– **Typeset by GMNI &** FoilTEX –

Tutorial de Matlab

TUTORIAL DE MATLAB

J. París, H. Gómez, F. Navarrina, I. Colominas, X. Nogueira, M. Casteleiro

CÁLCULO NUMÉRICO

Departamento de Métodos Matemáticos y de Representación
Escuela Técnica Superior de Ingenieros de Caminos, Canales y Puertos
Universidade da Coruña

Tutorial de Matlab 7.0

Tutorial de Matlab

- Introducción
- Entorno de trabajo
- Variables en Matlab
- Operaciones básicas
- Dibujo de funciones en 2D y 3D
- Programación en Matlab
- Ayuda de Matlab
- Ejemplos prácticos

Introducción

- ♡ Matlab es el nombre abreviado de Matrix Laboratory
- ♡ Es un programa que permite realizar cálculos con matrices y vectores
- ♡ También permite realizar gráficos de forma sencilla
- ♡ Dispone de lenguaje de Programación propio
- ♠ No se utilizará en la asignatura como lenguaje de programación
- ♠ El programa del trabajo de curso debe hacerse en lenguaje Fortran

Entorno de trabajo (I)

- El entorno de trabajo presenta tres ventanas:

"Launch Pad"	Ventana que da acceso a todos los módulos de Matlab
"Workspace"	Contiene e indica todas las variables de cada sección
"Command window"	Ventana donde se introducen los comandos de ejecución
"Command history"	Muestra las últimas instrucciones ejecutadas en la ventana de comandos
"Current directory"	Indica el directorio actual de trabajo donde se encuentran programas y funciones

Entorno de trabajo (II)

Tutorial de Matlab

MATLAB

File Edit View Web Window Help

Current Directory: C:\MATLAB6p1\work

Workspace

Stack: Base

Name	Size	Bytes	Class

Launch Pad | Workspace

Command History

```
X2((i-1)*21+j))=.X(i,j);
Y2((i-1)*21+j))=.Y(i,j);
end
end
for i=1:21
for j=1:21
X2((i-1)*21+j)=X(i,j);
Y2((i-1)*21+j)=Y(i,j);
end
end
mesh(X2.^2+Y2.^4)
plot3(X2,Y2,X2.^2+Y2.^4)
mesh(X.^2+Y.^4)
mesh(X2.^2+Y2.^4)
exit
%--  1:06 PM 10/01/07 --%

clear
```

Command History | Current Directory

Command Window

>>

Ready

Comandos básicos

`clear`:	elimina las variables almacenadas anteriormente
`clc`:	elimina todas las salidas anteriores y limpia la ventana de comandos
`home`:	limpia la ventana de comandos
`clear all`:	borra todas las variables

En Matlab existen dos formas principales de trabajar

- De modo interactivo sobre la ventana de comandos
- A través de scripts (archivos *.m)

- La forma más sencilla de trabajar es de modo interactivo
- Si se desean hacer pequeños programas o aplicaciones se recomienda crear scripts (*.m)

Variables

- Todas las variables numéricas se almacenan como reales en doble precisión (8 bytes)
- La forma de representación por pantalla puede ser de tipo:

`short`:	Coma fija con 4 decimales (defecto)
`long`:	Coma fija con 15 decimales
`hex`:	Cifras hexadecimales
`bank`:	números con 2 decimales
`short e`:	notación científica, 4 decimales
`short g`:	notación científica o decimal, dependiendo del valor
`long e`:	notación científica, 15 decimales
`long g`:	notación científica o decimal, dependiendo del valor
`rat`:	números racionales como cociente de enteros

- Para cambiar el formato de los números: >> `format` "formato"
- Las líneas que comienzan por "%" son comentarios
- Las líneas que finalizan con ";" no muestran resultados por pantalla

Operaciones básicas

Definición de vectores

`>> x=[10 20 30]` $\rightarrow$ (10 20 30)

`>> x=[10; 20; 30]` $\rightarrow \begin{pmatrix} 10 \\ 20 \\ 30 \end{pmatrix}$

Definición de matrices

`>> A=[1 2 3; 4 5 6]` $\rightarrow \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$

`>> A=[[1 2 3]; [4 5 6]]` $\rightarrow \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$

Operadores algebraicos

`+`	Suma	`.*:`	Producto elemento a elemento
`-`	Resta	`./:`	División derecha elemento a elemento
`*`	Multiplicación	`.\:`	División izquierda elemento a elemento
`'`	Transpuesta	`.^:`	Potencia elemento a elemento
`^`	Potencia		
`\`	División izqda.	$\rightarrow$	`x=A\b` $\Rightarrow$ `x=inv(A)*b`
`/`	División dcha.		

Operador ":" Representa de forma general un rango

`x=1:2:9` $\rightarrow$ x=(1 3 5 7 9)

`x=A(:,2)` $\rightarrow$ almacena en el vector `x` la segunda columna de la matriz `A`

Gráficos 2D

- Comandos de dibujo:
  - `plot()`: Crea un gráfico con ejes lineales
  - `loglog()`: Crea un gráfico con escala logarítmica en los ejes coordenados
  - `semilogx()`: Crea un gráfico con escala logarítmica en el eje x
  - `semilogy()`: Crea un gráfico con escala logarítmica en el eje y
  - `close`: Cierra la ventana gráfica anterior
- Ejemplos:
  - `plot(A)`: Dibuja una curva representando los valores de las columnas de la matriz A en ordenadas frente al índice del elemento en abscisas
  - `plot(x,A)`: Igual que el anterior, pero en abscisas utiliza los valores de `x`
  - `plot(A,B)`: Dibuja las funciones obtenidas de representar en ordenadas las columnas de B y en abscisas las columnas de A

Dibujo de funciones en 2D y 3D (II)

Complementos del dibujo:

`title('titulo')`	Introduce un título en el gráfico dibujado anteriormente
`xlabel('etiqueta eje x')`	Introduce una etiqueta para las variables del eje x
`ylabel('etiqueta eje y')`	Idem que el anterior para el eje y
`axis('square')`	Genera un gráfico cuadrado
`axis([xmin,xmax,ymin,ymax])`	Genera un gráfico con los límites que se indican en los ejes

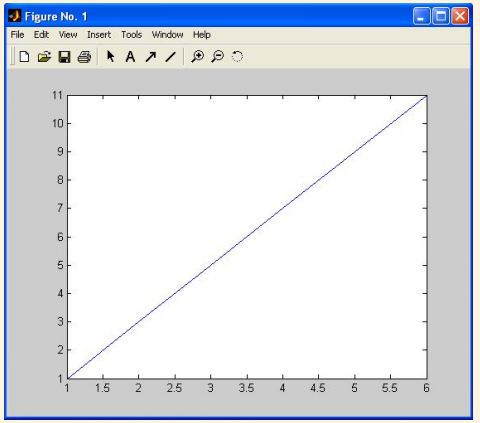
Otras funciones gráficas:

- `bar(x)` Genera un diagrama de barras con el vector x
- `pie(x)` Genera un diagrama de sectores circulares
- `hist(x)` Dibuja un histograma con el vector x en 3D
- `rose(x)` Dibuja un histograma de (ángulos en radianes)

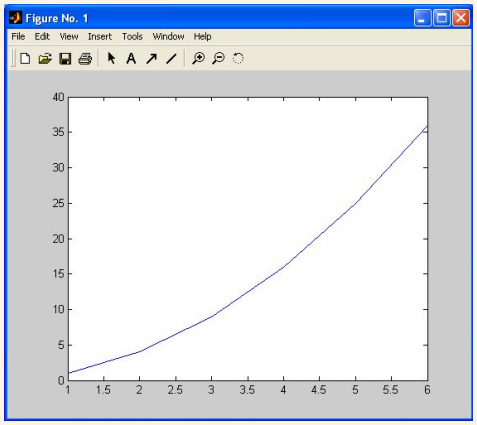
Dibujo de funciones en 2D y 3D (III)

Ejemplos prácticos:

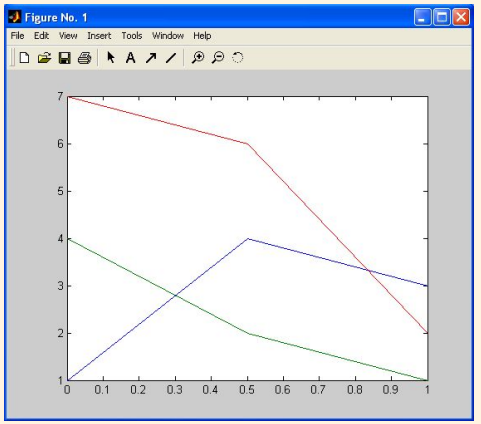
```
>> x=[1 3 5 7 9 11]
>> plot(x)
```




```
>> x=[1 2 3 4 5 6]
>> y=[1 4 9 16 25 36]
>> plot(x,y)
```




```
>> x=0:0.5:1
>> A=[1 4 7; 4 2 6; 3 1 2]
>> plot(x,A)
```



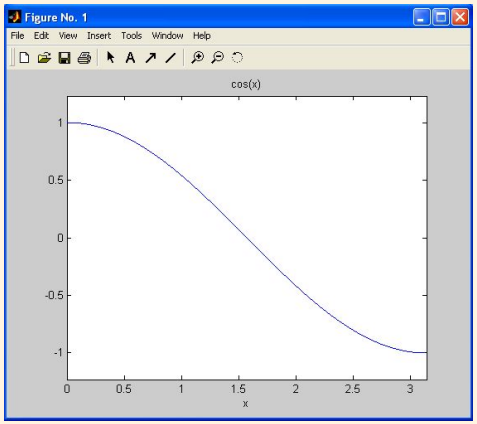
Dibujo de funciones en 2D y 3D (IV)

Comandos de dibujo de funciones

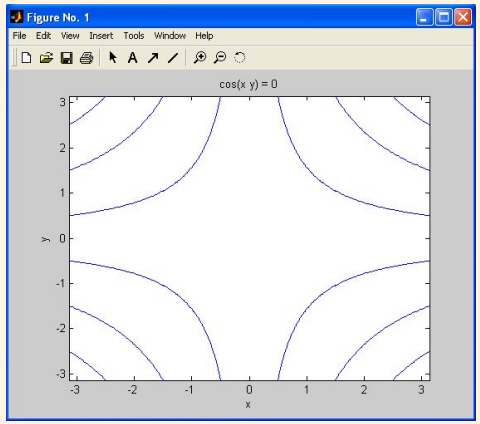
`ezplot(f,a,b)`:	Dibuja la función f(x) en el intervalo $a \leq x \leq b$ Si se omite el intervalo, por defecto es $-2\pi \leq x \leq 2\pi$
`ezplot(f,[xmin,xmax,ymin,ymax]`	Dibuja la función f(x,y)=0 en el intervalo $xmin \leq x \leq xmax$ e $ymin \leq y \leq ymax$

Ejemplos:

```
>> ezplot('cos(x)',0,pi)
```




```
>> ezplot('cos(x*y)',[-pi,pi,-pi,pi])
```



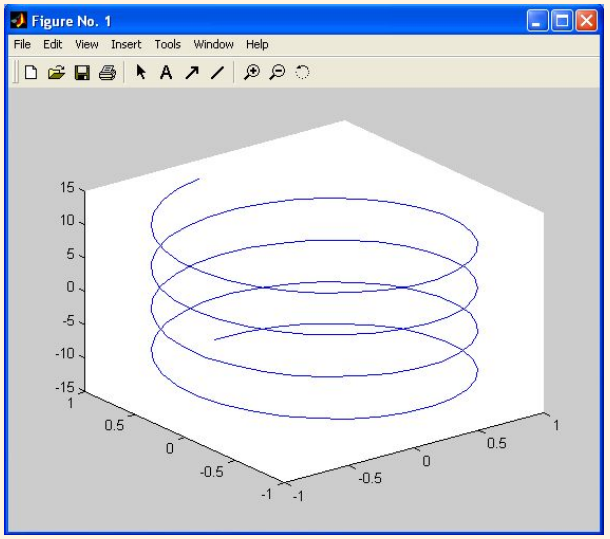
Gráficos 3D

- Comandos de dibujo de líneas 3D

`plot3(x,y,z)`: Dibuja la función que pasa por los puntos de coordenadas x, y, z

Por ejemplo,

```
>> x=[-4*pi:0.1:4*pi];
>> plot3(sin(x),cos(x),x)
```



Dibujo de funciones en 2D y 3D (VI)

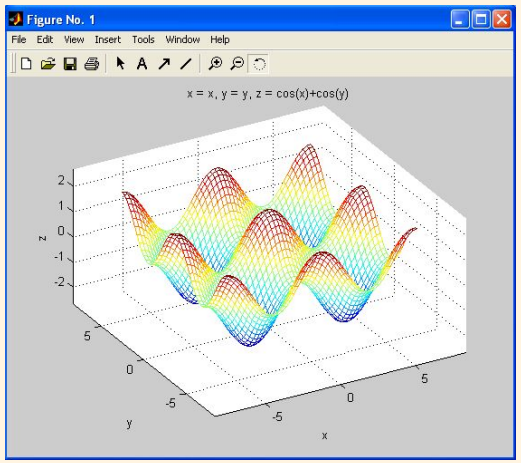
- Comandos de dibujo de funciones en 3D

```
ezmesh(x,y,z,[smin,smax,tmin,tmax])
```

Dibuja la superficie de coordenadas (x,y,z) siendo x=x(s,t), y=(s,t), z=z(s,t)
De modo similar se pueden utilizar las funciones `ezsurf` y `ezcontour`

Ejemplo:

```
>> ezmesh('x','y','cos(x)+cos(y)',[-2*pi,2*pi,-2*pi,2*pi])
```



Dibujo de funciones en 2D y 3D (VII)

Generación de mallas de puntos equiespaciados

meshgrid Genera una malla rectangular de puntos a partir de las coordenadas de puntos en `x` y en `y`

Por ejemplo,

```
>> x=-2:0.2:2; y=x
>> [X,Y]=meshgrid(x,y)
```

Comandos de dibujo de superficies 3D

mesh(Z) genera una malla en modo alámbrico en 3D con la función Z a partir de la malla original

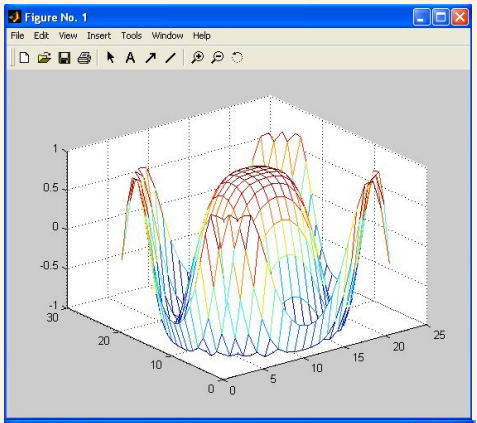
surf(Z) genera una superficie 3D con la función Z a partir de la malla original

contour(Z) genera una imagen bidimensional con las curvas de nivel correspondientes a la superficie tridimensional

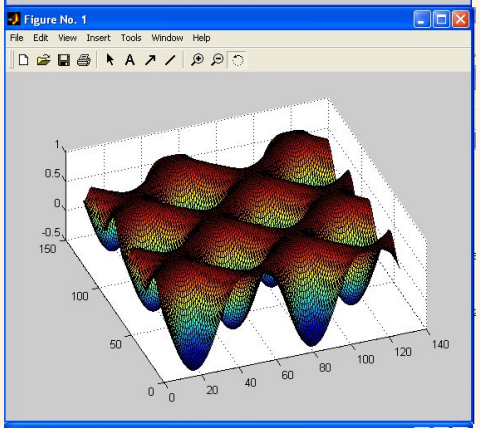
Dibujo de funciones en 2D y 3D (VIII)

Por ejemplo,

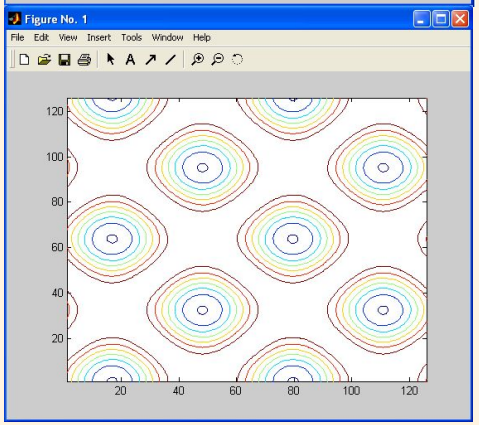
```
>> x=-2:0.1:2; y=x;
>> [X,Y]=meshgrid(x,y);
>> Z=cos(X.^2+Y.^2);
>> mesh(Z)
```




```
>> x=-2*pi:0.1:2*pi; y=x;
>> [X,Y]=meshgrid(x,y);
>> Z=cos(sin(X)+cos(Y));
>> surf(Z)
```




```
>> x=-2*pi:0.1:2*pi; y=x;
>> [X,Y]=meshgrid(x,y);
>> Z=cos(sin(X)+cos(Y));
>> contour(Z)
```



Programación en Matlab (I)

- Además de las operaciones de modo interactivo podemos crear programas
- Los programas en Matlab se definen como secuencias de operaciones (script)
- El código de programación se implementa en archivos *.m
- La ejecución de los programas se realiza:
  - Seleccionando el directorio que contiene el programa como directorio de trabajo
  - Ejecutar el archivo *.m tecleando su nombre en la línea de comandos.
- Además los programas pueden llamar a funciones (archivos *.m) que actúan como subrutinas
- Estas funciones pueden utilizarse en numerosas ocasiones dentro de un programa

Funciones en Matlab

```
function [variables_salida]=nombre_función (variables_entrada)
...
comandos
...
return
```

Por ejemplo,

```
function z=modulo(x,y)
z=sqrt(x*x+y*y)
return
```

Guardado en un archivo de nombre "modulo.m"

- Para llamar a una función desde un archivo de comandos (*.m):

```
>> [variables_salida]=nombre_función (variables_entrada)
```

Sentencias de Programación (I)

IF Realiza las sentencias cuando se cumplan las condiciones establecidas

```
if condición_1
 sentencia_1
elseif condición_2
 sentencia_2
else
 sentencia_por_defecto
end
```

```
if A==B
  print*,A
else
  B=A\B
  print*,B
end
```

- Operadores relacionales
  - < menor que (elemento a elemento)
  - > mayor que (elemento a elemento)
  - <= menor o igual que (elemento a elemento)
  - >= mayor o igual que (elemento a elemento)
  - == igual elemento a elemento (elemento a elemento)
  - ~= distinto elemento a elemento (elemento a elemento)

Sentencias de Programación (II)

FOR Repite un conjunto de sentencias un número determinado de veces

```
for i=1:n
  sentencia_1
end
```

```
for ind=1:10
  x(ind)=ind*ind;
end
```

WHILE Repite un conjunto de sentencias mientras se cumpla la condición

```
while condicion
  sentencia
end
```

```
ind=1;
while ind<=1000
  disp(x(ind)); ind=ind+1;
end
```

BREAK Termina la ejecución del bucle más interno

CONTINUE Salta a la siguiente iteración del bucle

Entrada y salida de datos por pantalla

input Permite escribir un mensaje en la línea de comandos y recibir el valor de una variable

```
>> n=input('Indique el numero de puntos')
```

disp Permite representar mensajes o variables por pantalla

```
>> disp('Solucion alcanzada')
```

Formatos de salida de variables:

`%s`	cadenas de caracteres. Por ejemplo, `%10s`
`%d`	número enteros. Por ejemplo, `%5d`
`%f`	números en coma flotante. Por ejemplo, `%10.4f`
`%lf`	números en doble precisión. Por ejemplo, `%10.4lf`

Lectura y escritura de ficheros

fopen: Abre un archivo para leer o escribir datos

`>> [`*log_unidad*, *texto_error*`]=fopen('`*nombre_archivo*`',`*permisos*`)`

log_unidad	variable que almacena el número de unidad lógica
texto_error	variable de texto que guardará mensajes de error si los hay
nombre_archivo	nombre del archivo ASCII incluida su extensión
permisos	`'r'` lectura `'w'` escritura `'a'` escritura a continuación de lo ya existente `'r+'` lectura y escritura

fclose: Cierra un archivo abierto

`>> [`*texto_error*`]=fclose(`*log_unidad*`)`

Funciones de lectura y escritura en archivo ASCII

fscanf: lee datos de un archivo previamente abierto

`[`*var1, var2, ...*`]=fscanf(`*log_unidad*`,'`*cadena_de_control*`',`*tamaño*`)`

var1, var2, ...	son las variables en las que se almacenan los datos
cadena_de_control	indica los formatos de las variables que se van a leer
tamaño	(opcional) tamaño de las variables a leer

Ejemplo: `>> [x,y,z]=fscanf(log_unit1,'%10.4f,%10.4f,%10.4f',3)`

fprintf: escribe datos en un archivo previamente abierto

`fprintf(`*log_unidad*`,'`*cadena_de_control*`',`*var1, var2, ...*`)`

Ejemplo: `>> fprintf(log_unit2,'%10.4f,%10.4f,%10.4f',x,y,z)`

Ayuda de Matlab

La ayuda de Matlab puede activarse mediante:

`>> help` *tópico*

Por ejemplo, `>> help plot3`

Para obtener más información:

- Colección: "Aprenda Matlab como si estuviera en Primero" Javier García de Jalón
- "Análisis Numérico y visualización gráfica con Matlab" Shoichiro Nakamura. Ed. Pearson Educación

Ejemplos prácticos (I)

Archivo de datos y dibujo de superficie (datos1.txt, malla1.m)

```
   4    7

0.00000E+00 0.39250E+00 0.78500E+00 0.11775E+01 0.15700E+01
0.00000E+00 0.39250E+00 0.78500E+00 0.11775E+01 0.15700E+01
0.00000E+00 0.39250E+00 0.78500E+00 0.11775E+01 0.15700E+01
0.00000E+00 0.39250E+00 0.78500E+00 0.11775E+01 0.15700E+01
0.00000E+00 0.39250E+00 0.78500E+00 0.11775E+01 0.15700E+01
0.00000E+00 0.39250E+00 0.78500E+00 0.11775E+01 0.15700E+01
0.00000E+00 0.39250E+00 0.78500E+00 0.11775E+01 0.15700E+01
0.00000E+00 0.39250E+00 0.78500E+00 0.11775E+01 0.15700E+01

0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
0.44857E+00 0.44857E+00 0.44857E+00 0.44857E+00 0.44857E+00
0.89714E+00 0.89714E+00 0.89714E+00 0.89714E+00 0.89714E+00
0.13457E+01 0.13457E+01 0.13457E+01 0.13457E+01 0.13457E+01
0.17943E+01 0.17943E+01 0.17943E+01 0.17943E+01 0.17943E+01
0.22429E+01 0.22429E+01 0.22429E+01 0.22429E+01 0.22429E+01
0.26914E+01 0.26914E+01 0.26914E+01 0.26914E+01 0.26914E+01
0.31400E+01 0.31400E+01 0.31400E+01 0.31400E+01 0.31400E+01

0.20000E+01 0.19240E+01 0.17074E+01 0.13832E+01 0.10008E+01
0.19011E+01 0.18250E+01 0.16085E+01 0.12843E+01 0.90186E+00
0.16238E+01 0.15478E+01 0.13312E+01 0.10071E+01 0.62464E+00
0.12232E+01 0.11471E+01 0.93057E+00 0.60642E+00 0.22398E+00
0.77837E+00 0.70232E+00 0.48575E+00 0.16160E+00-0.22084E+00
0.37740E+00 0.30136E+00 0.84788E-01-0.23936E+00-0.62180E+00
0.99624E-01 0.23580E-01-0.19299E+00-0.51714E+00-0.89958E+00
0.12683E-05-0.76043E-01-0.29261E+00-0.61676E+00-0.99920E+00
```

```
clear
[fi,txterr]=fopen('datos1.txt','r');
[n]=fscanf(fi,'%5d',2);
nx=n(1);
ny=n(2);
fscanf(fi,'%s',0);
for i=1:(ny+1)
  [a]=fscanf(fi,'%f',nx+1);
  AX(i,:)=a';
end
fscanf(fi,'%s',0);
for i=1:(ny+1)
  [a]=fscanf(fi,'%f',nx+1);
  BX(i,:)=a';
end
fscanf(fi,'%s',0);
for i=1:(ny+1)
  [a]=fscanf(fi,'%f',nx+1);
  CX(i,:)=a';
end
st=fclose(fi);
surf(AX,BX,CX);
axis('normal')
```

Ejemplos prácticos (II)

Tutorial de Matlab

Lectura de archivo y dibujo de funciones (datos2.txt, malla2.m)

```
 6    3

0.000000E+00
0.523333E+00
0.104667E+01
0.157000E+01
0.209333E+01
0.261667E+01
0.314000E+01

0.000000E+00
0.499770E+00
0.865760E+00
0.100000E+01
0.866556E+00
0.501149E+00
0.159265E-02

0.000000E+00
0.999540E+00
0.173152E+01
0.200000E+01
0.173311E+01
0.100230E+01
0.318531E-02

0.000000E+00
0.149931E+01
0.259728E+01
0.300000E+01
0.259967E+01
0.150345E+01
0.477796E-02
```

```
clear
%
[fi,txterr]=fopen('datos2.txt','r');
%
[n]=fscanf(fi,'%5d',2);
nx=n(1);
nf=n(2);
%
fscanf(fi,'%s',0);
%
for j=1:nx+1
  [x(j)]=fscanf(fi,'%f',1);
end
%
fscanf(fi,'%s',0);
%
for i=1:nf
  for j=1:nx+1
    [CX(j,i)]=fscanf(fi,'%f',1);
  end
  fscanf(fi,'%s',0);
end
%
st=fclose(fi);
%
plot(x,CX);
axis([0,x(nx+1),-(nf+1),nf+1]);
title('Dibujo de funciones seno(x)');
xlabel('Posicion x');
ylabel('Amplitud');
```

Ejemplos prácticos (III)

Tutorial de Matlab

Lectura de archivo y dibujo de superficies (datos3.txt, malla3.m)

```
  2    5

 0.396633E-12   0.793266E-12   0.118990E-11
-0.951057E+00  -0.190211E+01  -0.285317E+01
-0.587780E+00  -0.117556E+01  -0.176334E+01
 0.587792E+00   0.117558E+01   0.176338E+01
 0.951053E+00   0.190211E+01   0.285316E+01
-0.146928E-04  -0.293856E-04  -0.440785E-04

 0.100000E+01   0.200000E+01   0.300000E+01
 0.309014E+00   0.618028E+00   0.927043E+00
-0.809020E+00  -0.161804E+01  -0.242706E+01
-0.809012E+00  -0.161802E+01  -0.242704E+01
 0.309028E+00   0.618056E+00   0.927085E+00
 0.100000E+01   0.200000E+01   0.300000E+01

 0.157080E+01   0.314159E+01   0.471239E+01
 0.282744E+01   0.565487E+01   0.848231E+01
 0.408408E+01   0.816815E+01   0.122522E+02
 0.534072E+01   0.106814E+02   0.160221E+02
 0.659736E+01   0.131947E+02   0.197921E+02
 0.785400E+01   0.157080E+02   0.235620E+02
```

```
clear
%
[fi,txterr]=fopen('datos3.txt','r');
%
[n]=fscanf(fi,'%5d',2);
nr=n(1);
nt=n(2);
%
fscanf(fi,'%s',0);
%
for j=1:nt+1
  [a]=fscanf(fi,'%f',nr+1);
  AX(j,:)=a';
end
%
fscanf(fi,'%s',0);
%
for j=1:nt+1
  [a]=fscanf(fi,'%f',nr+1);
  BX(j,:)=a';
end
%
fscanf(fi,'%s',0);
%
for j=1:nt+1
  [a]=fscanf(fi,'%f',nr+1);
  CX(j,:)=a';
end
%
st=fclose(fi);
%
surf(AX,BX,CX);
title('Dibujo de helice');
```