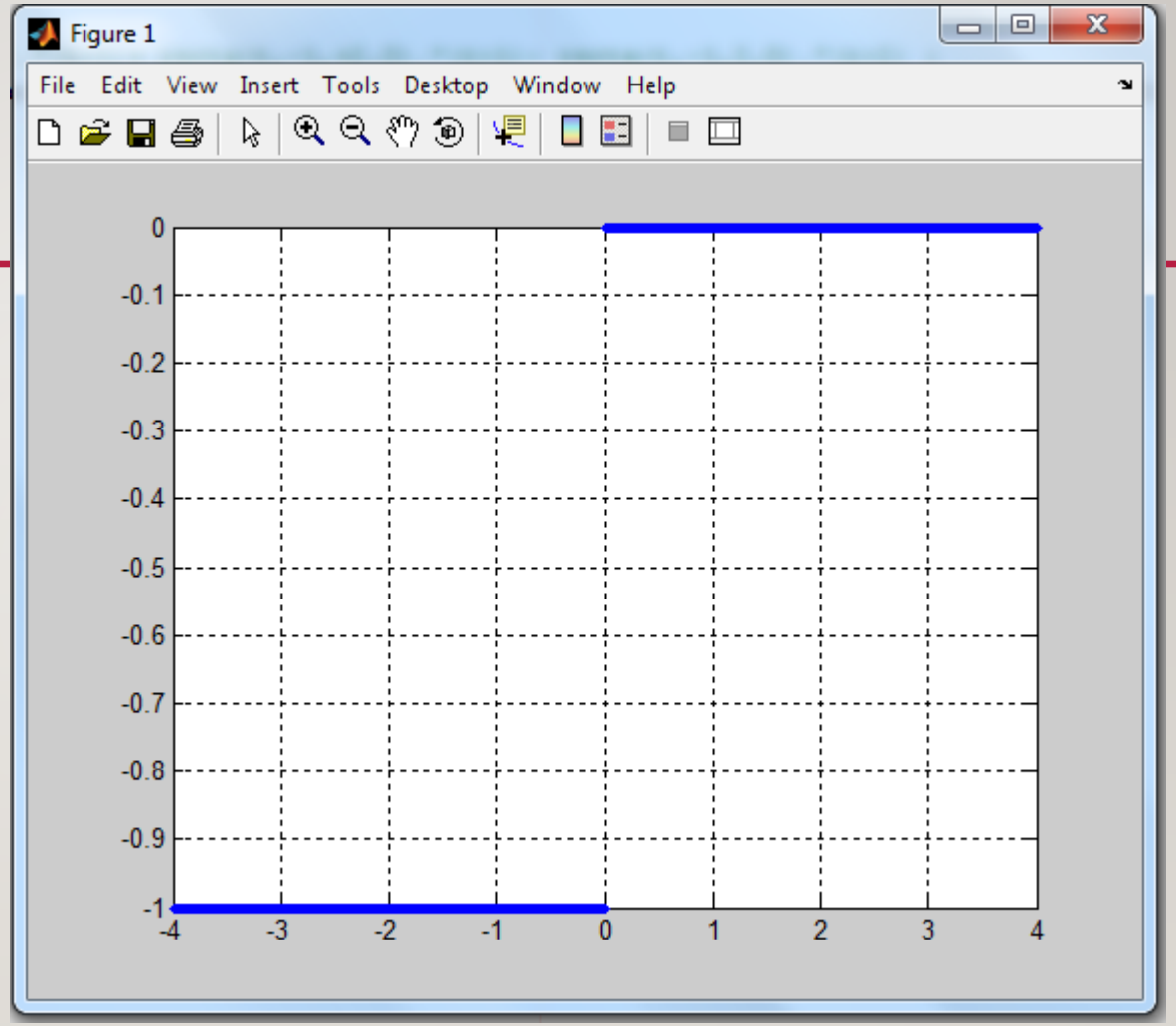
```
y=recta(t,1,0,2).*u(t,2)- recta(t,1,0,1).*u(t,1)- recta(t,1,0,-1).*u(t,-1)+recta(t,1,0,-2).*u(t,-2);
```



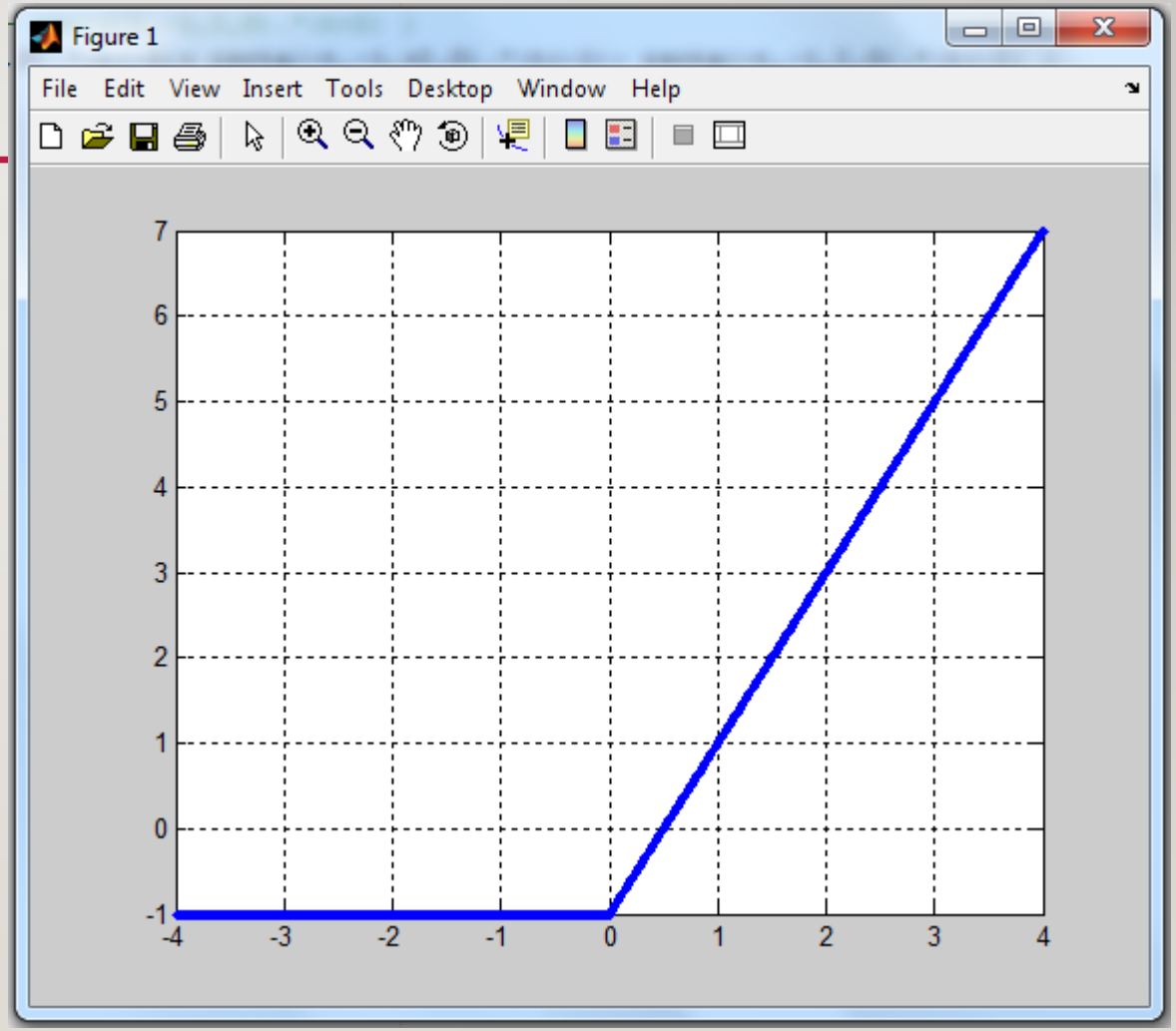
ENCONTRAR PARTE PAR E IMPAR DE G(T)



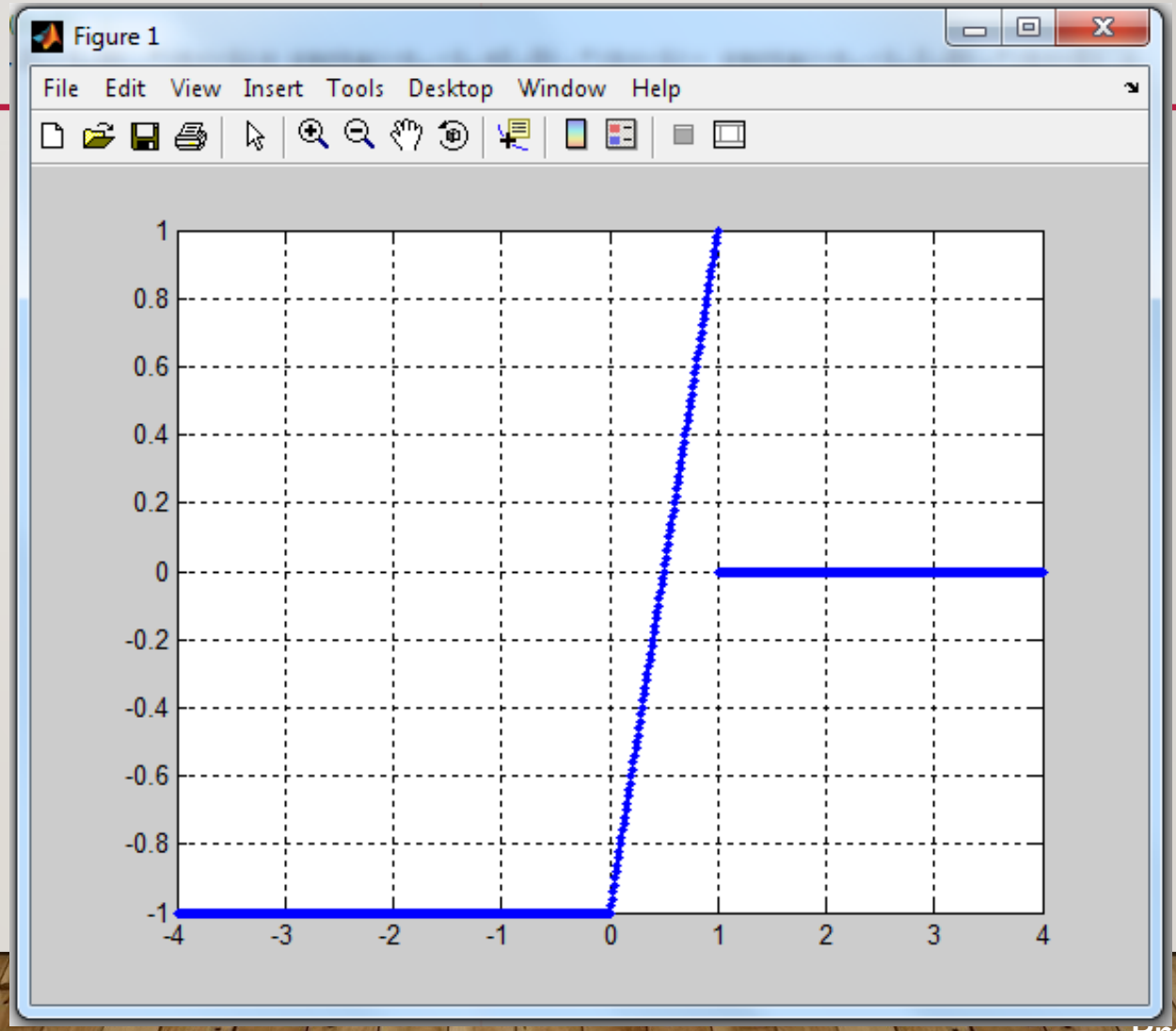
```
t=-4:0.01:+4;
g=-u(-t,0);
plot(t,g,'.');grid on;
```



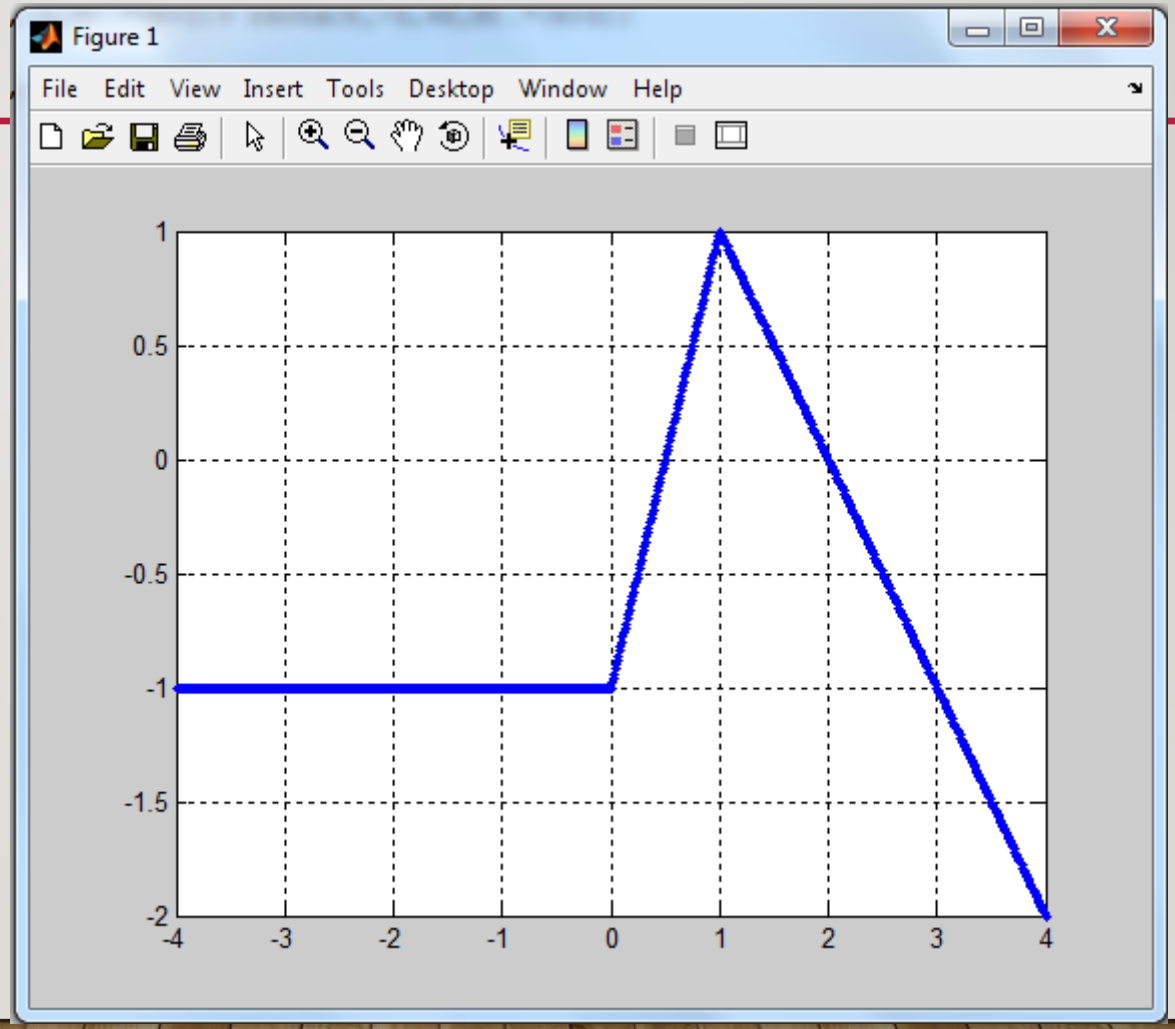
```
t=-4:0.01:+4;
g=-u(-t,0) + recta(t,2,-1,0).*(t>0);
plot(t,g, '.');grid on;
```



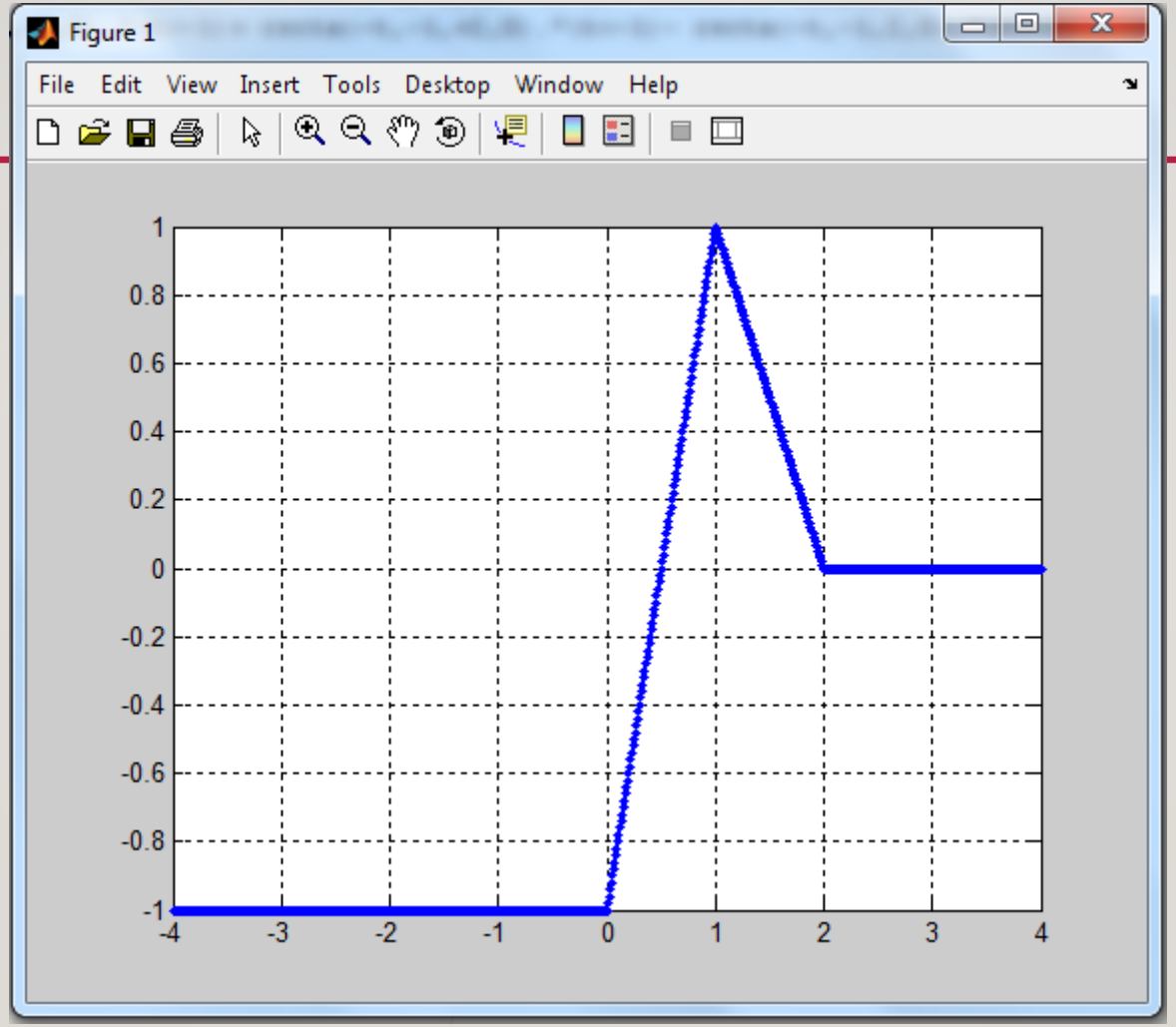
```
t=-4:0.01:+4;
g=-u(-t,0) + recta(t,2,-1,0).*(t>0) - recta(t,2,-1,0).*(t>1);
plot(t,g,'.');grid on;
```



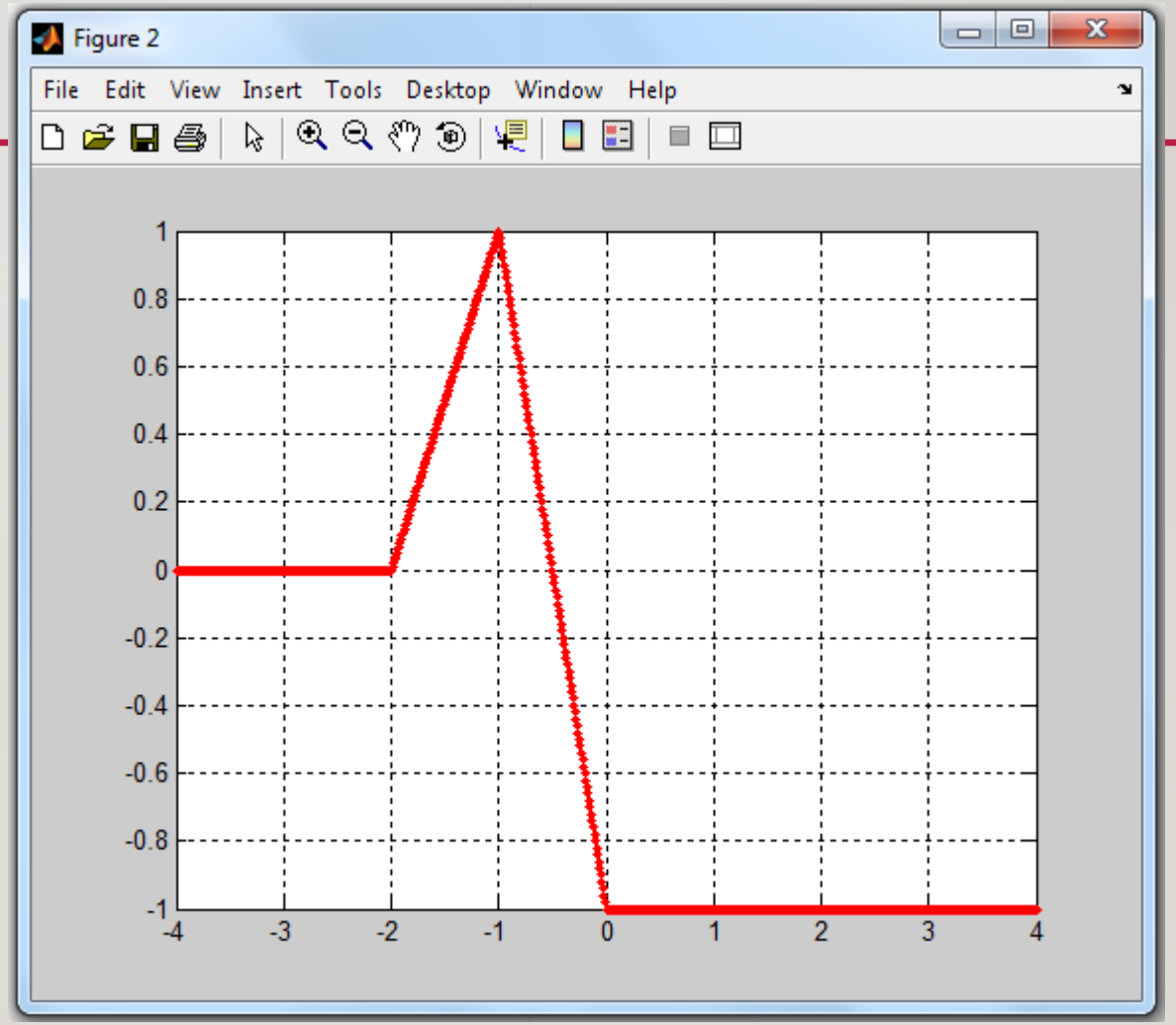
```
t=-4:0.01:+4;
g=-u(-t,0) + recta(t,2,-1,0).*(t>0) - recta(t,2,-1,0).*(t>1)+ recta(t,-1,+2,0).*(t>1);
```



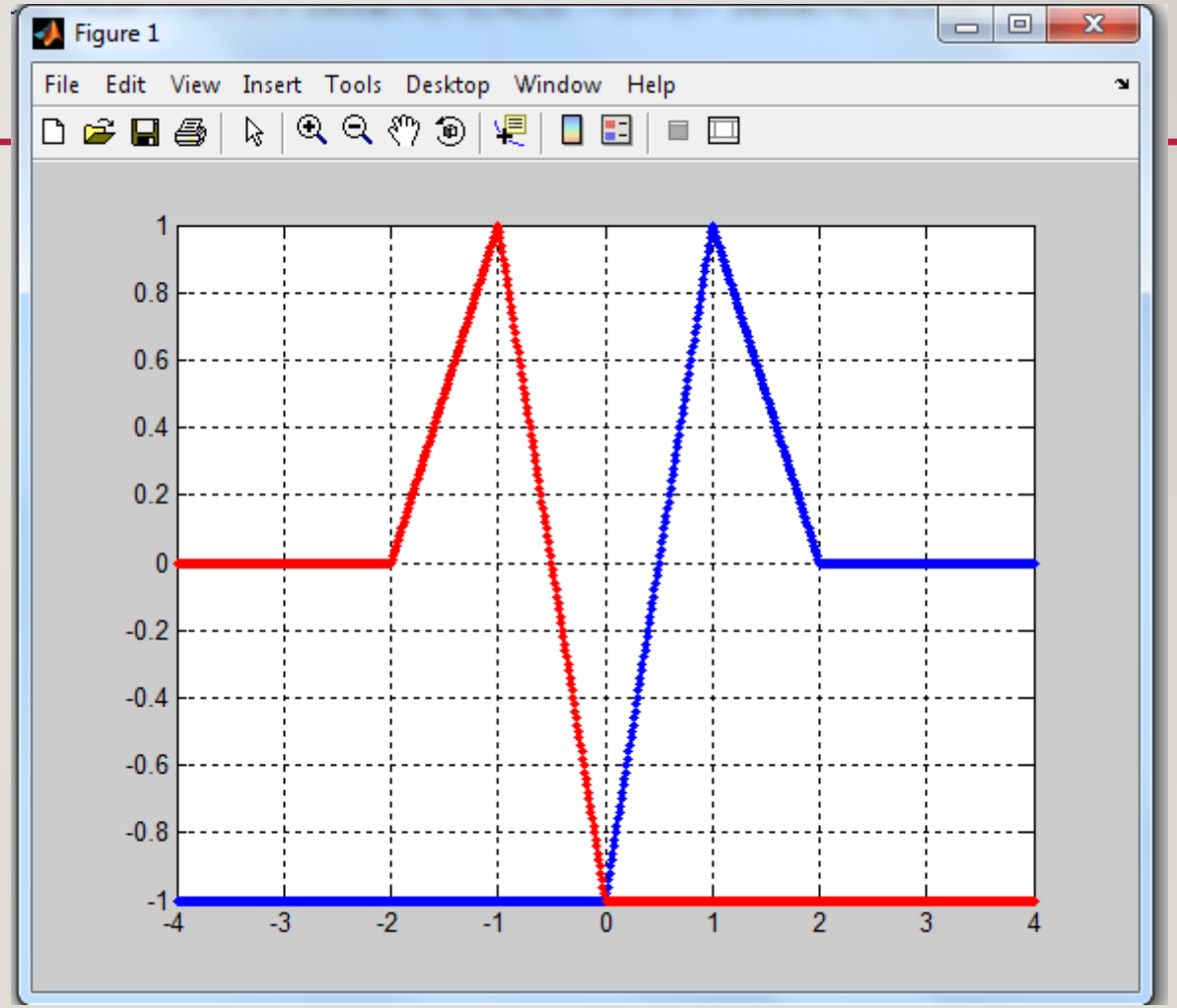
```
g=-u(-t,0) + recta(t,2,-1,0).*(t>0) - recta(t,2,-1,0).*(t>1)+ recta(t,-1,+2,0).*(t>1)- recta(t,-1,2,0).*(t>2);
% sería g=g(t)
```



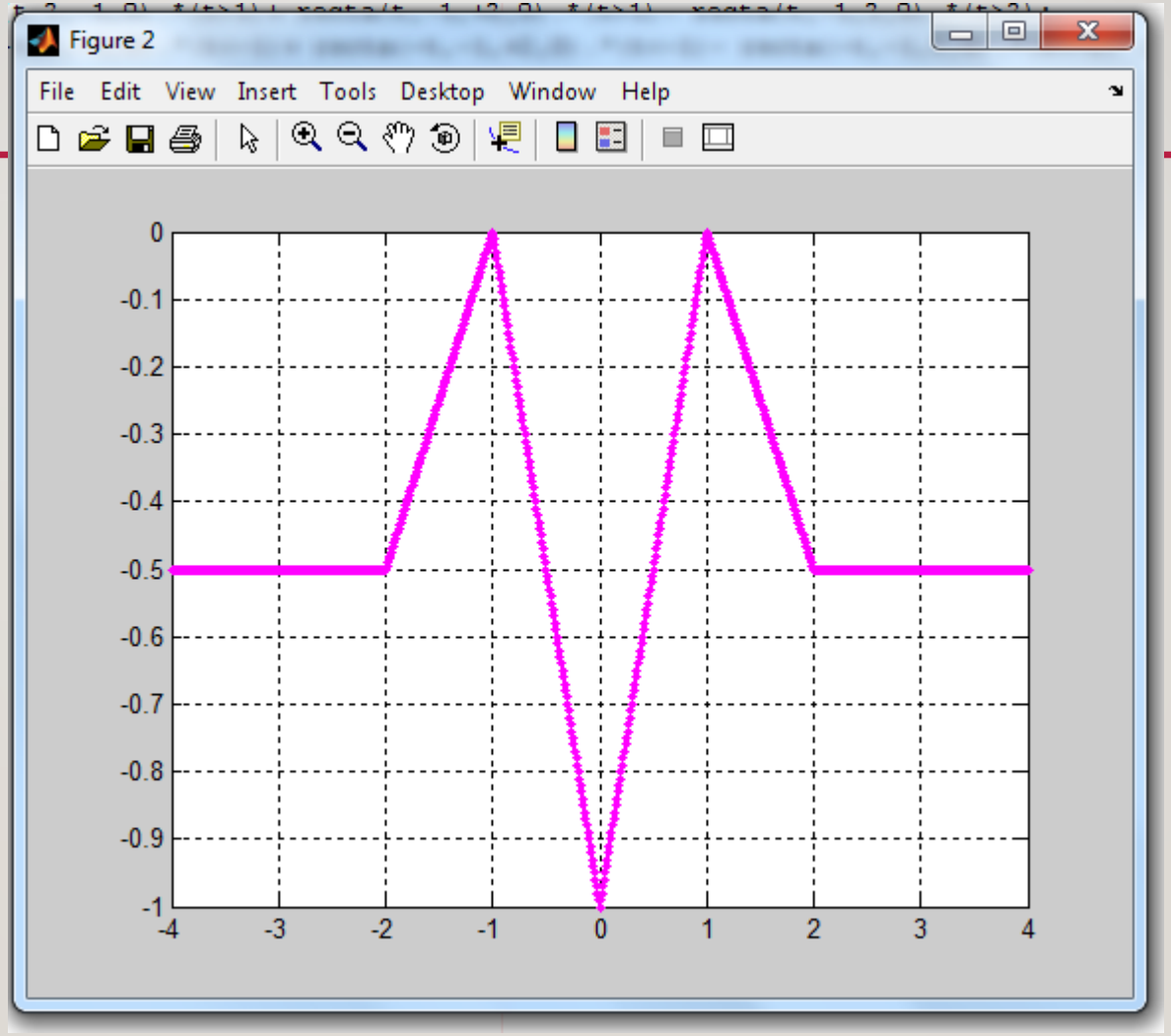
```
gg=-u(t,0)+ recta(-t,2,-1,0).*(t<0)- recta(-t,2,-1,0).*(t<-1)+ recta(-t,-1,+2,0).*(t<-1)- recta(-t,-1,2,0).*(t<-2) ;
figure; plot(t,gg,'. r');grid on;
% sería gg=g(-t)
```



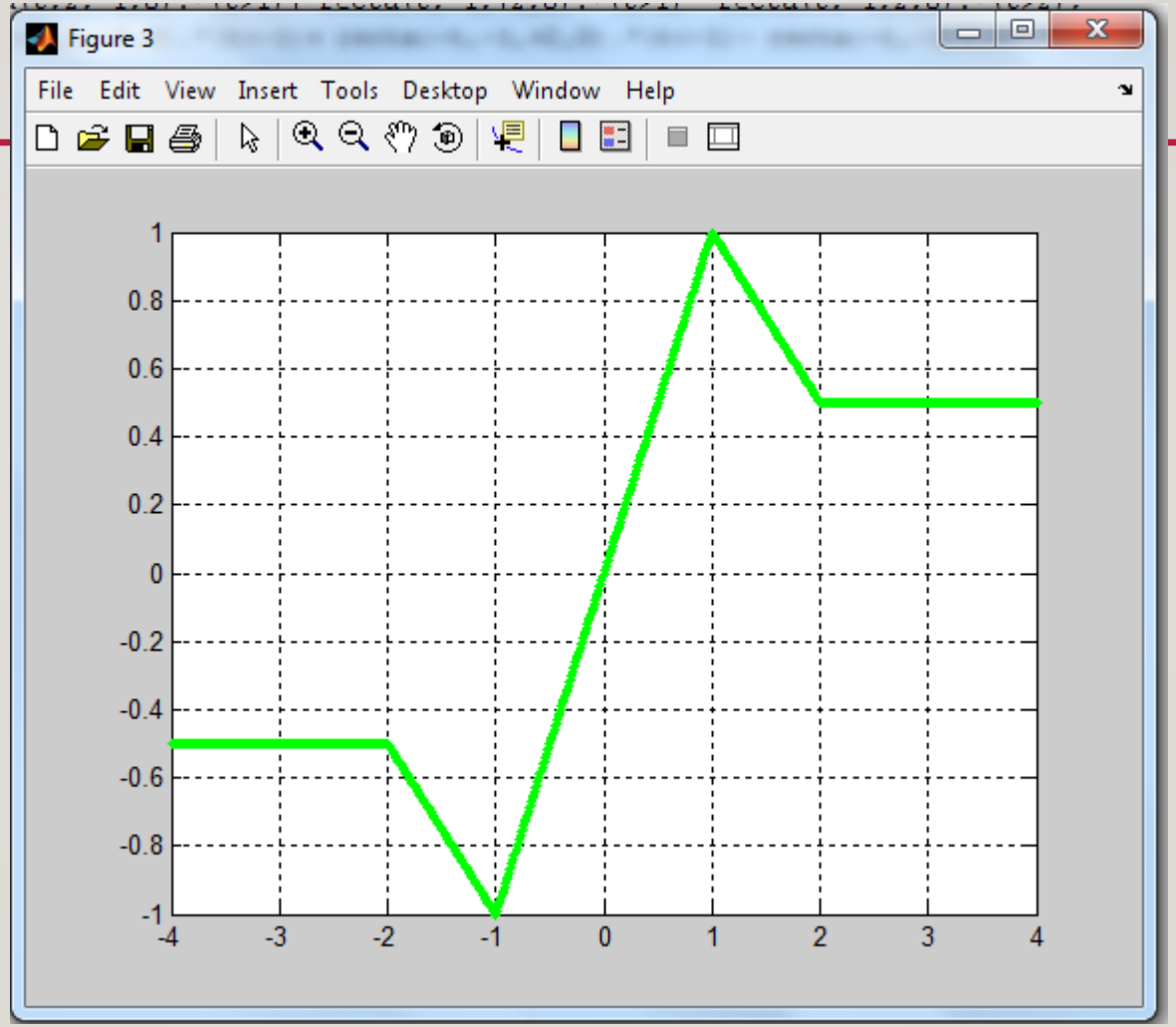
```
plot(t,g,'.'); grid on; hold on;
plot(t,gg,'. r'); grid on;
```



figure; plot(t, (g+gg)/2, '. m'); grid on; %Función par



figure; plot(t, (g-gg)/2, '. g'); grid on; %Función impar



Otra manera de escribir la función escalón (sin retardo)

```
function y=u(t)
y=0.*(t<0)+1.*(t>=0);
end
```

La ecuación del trapecio quedaría:

```
y=recta(t,1,0,2).*u(t+2)- recta(t,1,0,1).*u(t+1)- recta(t,1,0,-1).*u(t-1)+recta(t,1,0,-2).*u(t-2);
```