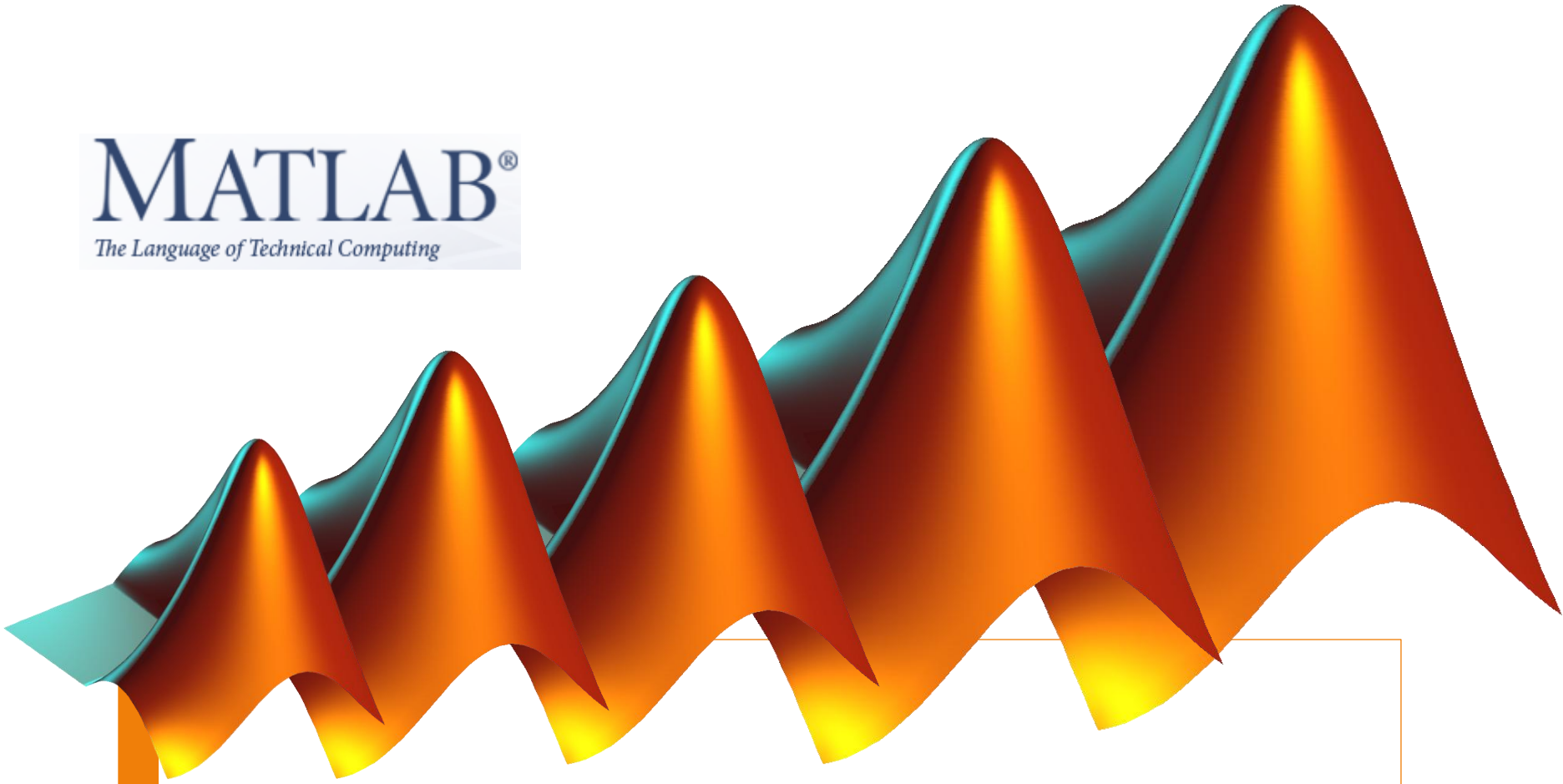


Introducción a Matlab

Ing. Laura López López

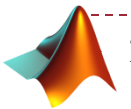
¿Qué es Matlab?

MatLab significa "MATrixLABoratory"

MATLAB es un lenguaje de alto nivel para realizar cálculos científico-técnicos.

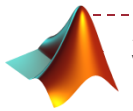
Integra las herramientas de cálculo necesarias con otras de visualización así como, un entorno de programación de fácil uso.

Proporciona unos paquetes de extensión ("toolboxes") para aplicaciones específicas

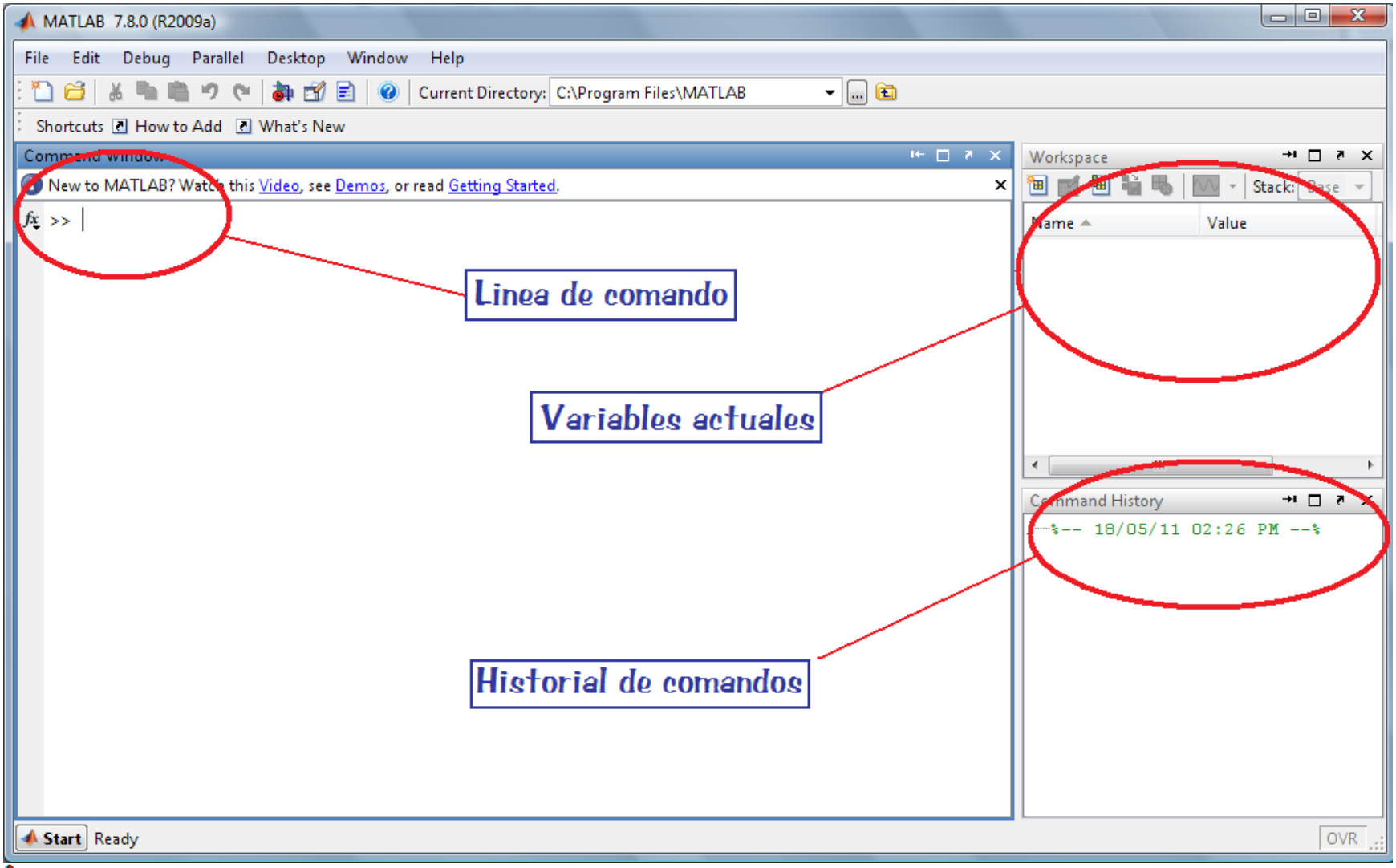


Aplicaciones típicas

- ✓Cálculo matemático
- ✓Desarrollo de algoritmos
- ✓Adquisición de datos
- ✓Modelado, simulación y prototipito
- ✓Análisis de datos y visualización
- ✓Gráficos
- ✓Desarrollo de aplicaciones e interfaces gráficas de usuario (GUI)



El entorno de Matlab



Matrices

Definir una variable:

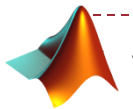
```
>> A = 7
A =
7
```

Definir una matriz:

```
>> B = [1 2 3; 4 5 6; 7 8 9]
B =
1 2 3
4 5 6
7 8 9
```

Para comprobar el valor de una variable se puede, bien mirar en la parte superior izquierda dedicada a las variables activas, bien introduciendo su nombre.

```
>> B
B =
1 2 3
4 5 6
7 8 9
```



Eliminar una variable de memoria:

```
>> clear B
>> clear all  %elimina todas las variables
```

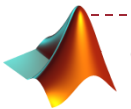
Para acceder a un elemento de una matriz: tomando la matriz B, queremos acceder al valor de la posición (1,2).

```
>> B(1,2)
ans =
2
```

Acceder a todos los elementos de una fila o columna:

```
>> B(:,1) % Para obtener todos los elementos de la columna 1.
ans =
1
4
7
```

```
>> B(1,:) % Para obtener todos los elementos de la fila 1.
ans =
1 2 3
```



Añadir elementos a una matriz:

```
>> B(4,1) = -1
B =
1 2 3
4 5 6
7 8 9
-1 0 0
```

Producto escalar:

```
>> A = [1 2;3 4]
A =
1 2
3 4
>> B = [4 3; 2 1]
B =
4 3
2 1
>> A(1,1)*B(1,1)
ans =
4
```

Producto matricial:

```
>> A*B

ans =
8 5
20 13
```

Ejemplo

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

A =

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

% Asignaremos un 0 en la posición 3,3

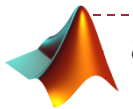
```
>> A(3,3)=0

A =

     1     2     3
     4     5     6
     7     8     0
```

% Cambiar a 77 el elemento de la tercera fila y columna

```
>> A(3,3)=77

A =

     1     2     3
     4     5     6
     7     8    77
```

% Cambiar a 200 el elemento de la primera fila y tercer columna

```
>> A(1,3)=200

A =

     1     2   200
     4     5     6
     7     8    77
```



% Incrementar el arreglo a 3x6 com ceros y poner um 1 em la posicion (2,6)

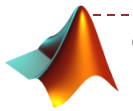
```
>> A(2,6)=1
```

```
A =

     1     2   200     0     0     0
     4     5     6     0     0     1
     7     8    77     0     0     0
```

% Crear una matriz B tomando filas de A en orden inverso

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

A =

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

```
>> B =A(3:-1:1,:)

B =

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



% Crear B sacando las primreas dos filas y las ultimas columnas de A

```
>> A= [10 20 30; 40 50 60; 70 80 90]

A =

    10    20    30
    40    50    60
    70    80    90
```

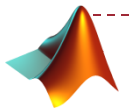
matriz base A

```
>> C= [1 3]

C =

     1     3
```

arreglo C

```
>> B= A(C,C)

B =

    10    30
    70    90
```

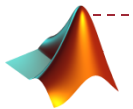
Crea B de A en base a C



% Calcular la transpuesta de uma matriz

```
>> B= [1 2 3 4 5 6 7 8 9]

B =

     1     2     3     4     5     6     7     8     9

>> B=B'

B =

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

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

A =

     1     2     3
     4     5     6
     7     8     9

>> A=A'

A =

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



% Elimina la segeunda fila de B

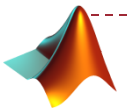
```
>> B=A

B =

     1     4     7
     2     5     8
     3     6     9

>> B(2,:)=[]

B =

     1     4     7
     3     6     9
```

% Crear datos desde -3 hasta 3

```
>> x=-3:3

x =

    -3    -2    -1     0     1     2     3
```

% Valor absoluto de x donde el mayor numero sea 1

```
>> abs(x)>1

ans =

     1     1     0     0     0     1     1
```



Funciones especiales.

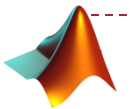
Matlab proporciona una serie de funciones matemáticas básicas además de funciones más complejas.

Como ejemplo de funciones aritméticas básicas tenemos:

- abs() % proporciona el valor absoluto de un numero.
- cos() % coseno.
- sin() % seno.
- sqrt() % cálculo de la raíz cuadrada.
- inv () % calcula la inversa de una matriz.

Y como ejemplo de otras funciones tenemos:

- clock %Muestra, en un vector de seis componentes, la fecha y hora completa.
- display (‘ ‘) %Muestra el texto introducido por pantalla.



En Matlab existen una serie de comandos que permiten crear matrices especiales por ejemplo:

% Crear una matriz de ceros de 3x3

```
>> zeros (3)

ans =

     0     0     0
     0     0     0
     0     0     0
```

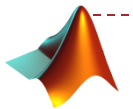
% Crear una matriz donde todos los elementos Del valor de PI

```
>> ones (3) *pi

ans =

    3.1416    3.1416    3.1416
    3.1416    3.1416    3.1416
    3.1416    3.1416    3.1416
```

% Crear una matriz de 2x4 de unos

```
>> zeros (3)

ans =

     0     0     0
     0     0     0
     0     0     0
```

```
>> ones (2,4)

ans =

     1     1     1     1
     1     1     1     1
```



% Crear una matriz de 3x1 com numeros aleatorios

```
>> rand (3,1)

ans =

    0.8147
    0.9058
    0.1270
```

% Crear una matriz identidad de 3x2

```
>> eye(3,2)

ans =

     1     0
     0     1
     0     0
```

% Crear una matriz identidad

```
>> eye(3)

ans =

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

% Para definir el tamaño de uma matriz se usa "size" por ejemplo se crea una matriz y em base a esa generar outra de números

```
>> A = [1 2 3 ; 4 5 6]

A =

     1     2     3
     4     5     6
```

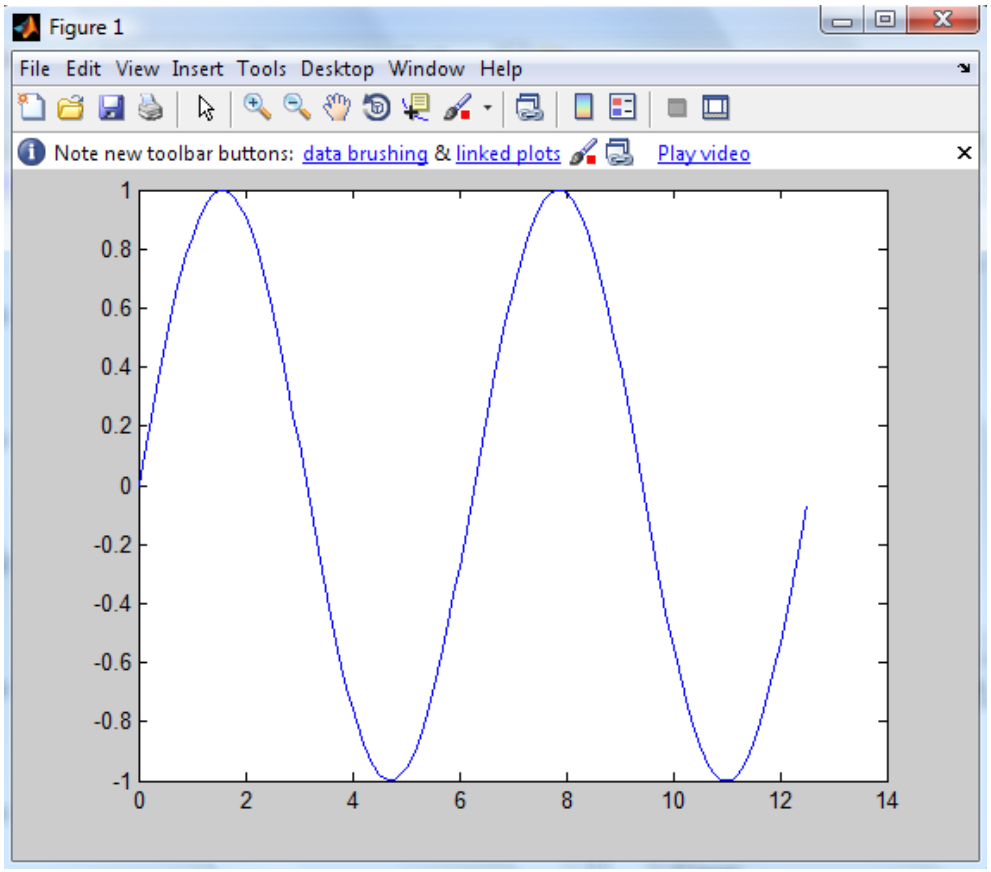
```
>> ones(size (A))

ans =

     1     1     1
     1     1     1
```

Representación gráfica.

Representación de una señal senoidal

Se trata de una señal analógica, puesto que existen infinitos valores entre dos puntos cualquiera del dominio.

```
a = 0:0.1:4*pi
plot(a,sin(a))
```



Sentencias de Control

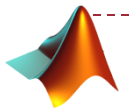
La sintaxis de las sentencias de control utilizadas dentro del entorno de Matlab es la siguiente:

FOR

```
for variable = valor_inicial:valor_final
sentencias
…
end
```

Ejemplo:

```
>> for i=1:3
display('hola mundo')
end

ans =
hola mundo
hola mundo
hola mundo
```

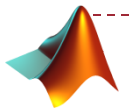


WHILE

while variable expresion
sentencias
...
end

Ejemplo:

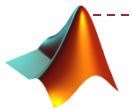
```
>> i = 1;
while i < 3
display('hola mundo')
i = i+1;
end
ans =

hola mundo
hola mundo
```



IF

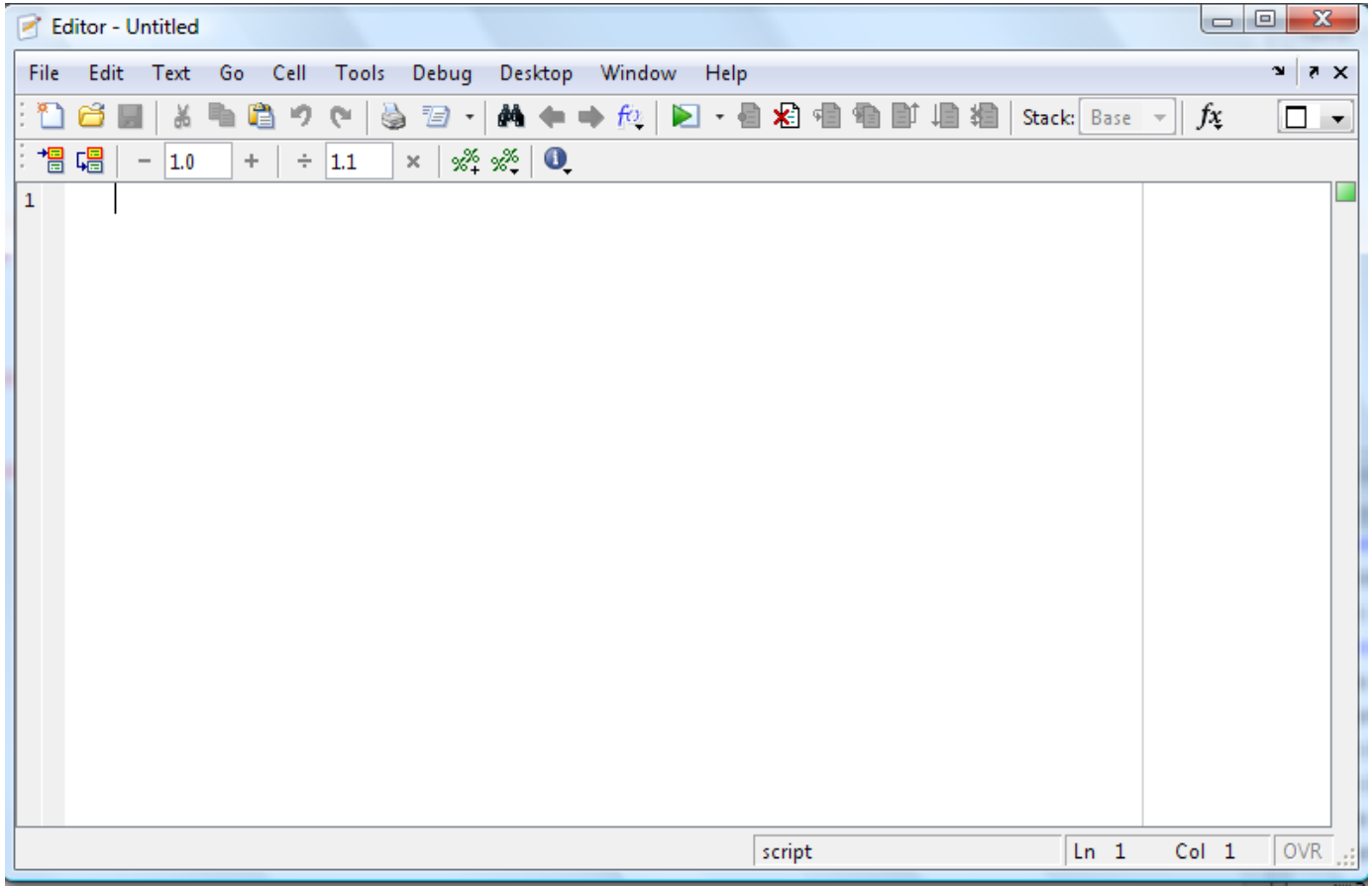
if expresion
sentencias
end

```
>> a=1

a =

     1

>> b=1

b =

     1

>> if (a==b)
display ('Hola Mundo')
end
Hola Mundo
```

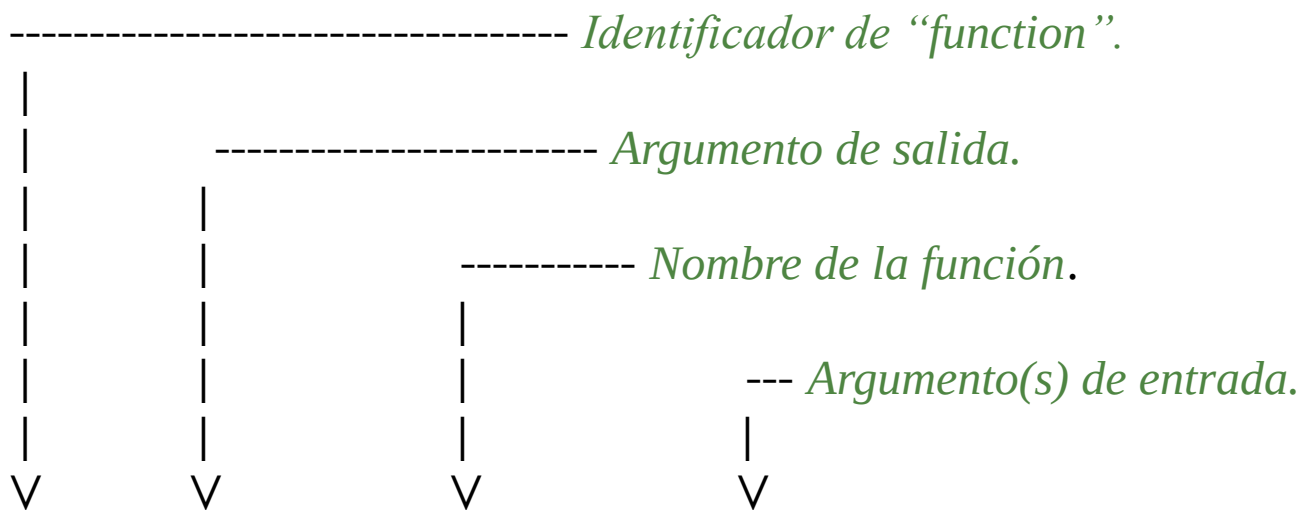


Creación de una función a través de un fichero .m

Con Matlab también es posible crear nuestras propias funciones. Para ello se puede utilizar bien el editor de texto de Matlab



La estructura a seguir a la hora de implementar una función es la siguiente:



function [o1,o2,...] = nombre_fun(i1,i2...) ← Definición

% Aquí se escribiría la ayuda que queremos que aparezca cuando

% el usuario escriba "help nombre_fun"

% ...

% ...

Cuerpo de la función (Aquí estaría la parte del código).

% Comentarios si los hubiera.

Ejemplo:

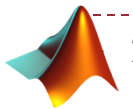
A continuación se muestra una función que calcula la inversa de una matriz.

```
function a= inversa (b)
% Función = Calcula la inversa de una matriz
% Parámetro de entrada = b;
% Parámetro de salida = a;
%
a = inv(b);
```

Si guardamos el texto anterior en un fichero .m (con el mismo nombre de la función, es decir, inversa.m) y
lo ejecutamos para la matriz f tenemos:

```
>> f=[1 2 ; 3 4]

f =

     1     2
     3     4
```

```
>> inversa (f)

ans =

   -2.0000    1.0000
    1.5000   -0.5000
```



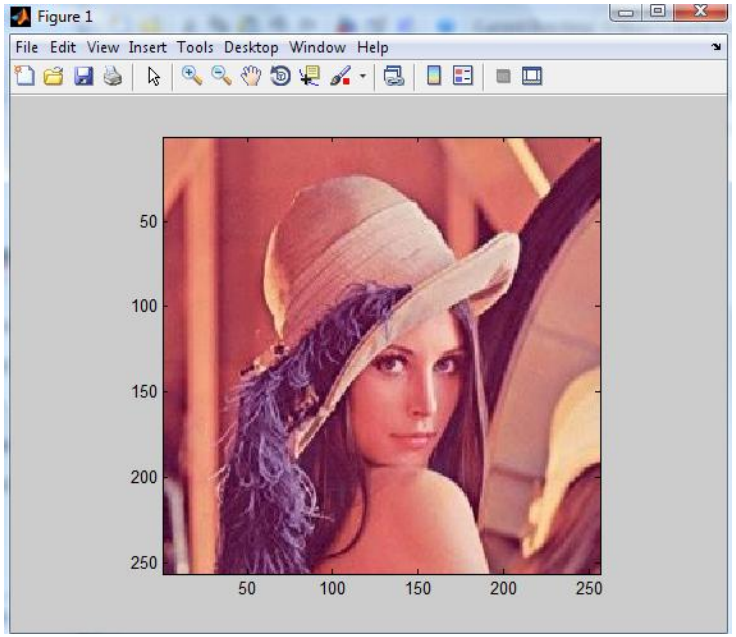
Toolbox de Image Processing

Leer y mostrar una imagen.

El comando utilizado por Matlab para leer una imagen es imread.

>> I=imread('*.tif');

Esta instrucción lee una imagen (en este caso en formato tiff) y la almacena en una matriz llamada I.



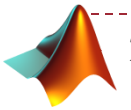

```
>> D=imread('lena.jpg');
>> subimage(D)
```

Histograma de una imagen

Con un histograma es posible ver la distribución de intensidades en una imagen basta con utilizar la función **imhist. Si nos interesa ecualizar el histograma de una imagen (para extender** los valores de intensidad) bastaría con utilizar la función **histeq.**

El código matlab para visualizar en una misma pantalla una imagen y su histograma podría ser:

```
clear all;
LN=imread('lena.jpg‘);
LN1=rgb2gray(LN); %pasar la imagen a escala de grises.
subplot(2,1,1),subimage(LN1),title('Imagen original');
subplot(2,1,2),imhist(LN1),title('Histograma');
```



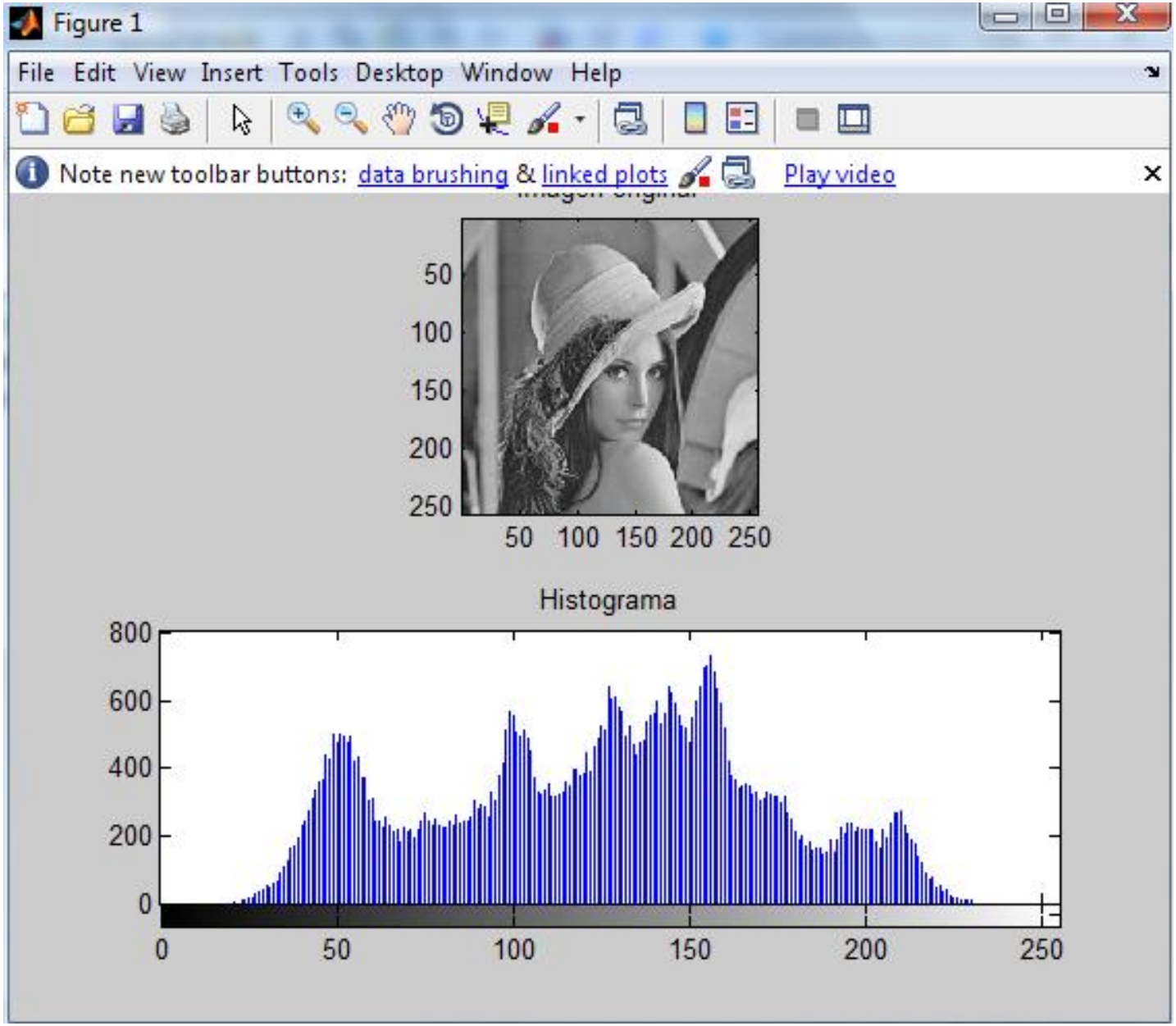


Figure 1
File Edit View Insert Tools Desktop Window Help
Note new toolbar buttons: data brushing & linked plots
Play video
50
100
150
200
250
50 100 150 200 250
Histograma
800
600
400
200
0
0 50 100 150 200 250

Transformada discreta de Fourier

```
close all

f = double(imread('cameraman.tif'));

figure(1);
imshow(f,[]);
figure(2);
mesh(f);
F=fft2(f);
```