

Guía básica de Matlab

Ésta es una pequeña guía de Matlab para poder realizar sin problemas la práctica. Lo primero de todo es explicar la ventana de Matlab que observamos nada más abrirlo y explicar cada una de sus partes. Como podemos ver en la Fig. 1, hay cuatro zonas diferenciadas:

1. **Carpeta actual:** en esta ventana podemos ver la carpeta en la que estamos y los archivos que tenemos, los tipos de archivos más comunes que podemos tener son funciones (.m), scripts (.m), archivos de datos (.mat) o gráficas (.fig). Cuando queramos usar una función que hayamos definido o ejecutar un script debemos estar en la misma carpeta que dicha función o script.
2. **Ventana de comandos:** en esta ventana podemos ejecutar en pantalla el cálculo que queramos, cualquier calculo que termine con “;” no se mostrará por pantalla, mientras que si termina con “,” o con nada, lo imprimirá por pantalla. Para ejecutar un script que tengamos en la carpeta actual simplemente tenemos que escribir su nombre en la ventana de comandos.
3. **Workspace:** en esta ventana podemos ver todas las variables que están definidas, ya sean vectores, números o cantidades simbólicas. Si escribimos en la ventana de comandos el nombre de cualquiera de ellas, nos la imprimirá en pantalla. Al cerrar Matlab, todas las variables en el Workspace se borrarán, si queremos guardar datos tenemos que usar la función *save*.
4. **Barra de herramientas:** como en muchos programas, en esta barra podemos buscar distintas herramientas u opciones.

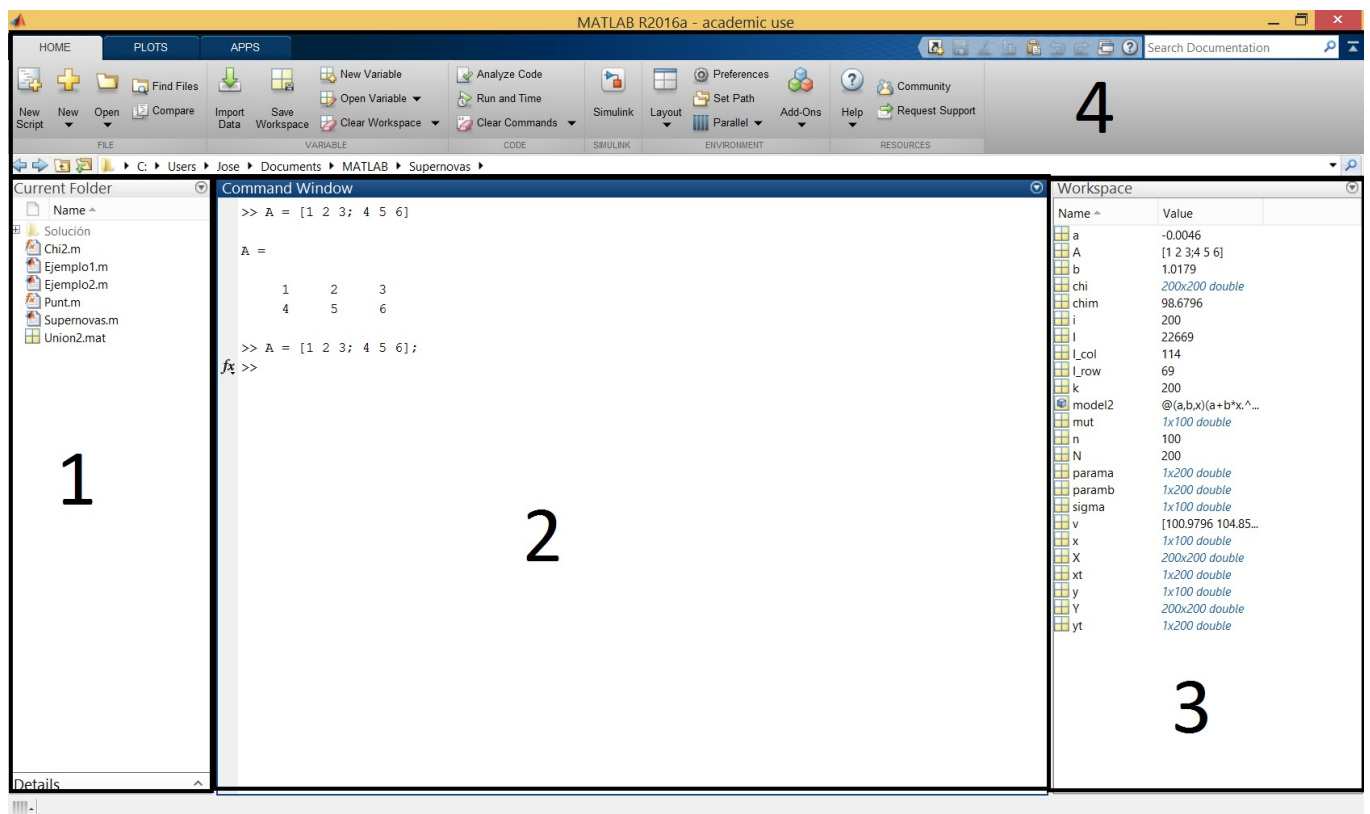


Figura 1: Pantalla inicial de Matlab.

A. Funciones y scripts

Las dos herramientas básicas para trabajar en Matlab son las funciones y los scripts. Para crear tanto una función como un script vamos a la ventana carpeta actual, hacemos click derecho, New File y allí pulsamos Script o Function. Esto nos creará un archivo que podemos ver en esa misma ventana, dándole doble click podemos abrirlo.

a. Funciones: las funciones tienen unos argumentos de entrada y unos de salida, la estructura es la que se puede ver en la Fig. 2. El nombre de la función tiene que coincidir con el nombre del archivo. Para llamar a una función (ya sea en un script como en la ventana de comandos) solo tenemos que poner las variables de salida entre corchetes, el nombre de la función y sus variables de entrada. Por último, todas las variables internas de la función que no sean variables de entrada ni de salida, no quedarán definidas en el workspace (en el ejemplo de la Fig. 2 la variable h no quedará definida en el workspace después de ejecutar la función).

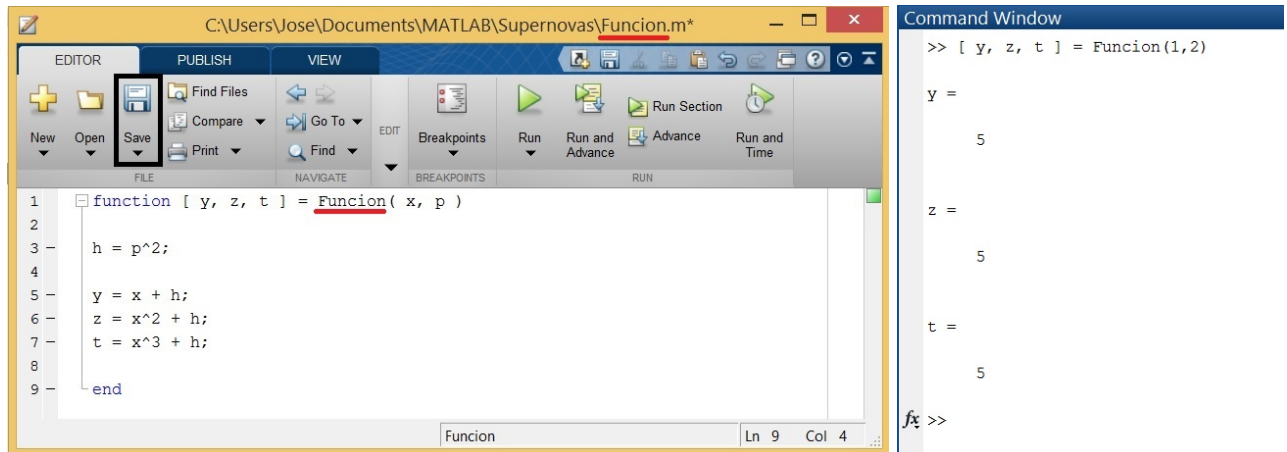


Figura 2: Ejemplo de función. Cuando el icono que está remarcado (save) tiene el color azul de la imagen, significa que la función o el script no está guardado, antes de usar la función debemos guardar los cambios.

b. Scripts: los scripts son archivos donde podemos escribir distintas operaciones como si las fuéramos a ejecutar en la ventana de comandos. Por ello, y a diferencia de las funciones, todas las variables que definamos se quedarán en el workspace. Como hemos explicado antes, los scripts se ejecutan escribiendo su nombre (y dando a enter) en la ventana de comandos. Por último, todas las líneas de comando (tanto en scripts como en funciones) que vengan precedidas por $\%$ se consideran comentarios.

c. Funciones definidas en línea: por último, es posible definir funciones en la ventana de comandos o en un script. Para ello, definimos la función de la siguiente forma:

$$f = @(x, y, z, \dots)(x^2 + y^4 + z^3 \dots), \quad (1)$$

donde la función en este ejemplo sería " $x^2 + y^4 + z^3 \dots$ ". Las variables de entrada pueden ser en general vectores y matrices. Para llamar a esta función simplemente escribimos:

$$Sol = f(x, y, z, \dots). \quad (2)$$

B. Vectores y matrices

En Matlab existen distintos tipos de variables con los que se puede trabajar, unas de las más potentes son las matrices y, en particular, los vectores. Las matrices son agrupaciones de números, en general reales (o complejos), con las que se pueden realizar operaciones. Pueden tener la dimensión que queramos: una dimensión (vector

fila/columna), dos dimensiones (matrices con filas y columnas), tres dimensiones (cubos), etc. En esta práctica trabajaremos principalmente con vectores fila y alguna matriz.

a. *Vectores*: un vector fila se define con una lista de números espaciados y entre corchetes ($v = [1\ 2\ 3]$), mientras que un vector columna se define espaciando con “;” los valores ($v = [1; 2; 3]$). Además, podemos extraer elementos de un vector escribiendo el nombre del vector y entre paréntesis el elemento que queremos ($v(3)$). Si lo que queremos es extraer del vector un conjunto de valores, tenemos que escribir el nombre del vector y entre paréntesis el elemento inicial y final separados por “:” ($v(1:2)$). Como podemos ver, el primer elemento del vector está etiquetado 1 (en python es 0). Podemos realizar operaciones con vectores elemento a elemento: suma (+), producto (*), cociente (/) o elevar a un número (.^). El punto antes de la operación es para indicarle a Matlab que haga la operación elemento a elemento (en el caso de la suma no hay posible confusión). Es necesario que los vectores con los que se opere tengan el mismo número de elementos y sean los dos vectores fila o columna. En el tutorial de la práctica se trabaja todo con vectores fila.

Command Window	Command Window	Command Window
<pre>>> v = [1 2 3] v = 1 2 3 >> v = [1; 2; 3] v = 1 2 3</pre>	<pre>>> v(3) ans = 3 >> v(1:2) ans = 1 2</pre>	<pre>>> v1 = [1 2 3]; >> v2 = [2 4 6]; >> v1.*v2 ans = 2 8 18</pre>

Figura 3: Ejemplo de vector fila y vector columna, ejemplo de cómo extraer elementos de un vector y, por último, ejemplo de operación con vectores elemento a elemento.

b. *Matrices*: una vez explicados los vectores, las matrices son una extensión natural. Para construirlas basta con escribir un vector fila y a continuación otro separado por “;”, todo ello entre corchetes ($A = [1\ 2\ 3; 4\ 5\ 6]$). Para extraer valores tenemos que especificar la fila y la columna, el primer elemento es la fila y el segundo la columna ($A(fila, columna)$). Las operaciones con matrices elemento a elemento se realizan de la misma forma que con los vectores.

Command Window	Command Window
<pre>>> A = [1 2 3; 4 5 6] A = 1 2 3 4 5 6</pre>	<pre>>> A(2,3) ans = 6</pre>

Figura 4: Ejemplo de matriz y cómo extraer elementos de ella.

C. Bucles

En esta práctica sólo es necesario el uso del bucle for. Éste recorre un índice desde el valor inicial hasta el final que queramos, su escritura la podemos ver en el siguiente ejemplo. En él se parte de un vector v como el definido anteriormente y se suman todos sus elementos, almacenando este dato en la variable z .

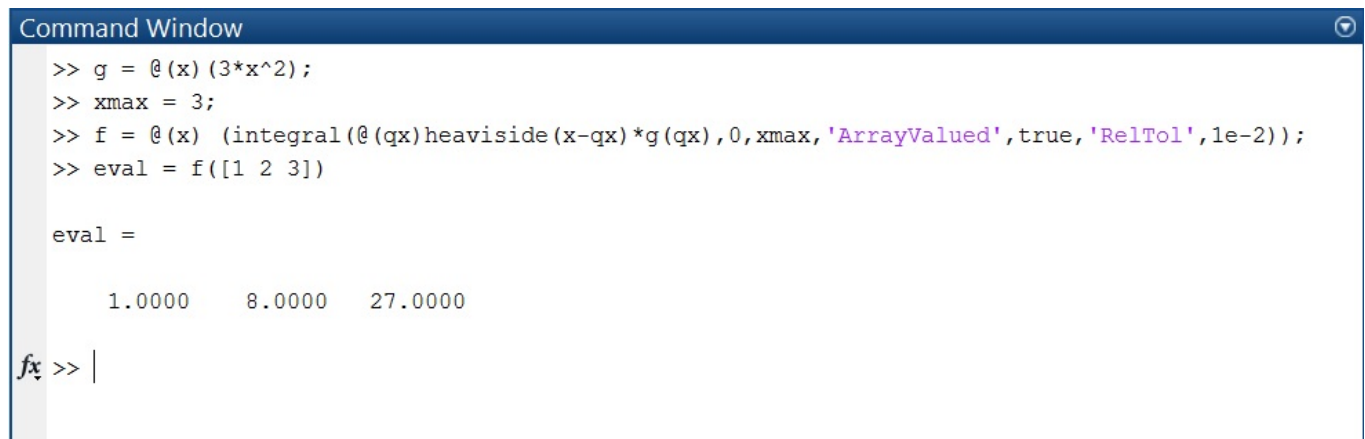
```
v = [1 2 3];
z = 0;
for i=1:3
    z = z + v[i];
end
```

D. Integrales definidas

En esta sección presentamos una forma sencilla de realizar una integral del siguiente tipo,

$$f(x) = \int_0^x g(x') dx'. \quad (3)$$

En la siguiente figura vemos la resolución de esta integral usando funciones definidas en línea, como ejemplo consideramos $g(x') = 3x'^2$. Para evitar errores numéricos debemos elegir una tolerancia suficientemente baja, en éste caso es suficiente con 0.01.



```
Command Window
>> g = @(x) (3*x^2);
>> xmax = 3;
>> f = @(x) (integral(@(qx) heaviside(x-qx)*g(qx),0,xmax,'ArrayValued',true,'RelTol',1e-2));
>> eval = f([1 2 3])

eval =

    1.0000    8.0000   27.0000

fx >> |
```

Figura 5: Ejemplo de integral definida desde cero hasta un cierto valor de x .

I. FUNCIONES ÚTILES MATLAB

Las siguientes funciones y comandos están definidos internamente en Matlab y son los únicos necesarios para realizar la práctica. Para más información sobre alguno de ellos basta con escribir su nombre en Matlab, seleccionar su nombre, dar click derecho y pulsar en help on selection.

Función	Entrada	Salida	Descripción
$y = \text{sqrt}(x)$	x	y	Realiza la raíz cuadrada de x. Si x es un vector, la función devuelve un vector del mismo tamaño con la raíz cuadrada de los elementos del vector x.
$y = \text{log10}(x)$	x	y	Realiza el logaritmo decimal de x. Si x es un vector, la función devuelve un vector del mismo tamaño con el logaritmo decimal de los elementos del vector x.
$y = \text{abs}(x)$	x	y	Realiza el valor absoluto de x. Si x es un vector, la función devuelve un vector del mismo tamaño con el valor absoluto de los elementos del vector x.
$y = \text{sum}(v)$	v	y	Suma los elementos del vector v.
$[M, I] = \text{min}(v)$	v	M, I	Obtiene el valor mínimo (M) del vector v, así como su posición (I) en el vector v. De modo que $v(I) = M$.
$y = \text{length}(v)$	v	y	Devuelve el tamaño del vector v, de modo que el último valor de v es $v(\text{length}(v))$.
$v = \text{zeros}(i, j)$	i, j	v	Dados el número de filas (i) y el número de columnas (j), la función nos devuelve una matriz de ceros con ese número de filas y columnas. Para crear un vector fila de n ceros simplemente ponemos $i = 1$ y $j = n$.
$v = \text{linspace}(x1, x2, n)$	x1, x2, n	v	La función nos devuelve un vector (v) que va de x1 a x2 equiespaciado en n elementos. De modo que $\text{length}(v) = n$.
$I = \text{trapz}(x, y)$	x, y	I	Dados un vector x y un vector y, la función nos calcula la integral por medio del método trapezoidal de la función y(x).
$\text{load}('Archivo.mat')$	Archivo.mat	datos	La función nos carga en el workspace las variables almacenadas en el archivo Archivo.mat, siempre que dicho archivo este en la carpeta actual.
$[X, Y] = \text{meshgrid}(xv, yv)$	xv, yv	X, Y	Dados un vector xv y otro yv, la función nos construye dos matrices X, Y. La matriz X tiene tantas filas como elementos tiene yv, y cada fila de X es un vector xv. La matriz Y tiene tantas columnas como elementos tiene xv, y cada columna de Y es un vector yv. Esta función se usa para realizar los contours.
$\text{plot}(x, y)$	x, y	Gráfico	Dados los vectores x e y, la función nos representa sus valores (y frente a x) unidos por líneas rectas.
$\text{errorbar}(x, y, dy)$	x, y, dy	Gráfico	Dados los vectores x e y, la función nos representa sus valores (y frente a x) unidos por líneas rectas, junto con las barras de error de y (dy).
$\text{contour}(X, Y, Z, v)$	X, Y, Z, v	Gráfico	Dada una matriz $Z = Z(x, y)$ y las matrices X, Y obtenidas por la función meshgrid a partir de los vectores xv e yv; la función nos representa los contornos de $Z = v$, siendo v un vector.
hold on	-	-	Comando que nos permite seguir dibujando en una misma gráfica.
$\text{xlabel}('varx')$	-	-	Comando que nos permite poner nombre (varx) al eje x de un gráfico.
$\text{ylabel}('vary')$	-	-	Comando que nos permite poner nombre (vary) al eje y de un gráfico.
$\text{legend}('datos1', 'datos2', \dots)$	-	-	Comando que nos permite poner leyenda a un gráfico, los primeros datos representados quedarán etiquetados por datos1, los segundos con datos2, etc.