VNIVERSITAT DE VALÈNCIA

MASTER DE INGENIERÍA BIOMÉDICA.

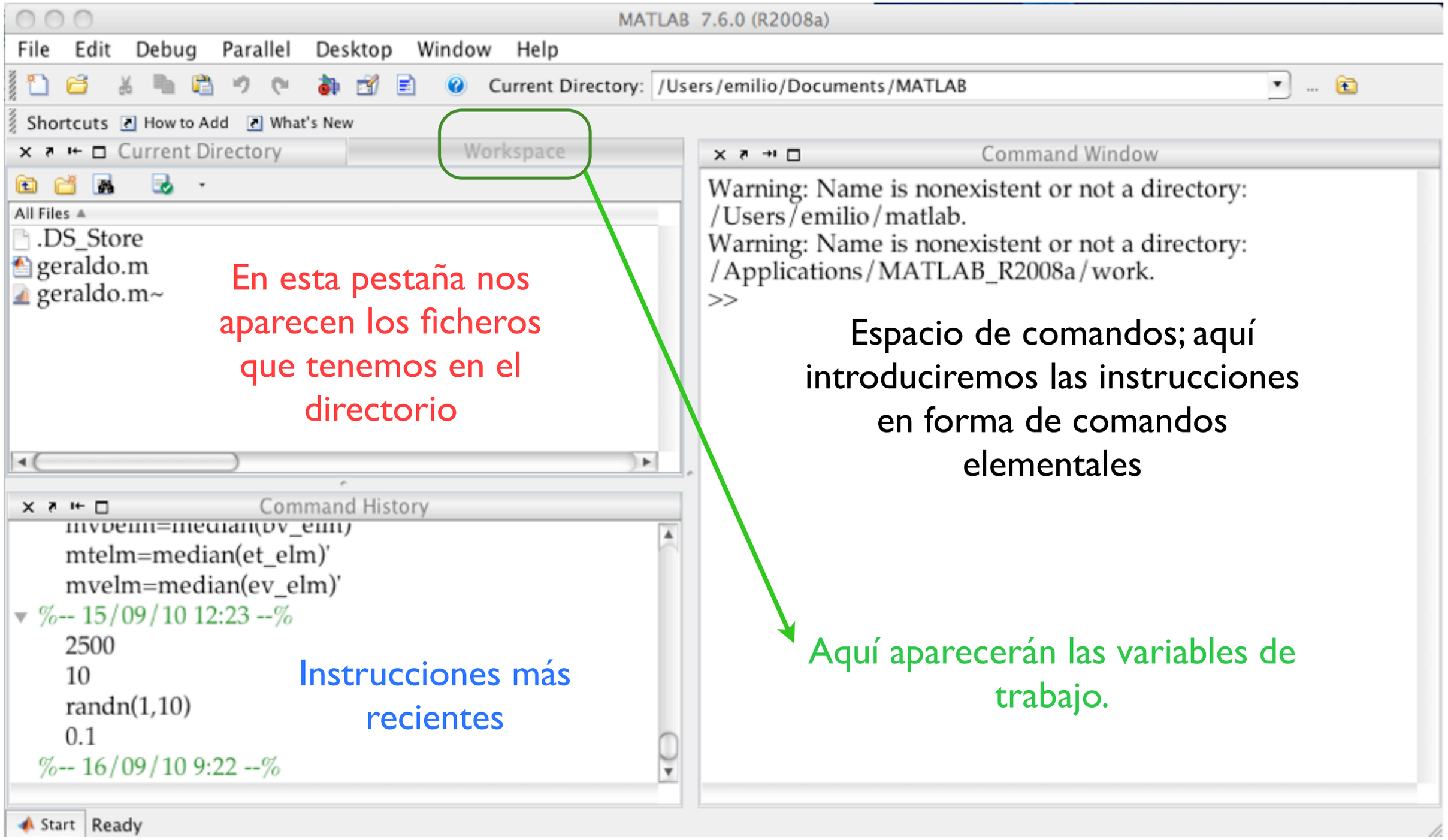
Guía para la realización de los trabajos (I)

Introducción al Matlab

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Arrancamos Matlab.....



Dependiendo de la versión aparecerán estos elementos de esta forma u otra.

Para que aparezcan todos los elementos seguir la secuencia de menús: Desktop--Desktop Layout--Default

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Algunas cuestiones.....

Atajos usando el teclado

↑	ctrl-p	Recall previous line
↓	ctrl-n	Recall next line
←	ctrl-b	Move back one character
→	ctrl-f	Move forward one character
ctrl-→	ctrl-r	Move right one word
ctrl-←	ctrl-l	Move left one word
home	ctrl-a	Move to beginning of line
end	ctrl-e	Move to end of line
esc	ctrl-u	Clear line
del	ctrl-d	Delete character at cursor
backspace	ctrl-h	Delete character before cursor
	ctrl-k	Delete (kill) to end of line

Constantes y valores especiales

`ans`	Most recent answer
`eps`	Floating point relative accuracy
`i or j`	$\sqrt{-1}$
`Inf`	Infinity
`NaN`	Not-a-Number
`nargin, nargout`	Number of actual function arguments
`pi`	3.14159 26535 897 ...
`realmax`	Largest positive floating point number
`realmin`	Smallest positive floating point number
`varargin, varargout`	Pass or return variable numbers of arguments

INSTRUCCIÓN BÁSICA: help (nombre de la función)

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Primeras pruebas.....

En la pantalla de comandos escribimos...

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

y debemos ver:

```
A =

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

¿qué ocurre si repites la instrucción con ; al final de la instrucción??

Mira el espacio de trabajo (workspace)

Prueba ahora `>> B=A'`

¿Qué hace el operador '?

¿Serías capaz de introducir las siguientes matrices?

```
X =

    16     3     2    13     0
     5    10    11     8     0
     9     6     7    12     0
     4    15    14     1    17
```

```
B =

    16     3     2
     5    10    11
     9     6     7
     4    15    14
    64    51    50
    53    58    59
    57    54    55
    52    63    62
```

Mira el espacio de trabajo y comprueba lo que tienes.

En Matlab cada objeto es una MATRIZ; un escalar es una matriz de 1x1; un vector fila es una matriz de 1xN y un vector columna es una matriz de Nx1; visto los ejemplos anteriores (que son lo más generales posibles) introduce un escalar, un vector fila y un vector columna.

Para obtener elementos dentro de una matriz la filosofía es siempre la misma; primero se indican las filas y luego las columnas:

```
A =

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

A(2,3)

```
A =

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

A([1 2],[1 2])

```
A =

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

A([1 2 3],[2 3])

y ahora tú!!!

```
A =

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

```
A =

     1     2     3
```

```
A =

     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
```

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

El operador :

Existe un operador básico en Matlab; este operador es : (dos puntos); tiene dos funciones:

a) Localizar elementos dentro de una matriz (lo visto anteriormente puede resultar bastante pesado)

b) Generar vectores.

LOCALIZAR ELEMENTOS

El uso es siempre el mismo

VALOR INICIAL:INCREMENTO:VALOR FINAL;

si el incremento es 1 se puede no indicar; si se quieren todas las filas o todas las columnas se indica directamente con :

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

A([1 2],[1 2])

A(1:1:2,1:1:2)

A(1:2,1:2)

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

A([1 2 3],[2 3])

A(1:1:3,2:1:3)

A(1:3,2:3)

A(:,2:3)

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

A([2 3],[1 2])

A(2:1:3,1:1:2)

A(2:3,1:2)

Genera en Matlab la siguiente matriz y localiza, usando el operador: las matrices que se marcan

$$B = \begin{pmatrix} 5 & 2 & -8 \\ 3 & 6 & -10 \\ 3 & 3 & -7 \end{pmatrix} \qquad B = \begin{pmatrix} 5 & 2 & -8 \\ 3 & 6 & -10 \\ 3 & 3 & -7 \end{pmatrix}$$

El otro uso que se le da (bastante extendido en el procesado digital de señales) es el de generar vectores:

Ejemplos:

Vector de 1 a 100: >>v=1:1:100

Pares de 0 a 50 >>p=0:2:50

Vector temporal de 0 a 1 segundo en intervalos de milisegundos:

>>v=0:0.001:1

Intenta generar ahora tú los siguientes vectores

Impares de1 a 100; vector temporal de 0 a 2 segundos (intervalo: 2 décimas de segundo); pares de 100 a 0 (clave:intervalo!!!)

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Operaciones aritméticas (I)

+	adición o suma
–	sustracción o resta
*	multiplicación
'	traspuesta
^	potenciación
\	división-izquierda
/	división-derecha
.*	producto elemento a elemento
./ y .\	división elemento a elemento
.^	elevar a una potencia elemento a elemento

CUANDO SE APLIQUEN A MATRICES HAY QUE RESPETAR LAS REGLAS DE LA OPERACIÓN (ERROR MÁS COMÚN!!!); por ejemplo si tenemos A*B el número de columnas de A debe ser igual al número de filas de B; para sumar/restar deben tener la misma dimensión

A\B=inversa(A)*B; A/B=A*inversa(B); cuidado que no es lo mismo A*B que A*B en matrices!!!!!!!

Una primera aplicación muy útil (con lo que sabemos!!!!); resolver sistemas de ecuaciones lineales. A modo de ejemplo

$$x + 2y = 4,$$
$$2x - y = 3.$$

se puede poner en forma matricial de la siguiente forma: A · X=B; donde A es la matriz de los coeficientes; X el vector columna de incógnitas y B el vector columna de coeficientes; despejando esa igualdad se llega a X=inversa(A) · B.

A= 3 2 1
6 7 4

A.^2= 9 4 1
36 49 16

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Operaciones aritméticas (II)

$$\begin{aligned} x + 2y &= 4, \\ 2x - y &= 3. \end{aligned}$$

Tecleamos en Matlab

```
>> a = [1 2; 2 -1];  <Enter>
>> b = [4; 3];  <Enter>
>> x = a\b  <Enter>
```

El vector x contiene las soluciones (x e y)

Ejercicios (los puedes resolver!!!!!).

Dados A y B como

$$\mathbf{A} = \begin{pmatrix} 3 & 8 & -1 \\ 5 & 2 & 0 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} -3 & 2 \\ 2 & 2 \\ -1 & 3 \end{pmatrix}$$

Determina:

a) la traspuesta de B (la llamamos C)

b) V=A-4 · C;

c) A/C y C\A

Determina un vector que contenga los cuadrados de los primeros 20 números pares

Resuelve el sistema de ecuaciones

$$\begin{aligned} 3x + 2y - z &= 10 \\ -x + 3y + 2z &= 5 \\ x - y - z &= -1. \end{aligned}$$

Podemos limpiar la pantalla de comandos con la instrucción **clc** (pruébalo); si queremos eliminar variables de la memoria hacemos **clear** A (A es lo que queremos eliminar); si las queremos eliminar todas tecleamos clear

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Seguimos.....

Matrices especiales

`eye`	Identity matrix
`linspace`	Vector with linearly spaced elements
`ones`	Matrix of ones
`rand`	Uniformly distributed random numbers and arrays
`randn`	Normally distributed random numbers and arrays
`zeros`	Matrix of zeros

Todas las instrucciones se manejan igual; primero se define el número de filas y de columnas; por ejemplo una matriz de unos de tamaño 3x2 sería ones(3,2); otro ejemplo: un vector fila con 20 ceros sería zeros(1,20). CUIDADO: si sólo se usa un argumento (N) se supone que es una matriz cuadrada de tamaño NxN; por ejemplo si ponemos A=zeros(10) nos generará una matriz de tamaño 10x10.

Operaciones sobre matrices

`det`	Determinant
`eig`	Eigenvalues and eigenvectors
`expm`	Matrix exponential
`inv`	Matrix inverse
`poly`	Characteristic polynomial
`rank`	Number of linearly independent rows or columns
`rcond`	Condition estimator
`trace`	Sum of diagonal elements
`{}\ and /`	Linear equation solution

Instrucción que se usará para el análisis de datos en la etapa de preprocesado de los datos cuando se hace un análisis de componentes principales (PCA)

Operadores relacionales y lógicos

Operadores relacionales

==	equal to	~=	not equal to
<	less than	>	greater than
<=	less than or equal to	>=	greater than or equal to

Hay que tener cuidado con estos operadores porque se aplican a TODOS los elementos de la matriz/vector

Ejercicio: a) genera una matriz X de números aleatorios entre 0 y 1 de tamaño 4x2; b) observa la matriz; c) determina C=X>0.5; d) visualiza C...¿qué observas?

Operadores lógicos

MATLAB dispone de tres operadores lógicos para combinar posibles condiciones

Uso	Matlab
y	&
o	|
no	~

A modo de ejemplo, si quiero saber qué elementos de una matriz X estarían entre 0 y 1 lo haría de la siguiente forma: C=(X>0) & (X<1).

Ejercicio: a) genera una matriz X de números aleatorios (usando randn) de tamaño 10x3; b) determina qué valores están entre 0 y 0.5.

Una instrucción muy útil en el análisis de datos es la instrucción **find;** se utiliza de la siguiente forma [a,b]=find(condición); por ejemplo si en la matriz x quiero encontrar los valores que cumplen que son mayores que 10 haría: [a,b]=find(x>10); devuelve un vector a donde se indican las filas y un vector b que indican las columnas a(1) va con b(1); a(2) con b(2).....

Ejercicio: a) genera una matriz X de números aleatorios (usando randn) de tamaño 5x5; b) determina qué valores están entre -1 y 1 (interpreta a y b)

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Más instrucciones útiles para nuestro trabajo

l=length(v)	Si v es un vector devuelve la longitud del vector.
[a,b]=size(X)	Devuelve el número de filas y el de columnas de la matriz X.
m=mean(X)	Si X es un vector (fila o columna) determina su valor medio; si es una matriz devuelve un vector con la media de de cada columna
s=std(X)	Determina la desviación estándar en las mismas condiciones que el valor medio.
[M,I]=max(X)	Si X es un vector devuelve el valor máximo de un vector (M) y su posición (I); si es una matriz devuelve esos valores por cada columna de la matriz.
[M,I]=min(X)	Determina el mínimo en las mismas condiciones que el valor máximo.

Ejercicio: a) genera una matriz X de números aleatorios (instrucción randn) de tamaño 50x5; b) aplica las instrucciones vistas en esta tabla e intenta comprender cada uno de los resultados.

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Celdas y texto

Matlab dispone de otro tipo de elementos que nos van a aparecer en este curso: las celdas. Estos elementos son como matrices pero en los que, en cada elemento de la celda se puede definir cualquier cosa (una matriz, texto, etc). La instrucción para definir una celda de tamaño mxn es **cell(m,n).**

A nivel de ejemplo quiero definir una celda (c) de tamaño 2x2 con diferentes elementos; ejecuta las siguientes instrucciones: x=cell(3,2); x{1,1}='Pepe'; x{1,2}=randn(1,10); x{2,1}=eye(2); x{2,2}='Atópico'; x{3,1}=[0.1 0.2]; x{3,2}='Si';

Del ejemplo hay que destacar la dos diferencias con cualquier matriz vista hasta ahora: puedo tener diferentes elementos (texto y numéricos) y a los elementos de las celdas no se accede con los paréntesis () sino que se usan las llaves {}.

Otro elemento que aparece en el ejemplo es texto; nos encontraremos estos elementos al cargar los ficheros de datos al aparecer los nombres de las variables.

Ejercicio: define una celda de tamaño 2x2 con los siguientes elementos: tu nombre; vector columna de unos (10 elementos); matriz aleatoria 3x3; primer apellido); determina posteriormente el valor medio de la matriz aleatoria.

Primeras gráficas (I)

Matlab es un lenguaje que ofrece gran cantidad de posibilidades gráficas (aquí veremos las más sencillas!!!!); en el tema de preprocesado veremos otras. Teclea lo siguiente en la pantalla de comandos (tras el ; pulsa enter) x=0:pi/100:100;y=sin(x); plot(x,y); ¿Entiendes las etapas?; lo obtenido?; usa help para aprender la función sin (LA AYUDA DE MATLAB ES IMPRESCINDIBLE!!).

Se pueden modificar todos los parámetros del gráfico usando instrucciones pero hay un modo más fácil:

a) En la pantalla gráfica pulsa el icono

b) Seguidamente pulsa dos veces (rápido) sobre cualquier elemento de la gráfica abriéndose un menú donde puedes modificarlo TODO

Ejercicio: En la figura anterior modifica el tamaño, tipo y color de la línea; pon etiquetas a los ejes, título a la gráfica y ponle una rejilla a la figura.

Matlab, por defecto, siempre usa la misma ventana gráfica; si queremos hacer otra figura y que Matlab no nos machaque la anterior hay que usar la instrucción **figure (sin argumentos)** antes de hacer la nueva figura.

Ejercicio: Tomando como base lo anteriormente visto representa, en otra figura, coseno(x), instrucción cos donde x ahora va de 0 a 4π en intervalos de 0.1π

Si queremos ahora cerrar pantallas gráficas se usa **close(i)** *donde i es el índice de la pantalla que queremos cerrar; si queremos cerrarlas todas usamos* **close all;** *si lo que queremos es borrar usamos* **clf(i)**

Ejercicio: Tomando las anteriores gráficas; comprueba el uso de estas instrucciones.

Ahora queremos comparar dos resultados EN LA MISMA GRÁFICA; para ello se usa la instrucción **hold on;** a partir de esta instrucción todas las gráficas irán a la misma pantalla para desactivarla se usa **hold off;** *Ejercicio: Representa, en la misma gráfica un coseno y seno de un ángulo que varía de 0 a π en intervalos de 0.001π .*

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Primeras gráficas (II)

Ahora quiero dividir la pantalla gráfica en varias partes para poner varios resultados; la forma de hacer es con la instrucción **subplot(mns)**, donde m,n y s son parámetros. Para entenderlo se considera la división de la pantalla como una matriz; m es el número de filas y n es el número de columnas siendo s el índice que indica en qué división va la siguiente figura (se numera la división de izquierda a derecha y de arriba abajo).

Teclea los siguientes ejemplo, intenta comprender todos los pasos y lo que obtienes

```
t = 0:.1:2*pi;
subplot(2,2,1)
plot(cos(t),sin(t))
subplot(2,2,2)
plot(cos(t),sin(2*t))
subplot(2,2,3)
plot(cos(t),sin(3*t))
subplot(2,2,4)
plot(cos(t),sin(4*t))
```

Se pueden combinar diferentes tipos de subplot; teclea el siguiente ejemplo y observa el resultado.

Hay que tener en cuenta que, si quieres hacer algo sobre alguna gráfica primero le tienes que indicar a MATLAB la gráfica mediante esta instrucción.

```
x1=1:10;
x2=randn(1,10);
x3=0:2:18;
subplot(221)
plot(x1)
subplot(222);
plot(x2)
subplot(212)
plot(x3)
```

Para limpiar cualquier una determinada gráfica cuando se tienen varias porque se ha usado subplot; primero se indica cual se quiere borrar (usando subplot!!!!) y, posteriormente, se usa el comando **cla.** Como ejercicio intenta limpiar la gráfica superior izquierda del último ejemplo.

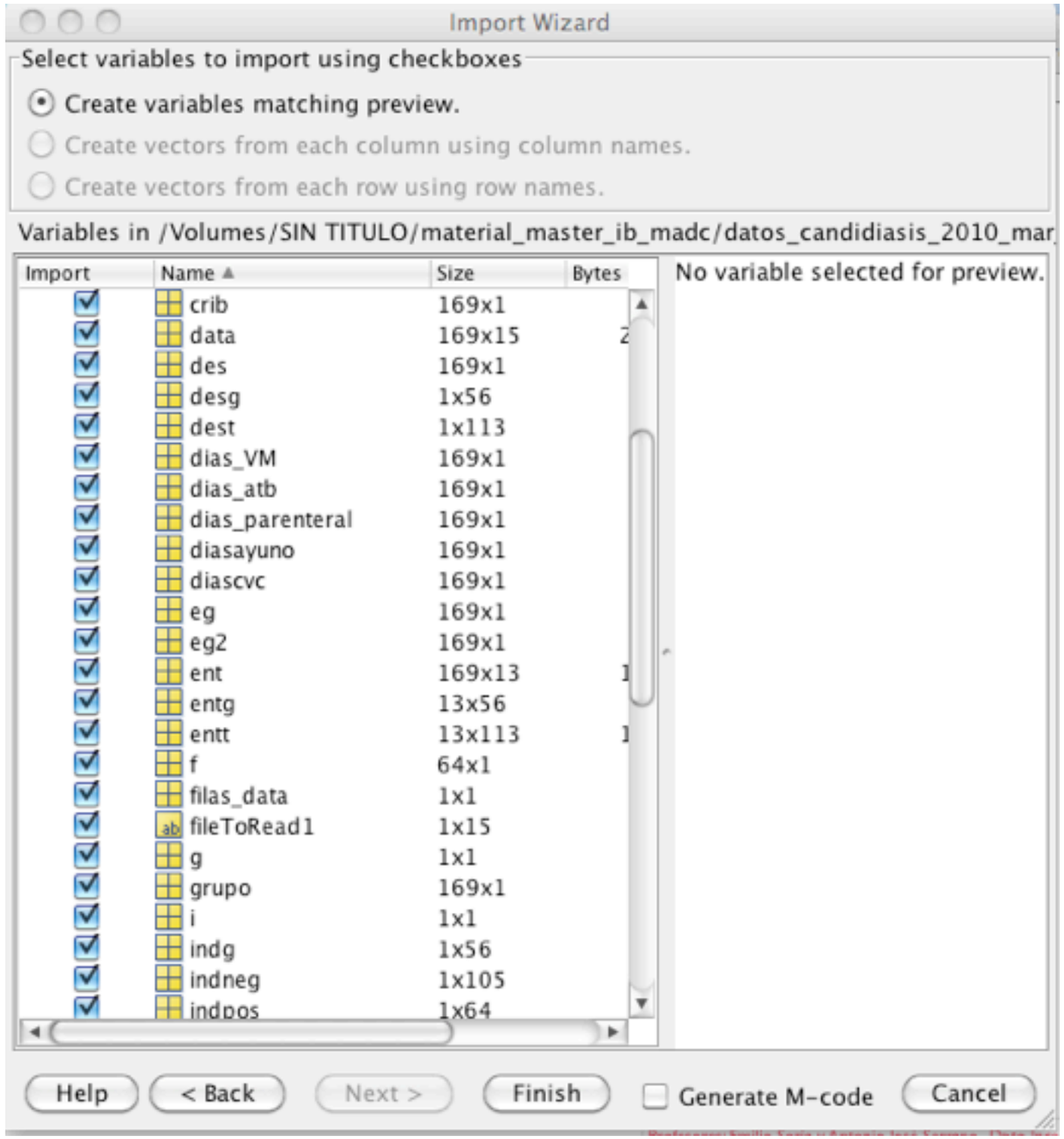
Si se quiere imprimir una gráfica para luego hacer un documento (por ejemplo la queremos insertar en Word) la instrucción directa es **print (mira la ayuda)** aunque tenemos otra opción más fácil; en el menú de la pantalla gráfica seguimos la secuencia File-Save as y ahí tenemos un montón de opciones (las que mejor quedan las .eps).

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Ficheros (I).

Matlab puede cargar diferentes tipos de ficheros; la instrucción a utilizar es **load** (mirar la ayuda) para ficheros .mat, .dat, .txt; **xlsread** para ficheros excel; **imread** para imágenes..... Más fácil es usar el asistente de Matlab que nos facilita la tarea; la secuencia a seguir es File-Import Data. Se nos abrirá un menú típico de búsqueda de archivos y, cuando encontremos el que buscamos, tendremos algo parecido a la siguiente pantalla (puede cambiar el aspecto según la versión); escogemos las variables de interés y pulsamos Finish



Ejercicio: Intenta cargar los ficheros *radondata.txt; ecg.mat* cuando hagas una carga y compruebes que tienes las variables en memoria, bórralas (¿te acuerdas cómo?) antes de cargar el siguiente fichero. Representa las variables del fichero ecg.mat; ¿qué son?

Ficheros (II).

Hemos terminado de trabajar y queremos guardar nuestro trabajo; la instrucción a usar es **save**; de la siguiente forma

save variables a guardar nombre del fichero opciones;

por ejemplo si usamos save A rollo guardaremos la variable A en el fichero rollo.mat; si no especificamos ninguna variable guardará todas. A continuación se muestran las opciones de tipo de dato guardado y versión de Matlab que podría leer el fichero guardado

MAT-file format Options	How Data Is Stored
`-ascii`	Save data in 8-digit ASCII format.
`-ascii -tabs`	Save data in 8-digit ASCII format delimited with tabs.
`-ascii -double`	Save data in 16-digit ASCII format.
`-ascii -double -tabs`	Save data in 16-digit ASCII format delimited with tabs.
`-mat`	Binary MAT-file form (default).

version Option	Use When Running ...	To Save a MAT-File That You Can Load In ...
`-v7.3`	Version 7.3 or later	Version 7.3 or later
`-v7`	Version 7.3 or later	Versions 7.0 through 7.2 (or later)
`-v6`	Version 7 or later	Versions 5 and 6 (or later)
`-v4`	Version 5 or later	Versions 1 through 4 (or later)

Ejercicio: Guarda la variable del ECG abdominal (¿sabes cómo conseguirla?) en un fichero ascii (16 dígitos) y que lo pueda leer la versión 7 o posteriores

Bucles y condicionales (I)

En el análisis de datos aparecen determinados bucles y condicionales para variar las condiciones iniciales de búsqueda de los parámetros por lo que hay que conocer las instrucciones básicas y "clásicas".

BUCLE FOR

Expresión en Matlab

for k=1:n
ACCIÓN
end

La primera linea indica el número de veces que se realizará la operación; ese índice se puede usar dentro de la operación claro!!!; el **end** final SIEMPRE HAY QUE PONERLO; fuente de error común!!!

El bucle for se utiliza cuando se conoce el número de veces que hay que realizar una operación; por ejemplo, supongamos que queremos calcular la suma de los elementos de un vector x; la siguiente secuencia de instrucciones lo hace; 1) definir el vector x; 2) suma=0; 3) l=length(x); 4) for k=1:l, suma=suma+v(k); end 5) suma ..ES IMPORTANTE QUE ENTIENDAS LOS PASOS REALIZADOS.

EJERCICIO: Modifica la secuencia anterior de instrucciones para calcular, usando un bucle for, el factorial de un número ($n!=1\cdot 2\cdot 3\cdot 4...\cdot n$).

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA

Bucles y condicionales (II)

BUCLE WHILE

Expresión en Matlab

while condición
ACCIÓN
end

Mientras se cumpla la condición se realizará la acción; otra vez cuidado con el **end** final SIEMPRE HAY QUE PONERLO; fuente de error común!!!. Aquí otro error común es no verificar que la condición no se cumple en algún momento.

El bucle while se utiliza cuando NO se conoce el número de veces que hay que realizar una operación; pero esa operación depende de una condición. El ejemplo de la suma anterior sería 1) definir el vector x; 2) suma=0; 3) I=length(x); 4) while I>0 suma=suma+v(I); I=I-1; end 5) suma ..ES IMPORTANTE QUE ENTIENDAS LOS PASOS REALIZADOS. Ejercicio: intenta repetir el ejercicio del factorial con el bucle while

CONDICIONALES

Expresión en Matlab

if condición
ACCIÓN 1
else
ACCIÓN 2
end

Si se cumple la condición se realizará la acción 1; en caso contrario se realiza la acción 2; otra vez cuidado con el **end** final SIEMPRE HAY QUE PONERLO; fuente de error común!!!. La parte del else se puede eliminar. Además se pueden encadenar varios if con la instrucción elseif (MIRA LA AYUDA). Esta instrucción la usaremos a continuación

Ficheros .m

Si queremos combinar muchas instrucciones (programar!!!); tenemos que ponerlas juntas en un fichero y decírselo a MATLAB. Los pasos a seguir 1) serían pulsar el icono de la hoja en blanco (editor); 2) se escriben las instrucciones en el editor; 3) se guarda con la extensión .m (no se empieza con número ni dejes espacios en blanco) y asegúrate de guardarlo en tu directorio de trabajo o en el path (MIRA LA AYUDA); 4) ejecútalo desde la pantalla de comandos escribiendo el nombre del programa.

El anterior fichero se conoce como *script;* existe otro tipo de fichero que son *las funciones* en las que se les pueden pasar unos parámetros. Se sigue la misma secuencia que en el caso anterior pero aparecen dos diferencias; a) la cabecera PRIMERA LÍNEA DEL PROGRAMA que tendría la siguiente forma **function** [a,b,...z]=nombre(p1,p2,..pn) donde a,b...z son los parámetros de salida de la función y p1,p2,...pn los parámetros de la función; b) la función se guarda con nombre.m.La manera de ejecutarlo desde la línea de comandos sería de la misma forma que en la primera línea de dicha función

Ejercicio; implementa la siguiente función y ejecútala comprobando su funcionamiento

```
function [a,b]=ejemplo(x,y)
 a=0.5*(x+y);
b=(x.*y).^.5
```

Ejercicio; implementa una función que, a partir del peso y la estatura, determine el índice de masa corporal

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

Acabamos.....

Descárgate el documento Introduccion Al Matlab de la página web de la asignatura;

Sólo nos centraremos en los dos primeros puntos; manejo de matrices y bucles. El resto se puede hacer usando la ayuda de las instrucciones (el famoso help!!!).

Sólo hemos visto una pincelada.....pero con esto y la ayuda de Matlab podemos hacer muchas cosas!!!!

VNIVERSITAT DE VALÈNCIA

MASTER DE INGENIERÍA BIOMÉDICA.

Guía para la realización de los trabajos (I)

Introducción al Matlab

Profesores: Emilio Soria y Antonio José Serrano, Dpto Ingeniería Electrónica, ETSE

VNIVERSITAT DE VALÈNCIA