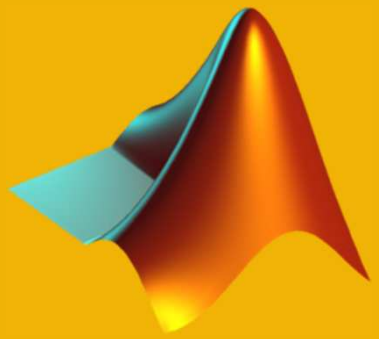


MATLAB

Introducción al uso del lenguaje

UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

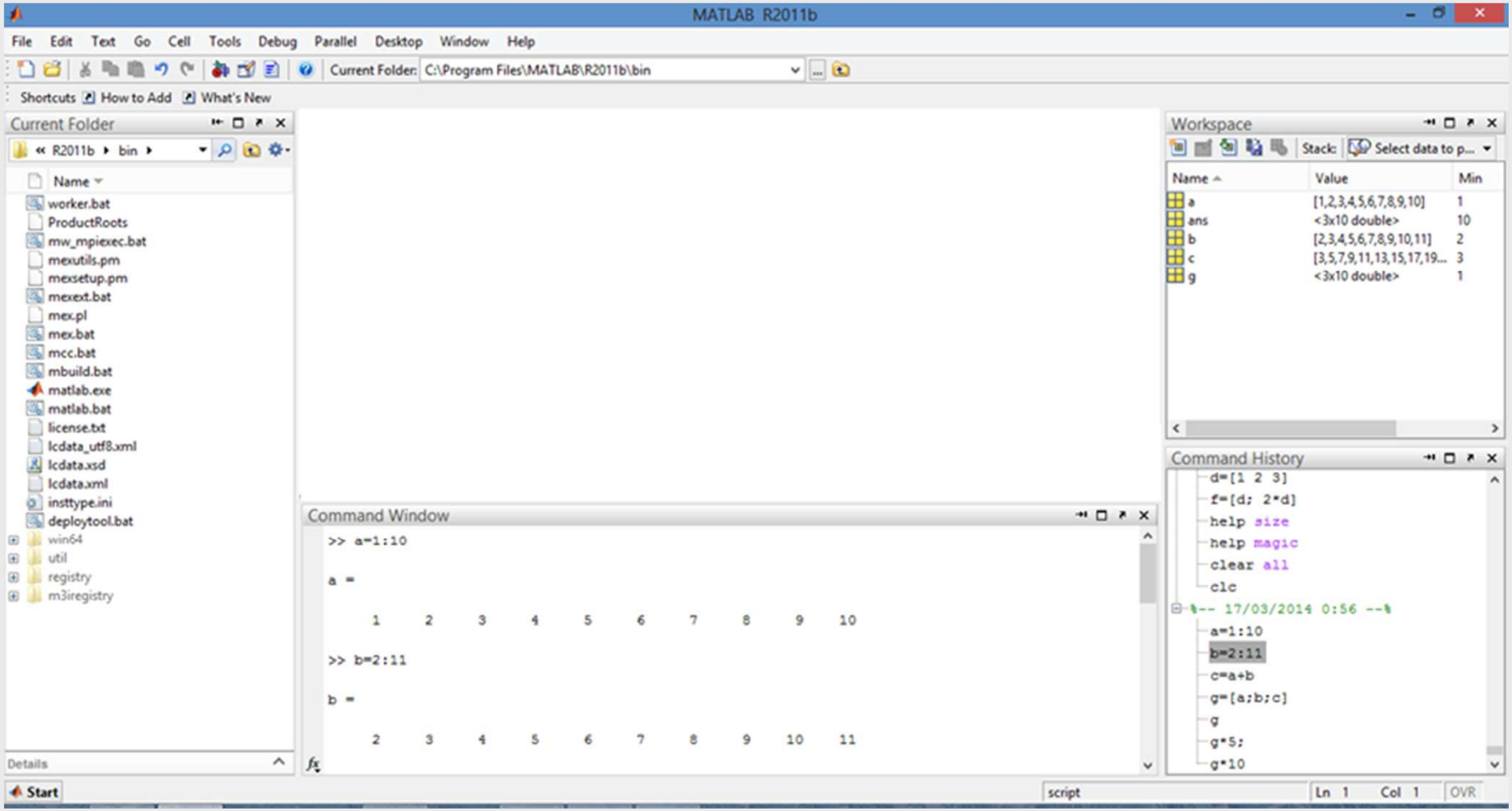
Introducción

- Matlab es un programa interactivo útil en computación numérica y visualización de datos; los ingenieros en control lo usan extensivamente en análisis y diseño.
- Están disponibles muchas "toolboxes" diferentes, las cuales extienden aún más las funciones básicas del Matlab a diferentes áreas.

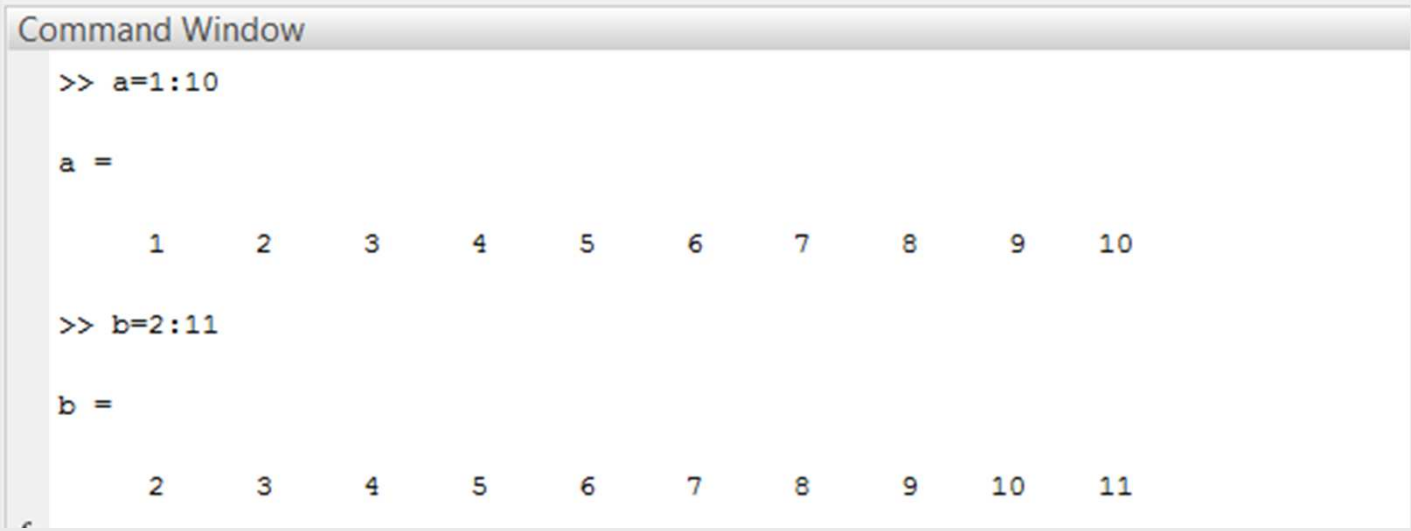
Consola de comandos

- A diferencia de los entornos de programación usados hasta el momento, MatLab no sólo permite ejecutar rutinas de código prearmadas, sino también ejecutar desde su consola secuencias de comandos, o comandos individuales, de manera separada y conservando en memoria resultados y variables.

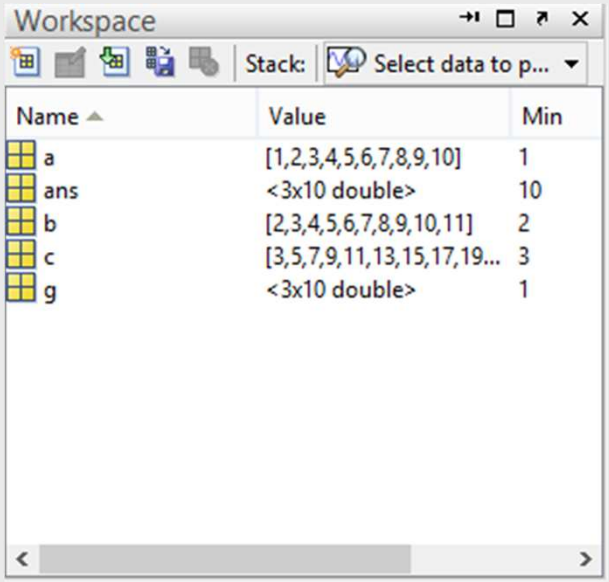
Pantalla principal

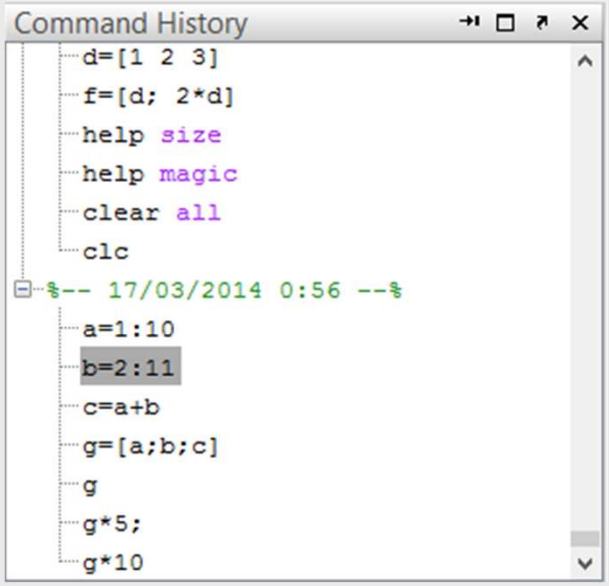


Command Window



Ventana para definir variables, llamar funciones y realizar pruebas de unas pocas líneas de código.





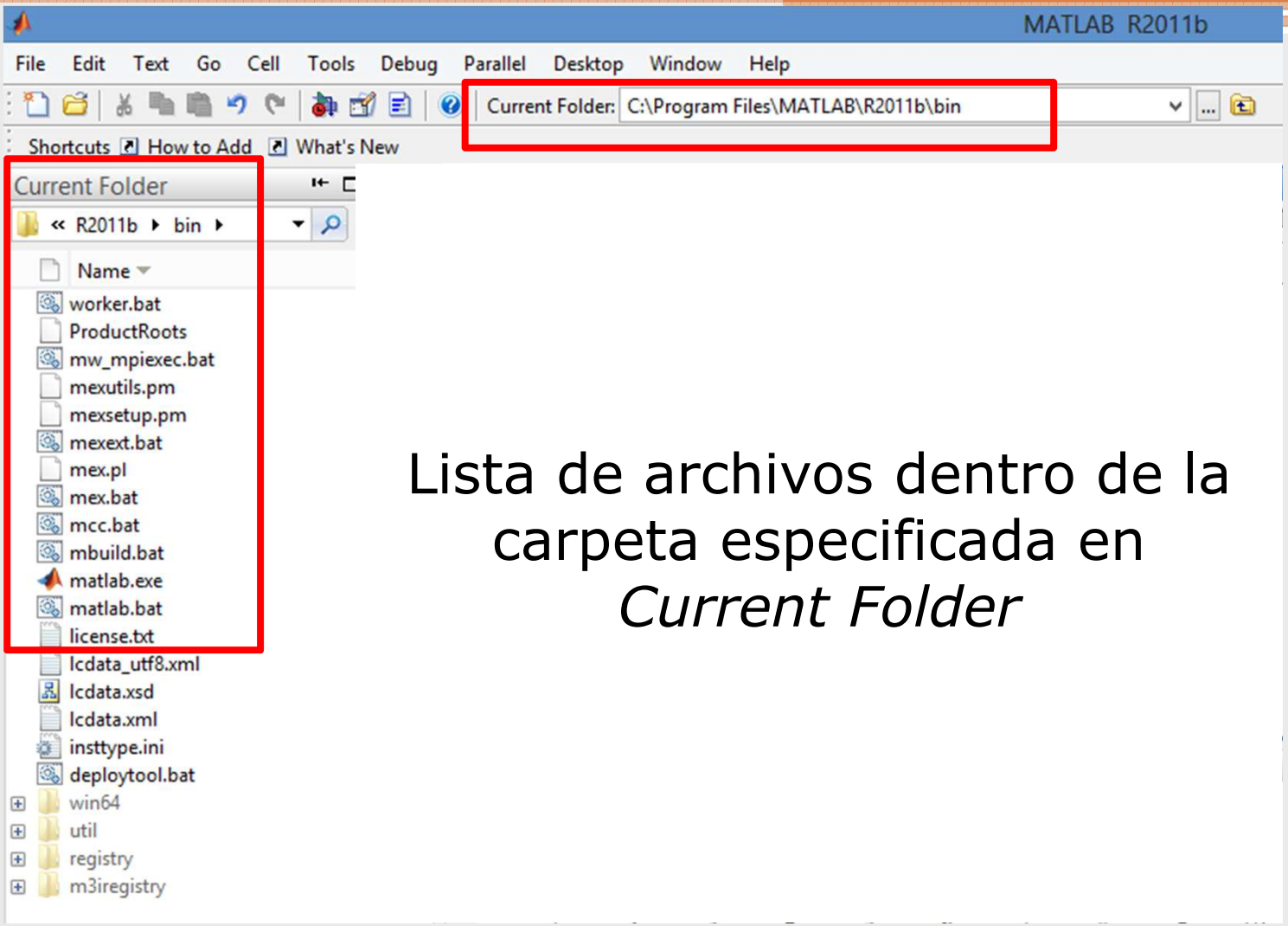
Workspace:

Listado de variables actuales con sus caracteristicas (dimensión,tipo,valor max/min,etc).

(si esta oculta puede agregarse en Desktop->Workspace)

Command History:

Historial del código ejecutado en la ventana de comandos



Lista de archivos dentro de la carpeta especificada en *Current Folder*

Variables

- Entero: *a=4*
- Decimal: *b=4.321*
- Array de chars: *c='sinergia'*
- Vectores: *d=[4 3.4 -4]*
  e=[4;3.4;-4]
- Matrices: *f=[7 -3 2; 3 4.1 5]*
  *g=[d; -2*d]*

Variables

- No es necesario especificar su tipo ni dimensión.

- Se redimensionan automáticamente

  X=[4 7 2] crea un vector *X*

  X=5 sobrescribe la variable X y la convierte en un escalar

Manipular vectores

- Creación de un vector:

  a = [1 2 3 4 5 6 7 8 9]

  a = 1 2 3 4 5 6 7 8 9

- Creación de un vector con elementos de 0 a 20 a incrementos de 2 en forma automática:

  t = 0:2:20

  t = 0 2 4 6 8 10 12 14 16 18 20

(Si se omite el parámetro central, el incremento por defecto es 1. Ejemplo: h=1:5 dará h=1 2 3 4 5)

- Suma de un vector con un **escalar**:

  b = a + 2

  b = 3 4 5 6 7 8 9 10 11

- Suma de dos vectores de **igual longitud**

  c = a + b

  c = 4 6 8 10 12 14 16 18 20

Manipular vectores

- Referenciar un elemento de un vector:
  a(posición)

Es importante recordar dos diferencias con respecto a C:

- *Para indicar el primer elemento del vector se utiliza la posición 1, no la 0.*
- *La posición se indica entre () y no entre [].*

- Ejemplo:
  a(15) = 10

  a = 1 2 3 4 5 6 7 8 9 0 0 0 0 0 10

  (Si se agrega un valor en una posición que no existía, automáticamente se completan las posiciones faltantes con ceros)

Manipular vectores

- Referenciar un subvector: *t(pos_inic:pos_fin)*

  t(1:4) = [1 2 3 4]

  t = 1 2 3 4 8 10 12 14 16 18 20

- Referenciar una fila o columna de una matriz

  m = [a;b]

  m= 1 2 3 4 5 6 7 8 9

  3 4 5 6 7 8 9 10 11

  c = m(:,1)

  c = 1

  3

Operaciones vectoriales

- Traspuesta:

  w=1:3

  w=1 2 3

  w'

  w= 1
  2
  3

- Multiplicación de vectores (cuidado dimensiones!):

  a=1:3; b=a;
  *c=a*b'*

  $$c=\begin{bmatrix}1 & 2 & 3\end{bmatrix}*\begin{bmatrix}1\\2\\3\end{bmatrix}= 1*1+2*2+3*3= 1+4+9=14$$

  c= 14

Operaciones vectoriales

- Multiplicación de vectores (cuidado dimensiones!):

  *d=a'*b*

  $$c=\begin{bmatrix}1\\2\\3\end{bmatrix} * \begin{bmatrix}1 & 2 & 3\end{bmatrix} = \begin{bmatrix}1 & 2 & 3\\ 2 & 4 & 6\\ 3 & 6 & 9\end{bmatrix}$$

  c= 1 2 3
  2 4 6
  3 6 9

- Multiplicación elemento a elemento

  *e=a.*b*

  *e=[a(1)*b(1) a(2)*b(2) a(3)*b(3)] =[1 4 9]*

  e = 1 4 9

Operaciones vectoriales

- Una sumatoria puede escribirse como una multiplicación de vectores:

$$\sum_{i=1}^{10} a_i . x_i = a_1 . x_1 + a_2 . x_2 + \cdots + a_{10} . x_{10}$$

$$\text{Si} \quad A = [a_1 \quad \ldots \quad a_N] \quad \text{y} \quad X = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}$$

$$A * X = a_1 . x_1 + a_2 . x_2 + \cdots + a_{10} . x_{10}$$

Funciones

- Matlab contiene todas las funciones estándares como **sin, cos, log, exp, sqrt**, así como tantas otras.
- También incorpora las constantes comúnmente usadas como ***pi***, e ***i*** o ***j*** para la raíz cuadrada de -1.
  - Ejemplo:

    sin(pi/4)

    ans = 0.7071
  - Cuidado al usar *i* como variable de iteración cuando se utilizan números imaginarios.
  - *Recomendación: Acostumbrarse a utilizar otra letra.*

Ayuda

- Para determinar la sintaxis y el funcionamiento de una función determinada utilice:

- *help nombredelafuncion (resumen)*
- *doc nombredelafuncion (más detallado)*

Ayuda (*help* vs *doc)*

```
>> help sin
 sin    Sine of argument in radians.
    sin(X) is the sine of the elements of X.

    See also asin, sind.
```

sin

Sine of argument in radians

Syntax

```
Y = sin(X)
```

Description

`Y = sin(X)` returns the circular sine of the elements of `X`. The `sin` function operates element-wise on arrays. The function's domains and ranges include complex values. All angles are in radians.

Definitions

The sine of an angle is:

$$\sin(x) = \frac{e^{ix} - e^{-ix}}{2i}$$

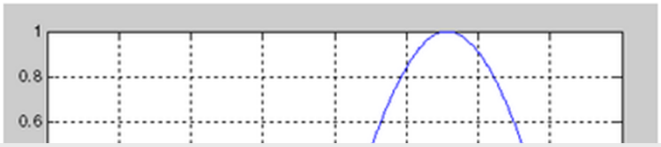
For complex x:

$$\sin(x+iy) = \sin(x)\cosh(y) + i\cos(x)\sinh(y)$$

Examples

Graph the sine function over the domain $-\pi \le x \le \pi$.

```
x = -pi:0.01:pi;
plot(x,sin(x)), grid on
```



Archivos .m

- *.m* es una extensión utilizada en archivos de Matlab.
- En particular utilizaremos dos tipos:
  - Scripts: un conjunto de líneas de código.
  - Funciones: son similares a las subrutinas de los lenguajes de programación, las cuales tienen entradas (parámetros que se pasan al archivo.m), salidas (valores que el archivo.m devuelve), y un cuerpo de comandos que puede contener variables locales
- Cuando ingresa un comando como *sqrt*, *plot*, o *top* en Matlab, lo que realmente está haciendo es correr un "archivo.m" con argumentos de entrada y salidas que se escribieron para realizar una tarea específica.

Funciones personalizadas

- Los archivos .m que contienen funciones utilizan la siguiente sintaxis:

  function [salida1]=nombre(entrada1, entrada2)

- Una función puede tener tantas entradas y salidas como se necesiten.

- Guardar el archivo con el mismo nombre de la función que contiene: *"nombre.m"*

Ejemplo: Función add

- Abrimos el editor:
  File->New->Function

- Escribimos nuestra rutina:

```
%add es una función que suma dos números
function [suma] = add(var1,var2)
    suma = var1+var2;
```

- Guardamos estas líneas en un archivo llamado "add.m" en el directorio donde estemos trabajando con Matlab

- Ahora podemos utilizarla desde la consola:

```
y = add(3,8)
    y = 11
```

Funciones personalizadas

- Por encima de la expresión:

  function [suma] = add(var1,var2)

  Puede agregarse el texto de ayuda que aparecerá cuando se utilice el comando *help* para esta función.

  Estas líneas son opcionales, pero deben ingresarse usando '*%*' al principio de cada línea.

  help add

  add es una función que suma dos números

Impresión

- Para imprimir en pantalla el contenido de una variable alcanza con escribir su nombre:

  nombredeunavariable

- Para imprimir un mensaje de texto podemos utilizar el comando:

  disp('mensajedetexto')

  mensajedetexto

- También podemos hacerlo de manera más compleja utilizando un comando similar al print de C:

  a='Dicaprio'; m=0;
  sprintf('%s ganó %d premios Oscar.\n', a, m)

  Dicaprio ganó 0 premios Oscar.

Varios

- La ejecución de un comando puede interrumpirse usando la combinación de teclas ***CTRL+C***.
- La variable ***ans*** contiene el resultado de la ultima operación que no fue salvada en una variable.
- El carácter '***%***' se utiliza para comentar código (equivalente a '//' en C).
- El comando ***clc*** borra la ventana de comandos (pero conserva las variables).
- El comando ***clear all*** borra todas las variables mientras que ***clear var1*** borra solo la variable *var1*.

Varios

- El carácter '***;***' es habitualmente usado para evitar que un comando produzca una salida por pantalla.
- Para ejecutar múltiples líneas en la consola de comandos se utiliza el carácter '***,***' (también reemplazable por '***;***') para separar las mismas.
- El comando ***what*** lista los archivos .m dentro de un directorio.
- El comando ***who*** muestra las variables en memoria y ***whos*** detalla sus características.