Programación con Octave/Octave-UPM (II)

Mario Bermejo

CLUB DE INFORMÁTICA CAMINOS

11 y 13 marzo 2015

Índice

Editor de Octave/UPM

Editor

Desde el menú → *Archivo* → *Archivo nuevo*
O directamente con el botón *Archivo nuevo*, o cargando un archivo guardado previamente.

- Permite guardar los comandos en un archivo y ejecutarlos las veces que se quiera.
- Tiene resaltado de sintaxis e indentación (sangría)
- Extensión *.m*
- % : para poner comentarios.
- El diario de comandos producido por el comando *diary* también se puede guardar.

Ejecución de archivos

Ejecución básica

Pulsado el triángulo de ejecutar.

- Los archivos guardados se pueden llamar directamente desde la ventana de comandos

Otros editores

No es necesario usar el editor de Octave UPM, se pueden usar otros.
Editor de texto plano: gedit

Funciones

Estructura de función

function [lista de valores de retorno] = name(lista de argumentos)

Ejemplo

```
function[x1, x2]= fseggrado(a,b,c)
  x1=-b+sqrt(b^2-4*a*c)/(2*a*c);
  x2=-b-sqrt(b^2-4*a*c)/(2*a*c);
end
```

LLamada y resultado

```
>> [x1,x2]=fseggrado(1,-4,2)

x1 =    4.7071
x2 =    3.2929
```


Características de las funciones

- Las variables definidas dentro de una función son variables locales, en el sentido de que son inaccesibles desde otras partes del programa y en el de que no interfieren con variables del mismo nombre definidas en otras funciones o partes del programa.
- Para que la función tenga acceso a variables que no han sido pasadas como argumentos es necesario declarar dichas variables como variables globales, tanto en el programa principal como en las distintas funciones que deben acceder a su valor.

LLamadas a funciones desde programas

LLamada

[lista de valores de retorno] = name(lista de argumentos)

Ejemplo: llamadaafuncion.m

```
%Cálculo de varias ecuaciones de segundo grado
[x1a,x2a]=fseggrado(1,-4,2);
disp(x1a),disp(x2a);
[x1b,x2b]=fseggrado(1,4,2);
disp(x1a),disp(x2a);
```

Resultado

```
    4.7071
    3.2929
   -3.2929
   -4.7071
```


LLamadas a funciones desde programas

Ejemplo: llamadaafuncion2.m

```
%Cálculo de varias ecuaciones de segundo grado
%Matriz de datos
M=[1 -4 2; 1, 4, 2; 3 -5 1; 2 5 -2];
%LLamadas a funcion
for i=1:length(M)
    a=M(i,1);
    b=M(i,2);
    c=M(i,3);
    [x1,x2]=fseggrado(a,b,c);
    disp('x1 es:'),disp(x1),disp('x2 es:'),disp(x2);
end
```

LLamadas a funciones desde programas

Resultado

```
x1 es:
    4.7071
x2 es:
    3.2929
x1 es:
   -3.2929
x2 es:
   -4.7071
x1 es:
    5.6009
x2 es:
    4.3991
x1 es:
   -5.8004
x2 es:
   -4.1996
```


Comando fprintf

Estructura

fprintf(fileID, format, A, ...)

Ejemplo: llamadaafuncion4fprintf.m

```
%Cálculo de varias ecuaciones de segundo grado
%Matriz de datos
M=[1 -4 2; 1, 4, 2; 3 -5 1; 2 5 -2];
%LLamadas a funcion
for i=1:length(M)
    [x1,x2]=fseggrado(M(i,1),M(i,2),M(i,3));
    fprintf('%s %g \t %s %g \n','x1 es: ',x1,'x2 es: ',x2);
end
```

Resultado

```
x1 es:  4.70711    x2 es:  3.29289
x1 es:  -3.29289   x2 es:  -4.70711
x1 es:  5.60093    x2 es:  4.39907
x1 es:  -5.80039   x2 es:  -4.19961
```

Guardar variables

Comando *save*

- Guarda la variable o variables que se le indiquen en un archivo.
- Se pueden editar con el editor de matrices
- Octave lo guarda de forma transparente
- Octave UPM lo guarda en un formato binario
- Matlab guarda en un formato binario, no editable

Ejemplo: salvar la matriz M

```
save matrizM M
```

Añade la extensión `.mat` al nombre que se le indique. Ejm: `matrizM.mat`

Guardar variables

Comando *save -ascii*

- Se guarda en un formato editable y visible
- Pero luego no es posible recuperar diferenciando las distintas matrices

Ejemplo: salvar la matriz M y M2

```
save matricesMyM2.asc M M2 -ascii
```

Archivo matricesMyM2.asc

```
 1.0000000e+000 -4.0000000e+000  2.0000000e+000
 1.0000000e+000  4.0000000e+000  2.0000000e+000
 3.0000000e+000 -5.0000000e+000  1.0000000e+000
 2.0000000e+000  5.0000000e+000 -2.0000000e+000
 1.0000000e+000  0.0000000e+000  0.0000000e+000
 0.0000000e+000  1.0000000e+000  0.0000000e+000
 0.0000000e+000  0.0000000e+000  1.0000000e+000
```

save y load

`save` sin indicar variables

- *save* con sólo en nombre de archivo guarda todas las variables del entorno.
- Lo guarda en un archivo binario.

`load «fichero.mat»`

- Carga todas las variables almacenadas en el fichero indicado.

Uso de `-ascii`

- Si se guarda con `-ascii` se guarda de forma transparente pero luego no se pueden recuperar cada variable por separado (para Matlab y Octave UPM)
- Octave por consola sin `-ascii` guarda de forma transparente y es posible recuperar todas las variables.

Lectura de datos

Lectura de archivos de datos

- Muchos progrmas dan salidas de datos por columnas
- Es posible leerlos desde Matlab
- Con el comando load se cargan como si fueran una matriz

Cargar archivo sseh.dat

```
load sseh.dat
```

Cargar archivo sseh.dat en la variable sismo

```
sismo = load('sseh.dat')
```

Entrada / salida de ficheros

Abrir/cerrar ficheros

- fopen
- fclose

Ejemplo

```
fid = fopen('resultadosbin.txt','w+');
fwrite(fid,eye(4));
fclose(fid);
```

Resultado archivo

El resultado es un fichero binario que no se puede leer directamente. Para leerlo hace falta el comando `fread`

Entrada / salida de ficheros

Abrir/cerrar ficheros

```
variable = fopen('nombre_archivo','permisos');
```

Permisos

- **r** Abre archivo existente para lectura
- **r+** Abre archivo existente para lectura y escritura
- **w** Crea archivo para escritura, y si existe lo borra y lo crea de nuevo vacío.
- **w+** Crea archivo para escritura y lectura, y si existe lo borra y lo crea de nuevo vacío.
- **a** Crea archivo para escritura, y si existe añade el contenido al final.
- **a+** Crea archivo para escritura y lectura, y si existe añade el contenido al final.

Escritura archivo ascci formateada

Escribir en archivos: *fprintf*

- Se usa el comando `fprintf`

Ejemplo

```
fid = fopen('resultados.txt','w+');
for i=1:length(M)
  fprintf(fid,'%g \t %g \t %g \n', M(i,1), M(i,2), M(i,3));
end
fclose(fid);
```

Resultado archivo

```
1    -4    2
1    4    2
3    -5    1
2    5    -2
```


Estructura del formato

Una cadena con los tipos

- **%d** Entero en sistema decimal
- **%f** Reales punto fijo
- **%e** Reales de punto flotante
- **%g** Usa d, e o f seleccionanado mayor precisión en mínimo espacio
- **%c** Un carácter
- **%s** Una cadena

y con las secuencias de escape

- **\n** Salto de linea
- **\t** Tabulador
- **\b** Backspace
- ...

Estructura del formato

Ejemplo

```
fid = fopen('resultados.txt','w+');
for i=1:length(M)
  fprintf(fid,'%g \t %s %3.2f \t %e \n', M(i,1),
    'segunda columna', M(i,2), M(i,3));
end
fclose(fid);
```

Resultado archivo

```
1    segunda columna -4.00    2.000000e+000
1    segunda columna 4.00    2.000000e+000
3    segunda columna -5.00    1.000000e+000
2    segunda columna 5.00    -2.000000e+000
```

Lectura archivo ascii formateada

Leer de archivos: *fscan*

- Se usa el comando `fscan`

Ejemplo

```
fid = fopen('sseh.dat','r');
[A,count]=fscanf(fid,'%g %g',[2 Inf]);
fclose(fid);
count
>>> 4010
```

Gráficas 2D

Comandos para gráficas

- `plot(x,y)` crea un gráfico a partir de vectores y/o columnas de matrices, con escalas lineales sobre ambos ejes.
- `loglog(x,y)` ídem con escala logarítmica en ambos ejes
- `semilogx(x,y)` ídem con escala lineal en el eje de ordenadas y logarítmica en el eje de abscisas
- `semilogy(x,y)` ídem con escala lineal en el eje de abscisas y logarítmica en el eje de ordenadas
- `bar(x,Y)` gráficos de barras. `barh(x,Y)` gráfico de barras horizontales
- `errorbar(x,y,e)` gráficos de errores.
- ...

Gráficas 2D

Comandos para gráficas

Existen comandos orientados a añadir títulos al gráfico, a cada uno de los ejes, a la salida en archivo, a introducir texto, etc. Algunos a continuación:

- `title('título')` añade un título al dibujo.
- `xlabel('abcisas')` añade una etiqueta al eje de abscisas. Con `xlabel off` se elimina
- `ylabel('ordenadas')` añade una etiqueta al eje de ordenadas. Con `ylabel off` se elimina
- `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. Si texto es también un vector de cadenas de texto de la misma dimensión, cada elemento se escribe en las coordenadas correspondientes.
- `legend()` define rótulos para las distintas líneas o ejes utilizados en la figura.
- `grid on` activa la inclusión de una cuadrícula en el dibujo. Con `grid off` desaparece la cuadrícula.

Realizar gráficas 2D

Comando plot

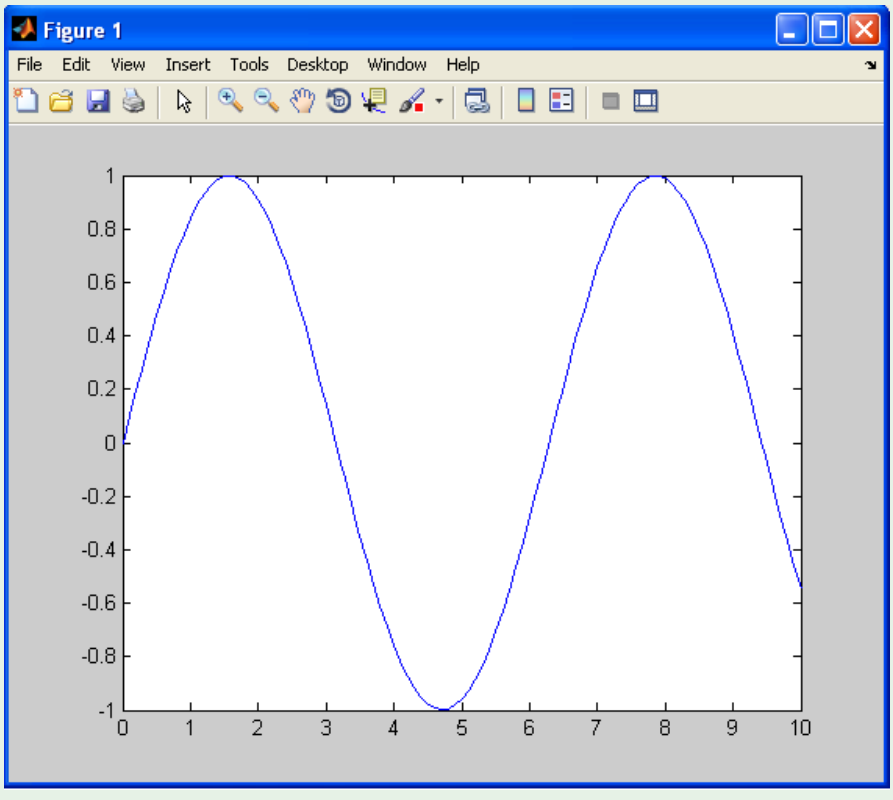
- `plot(x,y)` siendo x e y vectores de igual dimensión

Ejemplo

```
grid;
x=0:0.1:10;
y=sin(x);
plot(x,y);
```


Realizar gráficas 2D

Resultado gráfica



Realizar gráficas 2D

Comando bar

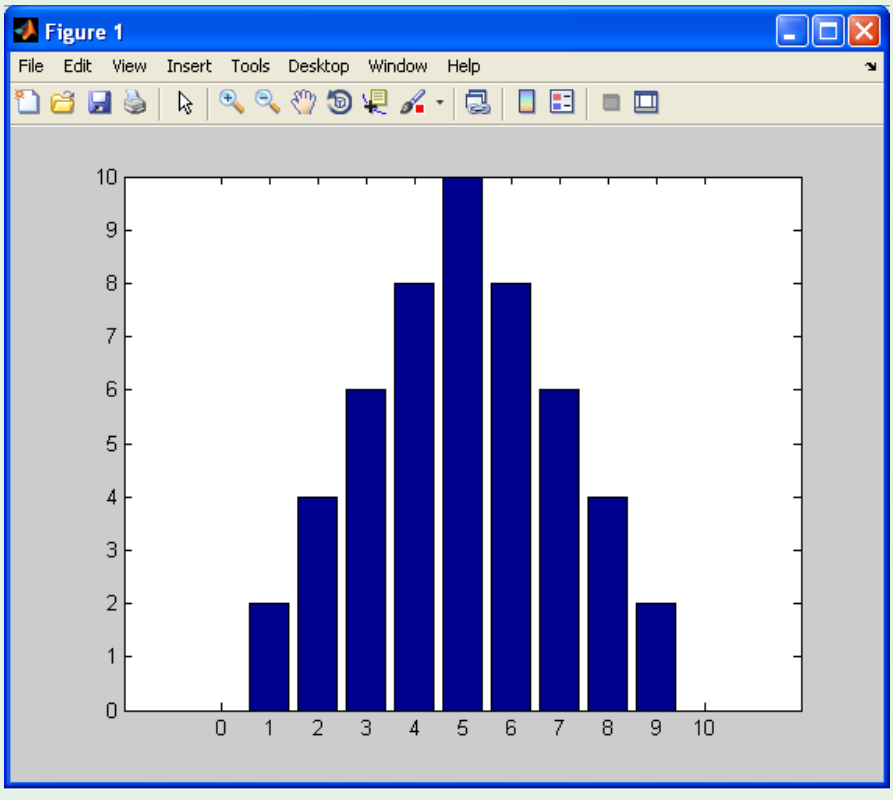
- `bar(x,y)` siendo x e y vectores de igual dimensión

Ejemplo

```
x=0:10;
y=[0 2 4 6 8 10 8 6 4 2 0];
bar(x,y)
```

Realizar gráficas 2D

Resultado gráfica



Formato de linea

`plot` con formato

`plot(x,y,'nm')`
`'nm'` asigna un color y un estilo de punto, donde `n` representa un color y `m` el estilo del punto.

Tipos de linea

- `y` amarillo, `m` magenta, `c` cyan, `r` rojo, `g` verde, `b` azul, `w` blanco, `k` negro
- '–' linea sólida (opción por defecto).
- `'.'` puntos
- `'o'` cículos
- `'x'` equis
- `'+'` cruces
- `'–.'` guión y punto

Graficas: tipos linea

Ejemplo 1

```
t=0:0.1:4.5;
plot(t, cos(t), 'g*', t , sin(t), 'b')
```

Ejemplo 2

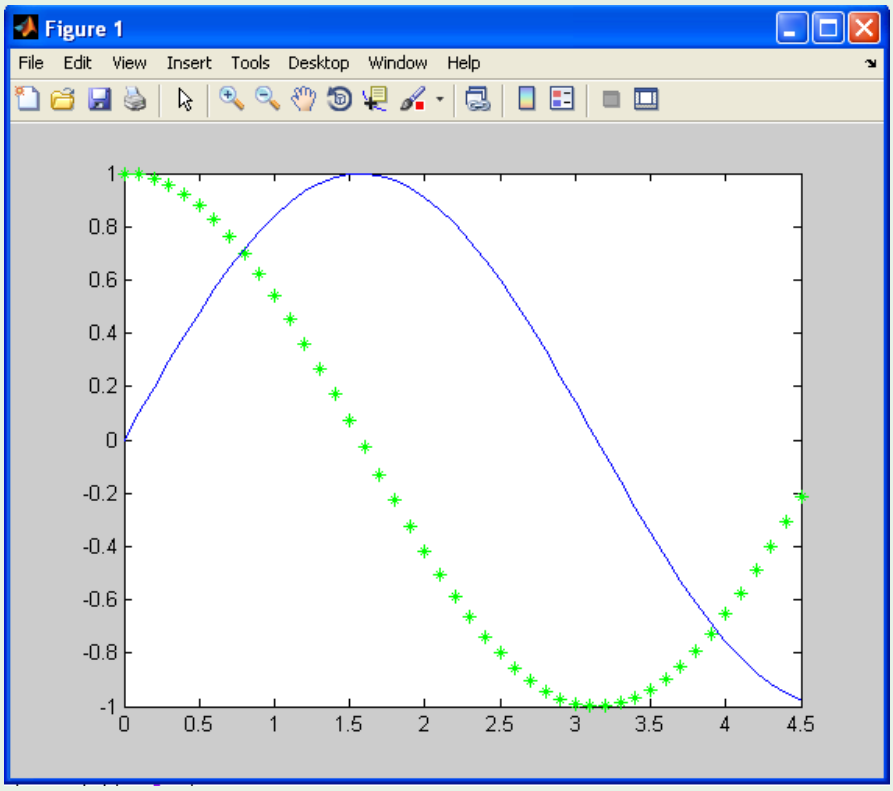
```
t=0:0.1:4.5;
plot(t, cos(t), 'g*', t , sin(t), 'b')
legend('Cos(t)', 'Sen(t)')
```

Ejemplo 3

```
t=0:0.1:4.5;
plot(t, cos(t), 'g*', t , sin(t), 'b')
legend('Cos(t)', 'Sen(t)')
title('Gráfico seno y coseno')
xlabel('Tiempo')
ylabel('Amplitud')
```

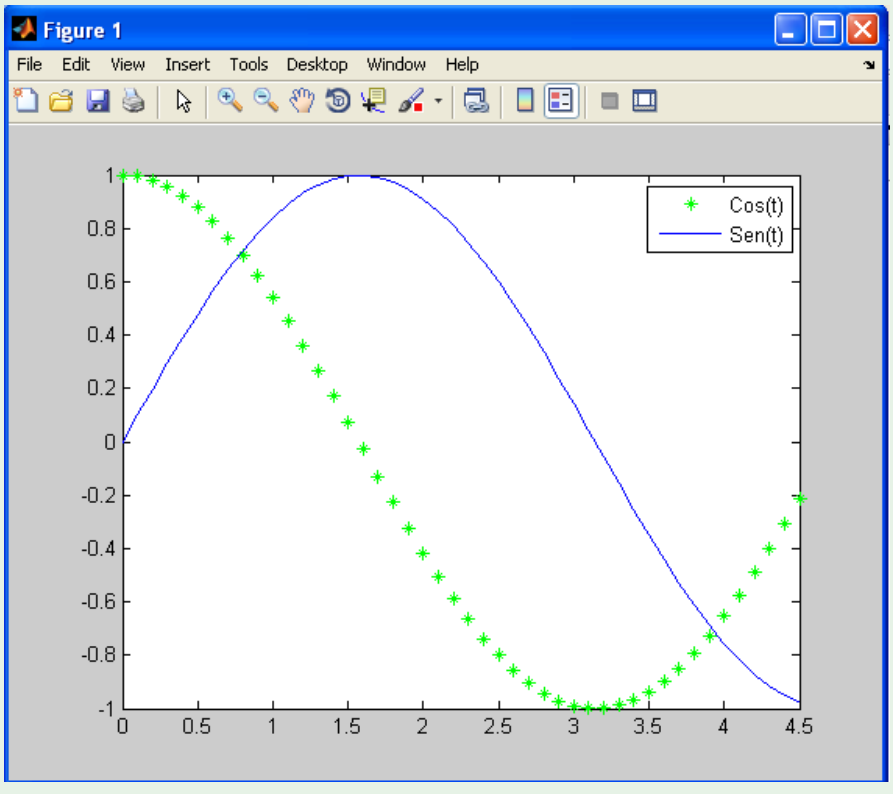
Graficas: tipos linea

Resultado ejemplo 1



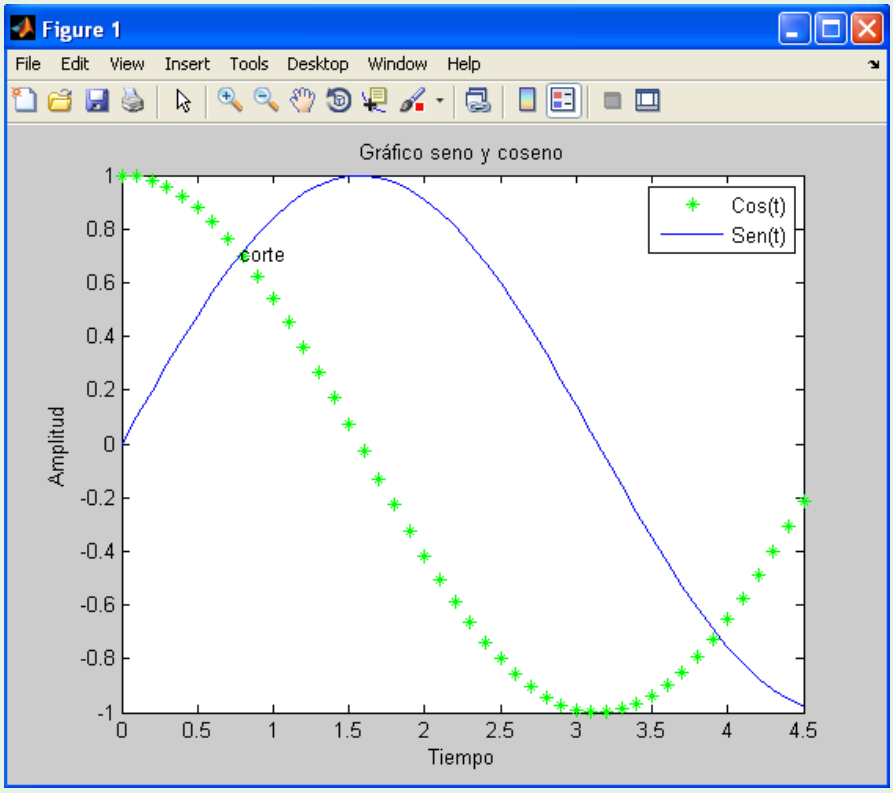
Graficas: tipos linea

Resultado ejemplo 2



Graficas: tipos linea y etiquetas

Resultado ejemplo 3



Gráficas sobre otra

hold

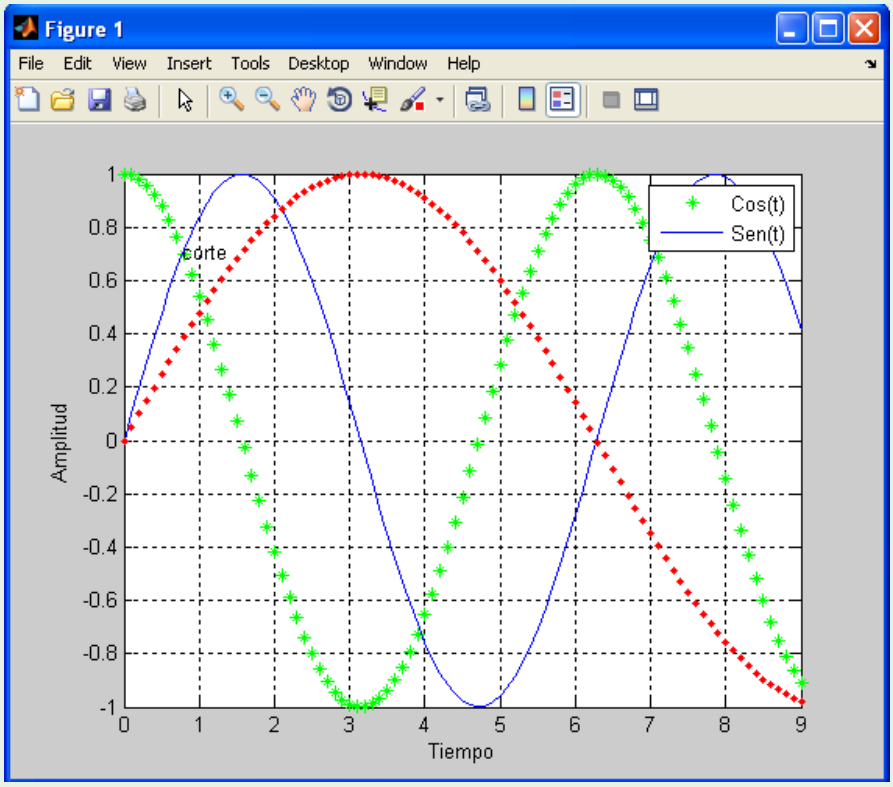
`hold` permite dibujar un gráfico sobre otro.

Ejemplo `hold`

```
t=0:0.1:4.5;
plot(t, cos(t), 'g*', t , sin(t), 'b')
legend('Cos(t)', 'Sen(t)')
title('Gráfico seno y coseno')
xlabel('Tiempo')
ylabel('Amplitud')
text(pi/4,0.707,'corte')
grid;
hold;
plot(t, sin(t/2), 'r.');
```

Gráficas sobre otra

Resultado ejemplo `hold`



Guardar gráficas

Manualmente

Con el botón de la interfaz se puede copiar al portapapeles..
Soporta diversos formatos.

Por comando: `print`

Para ver todas las posibilidades: `doc print`

Ejemplo

```
print -depsc -tiff grafica
print -dpdf grafica
print -dpng -r300 grafica
```

Guarda la grafica en formato eps (formato vectorial), en formato pdf y en formato png (con 300ppp).

Gráficas 3D

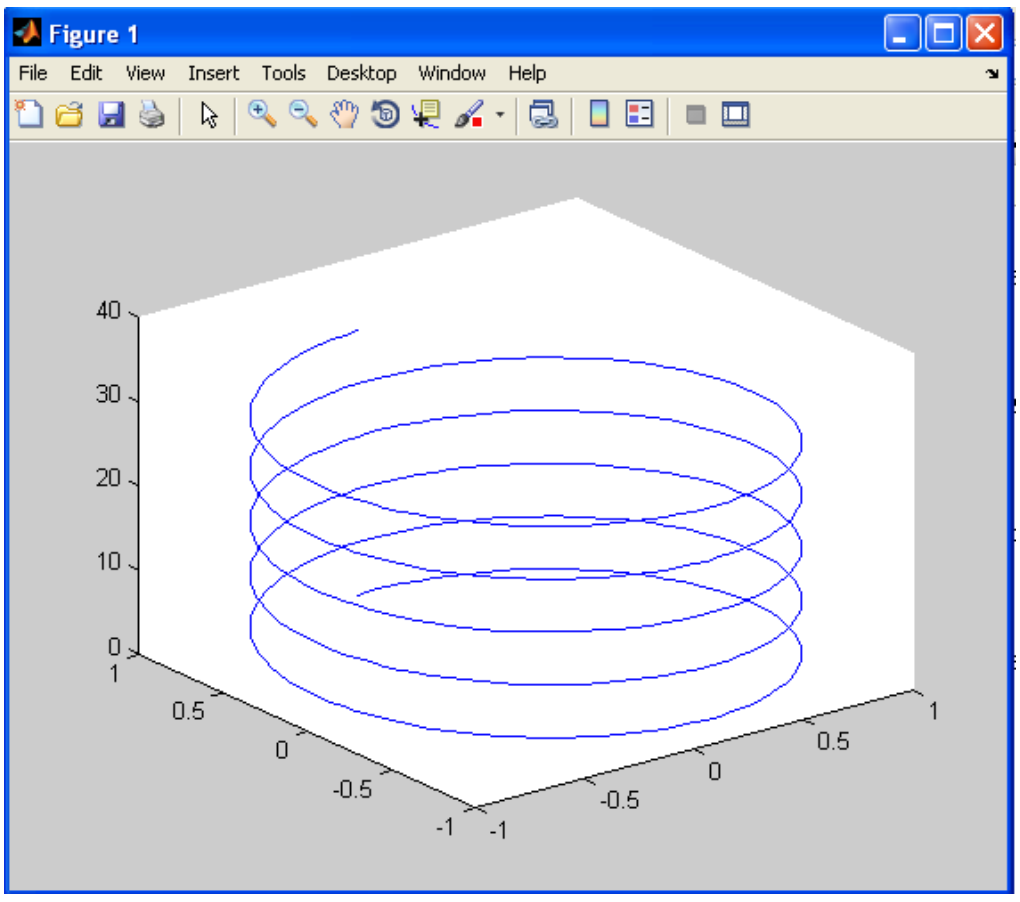
Ejemplo: gráfica línea 3D

```
t=0:pi/50:10*pi;
plot3(sin(t),cos(t),t)
```

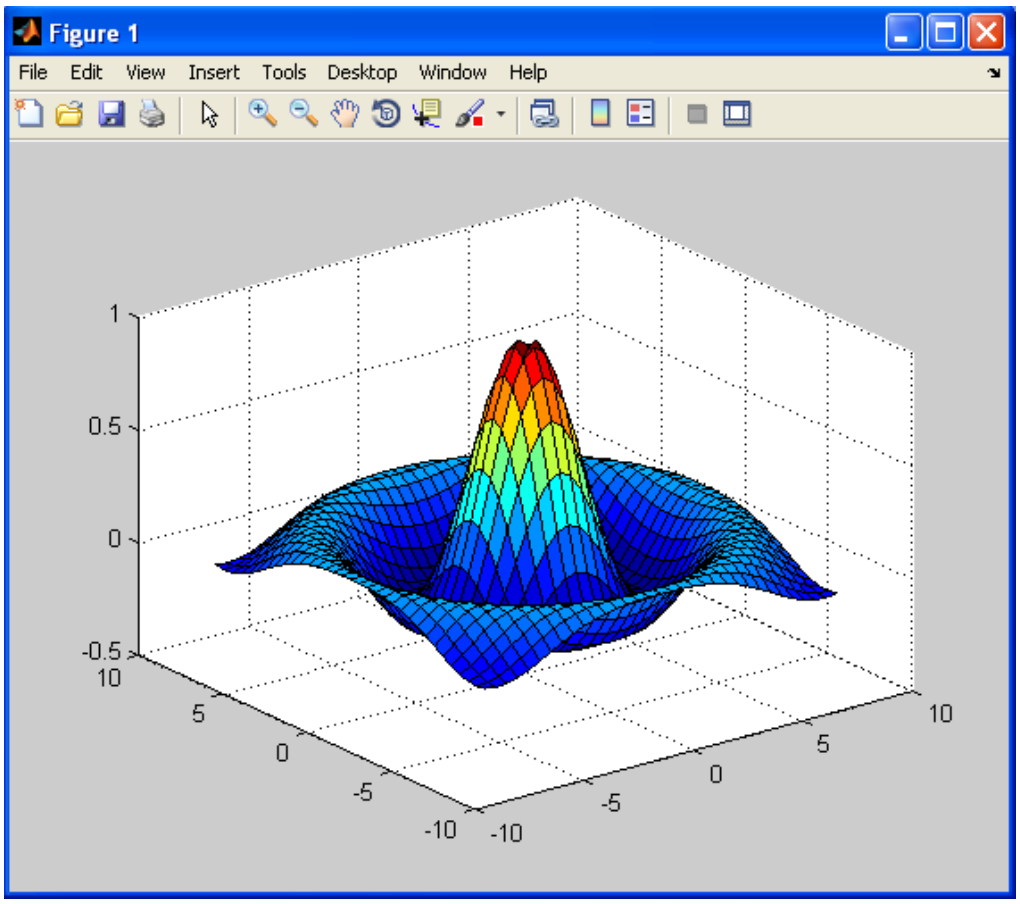
Ejemplo: gráfica superficie 3D

```
[X,Y]=meshgrid(-8:0.5:8);
Z=sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
surf(X,Y,Z)
```


Gráfica línea en 3d



Gráfica superficie 3D



Diferencias entre Octave y Matlab en gráficas

- La sintaxis de Matlab funciona en octave.
- Octave tiene otra sintaxis que también funciona.

Definición

Conjunto agrupado de funciones y comandos para unas tareas específicas.

Ejemplos

Se pueden ver en la ayuda: `Help - Function browser`

- Matemáticas
  - Álgebra lineal:
    - Resolución de sistemas de ecuaciones lineales
    - Autovalores y autovectores
    - Factorización
  - Trigonométricas, exponenciales, complejas
  - Polinomios:
    - Raices de un polinomio
    - Obtener un polinomio para una raices dadas
  - Conversión entre sistemas de coordendas
  - Ecuaciones diferenciales
  - Integración numérica
  - ...

Ejemplos

- Análisis de datos
  - Operaciones básicas:
    - Ordenación de datos
  - Estadística
    - Media, Mediana, Máximos y mínimos
  - Convolución
  - Interpolación y regresión
  - Transformada de fourier
  - Derivadas e integrales
  - ...
- Procesado de señales
- Procesado de imagenes
- Procesado de sonido
- ...

Ejemplos de funciones de librerías I

Ejemplo: Autovalores y autovectores: `eig`

```
A=[ 2 0 1 ; 0 4 0; 1 0 2]
 A =  2     0     1
      0     4     0
      1     0     2

eig(A)
 ans =      1
            3
            4

[V,D]=eig(A)
 V =
     0.7071    0.7071         0
          0         0   -1.0000
    -0.7071    0.7071         0
  D =
      1     0     0
      0     3     0
      0     0     4
```

Ejemplos de funciones de librerías II

Ejemplo: Redondeo

```
>> B=[1.34 3.56 6.89 2.5]

B =
    1.3400    3.5600    6.8900    2.5000

>> ceil(B)
ans =
     2     4     7     3

>> fix(B)
ans =
     1     3     6     2

>> floor(B)
ans =
     1     3     6     2
```

Ejemplos de funciones de librerías III

Ejemplo: Raices de polinomios

$$2 \cdot x^2 + x - 2$$

```
>> P1=[2 1 -2];
>> roots(P1)
ans =  -1.2808
        0.7808
```

Ejemplo: Raices de polinomios

$$3 \cdot x^4 + x^3 + 2 \cdot x^2 + x - 2$$

```
>> P2=[3 1 2 1 -2];
>> roots(P2)
ans =  -0.9045
       -0.0196 + 1.0986i
       -0.0196 - 1.0986i
        0.6105
```


Ejemplos de funciones de librerías IV

Ejemplo: Integración numérica

Creamos la función parab.m con el editor:

```
function y = parab(x)
  y = x.^2;
endfunction
```

Hacemos la integral numérica:

```
>> quad(@parab,0,1)
ans =    0.3333
```

Ejemplo: Integración numérica (con funcion definida de forma compacta

```
>> F = @(x)(x.^2);
>> Q = quad(F,0,1)
Q =    0.3333
```

Ejemplo procesamiento de imágenes

Ejemplo procesamiento de imágenes

1. Imagen RGB original

2. Transformacion a escala de grises

3. Visualizacion de Banda ROJA original

Comando `input`

Ejemplo: input

```
>>> a=input('Dame un número ');
Dame un número 55

>>> a

a =

    55
```

Una vez tenemos el número introducido por el usuario en una variable los podemos usar como una variable normal del programa.
Se pueden introducir números, cadenas, vectores...

Varios comandos: `tic toc`

En octave es más eficiente usar operaciones sobre vectores o matrices que realizar bucles. Los comandos `tic toc` dan el tiempo de cálculo

Ejemplo: bucle

```
x=1:0.001:10;
for i=1:length(x)
  y(i)=sin(x(i)^2)+cos(x(i)^3);
end
```

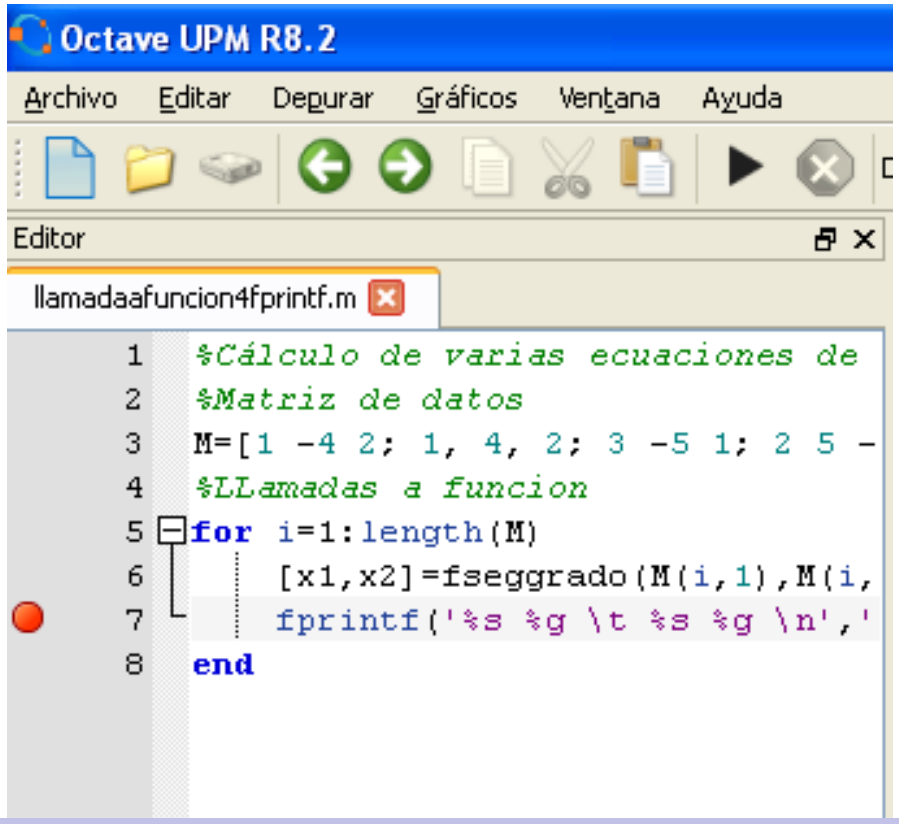
Ejemplo: array

```
x=1:0.001:10;
y=sin(x.^2)+cos(x.^3);
```

0.2466 segundos vs 0.0047 segundos

Debug / depurar

Es posible establecer puntos de parada e ir viendo la evolución de las variables.



Ayuda y documentación

- Manual Octave (html)
- Octave refcard.
- El editor tiene autompletado de funciones.
- Comando `help`. Muestra la ayuda para ese comando. Ejemplo: `help atan`
- Comando `doc`. Muestra una ayuda más extensa para ese comando. Ejemplo: `doc disp`
- Videos de Israel Herraiz
- Ayuda Matlab (online)
- Ejemplos en videotutorial de la web de Matlab

Bibliografía

- *GNU Octave*. (Manual en pdf)
- *Manual de iniciación a GNU Octave*. J.M Valiente Cifuentes.
- *Matlab y matemática computacional*. Sagrario Lantarón Sánchez, Bernardo Lanas. Ed. Belisco Ediciones.
- *Matlab y sus aplicaciones en las ciencias y la ingeniería*. César Pérez. Ed Pearson Prentice Hall. 2003
- *Cálculo científico con Mathlab y Octave*. A.Quarteroni, F.Saleri. Ed Springer. 2006.
- *Mástering Matlab*. Duane Hanselman, Bruce Littlefield. Ed Pearson Prentice Hall. 2005