Universidad de Oviedo
Departamento de Matemáticas

Facultad de Ciencias
Libre config. Mat./Fís.

Resumen de MATLAB[1]

1 La ayuda (Help)

- **La ayuda en línea.**

  La orden `help` sirve para obtener información sobre un tema o un comando concreto.

  Ejemplo: `help sqrt` proporciona información sobre el comando `sqrt`.

  Si no se conoce la orden exacta sobre la que deseamos ampliar la información, se puede escribir simplemente `help` para obtener una lista de temas de ayuda.

  Otras órdenes:

  - `more on` configura la orden `help` para que la información se muestre *pantalla a pantalla*. Se desactiva con `more off`.
  - `lookfor XYZ` busca la cadena `XYZ` en la primera línea de comentario de todos los ficheros de ayuda.

- **La opción Help del menú principal.**

  Es una forma de obtener ayuda mediante menús desplegables, como en la mayoría de los programas que funcionan bajo *Windows*. Se necesita ratón.

  También se puede acceder a estos menús con la orden `helpwin`.

2 El entorno operativo de MATLAB para Windows

Las órdenes siguientes son de propósito general. La forma de usarlas se puede consultar en la ayuda (`help`).

`cd`	Cambia de directorio.
`clc`	Limpia la pantalla.
`clear`	Elimina variables.
`close`	Cierra las ventanas del entorno gráfico.
`pwd`	Muestra el directorio elegido.
`type`	Lista un archivo.
`who`	Muestra las variables utilizadas.
← → ↑ ↓	Recuperan órdenes (como en DOS).

Como **norma general**, al comenzar una sesión de MATLAB cada usuario se situará en el directorio `c:\temp` tecleando `cd c:\temp` o bien `cd c:/temp`.

3 Manipulación de matrices

- En MATLAB no hay sentencias de dimensión ni de declaración de tipos, la memoria se gestiona de forma automática. Una de las formas de introducir una matriz es por listado explícito de sus elementos. Los elementos de una misma fila se separan por espacios blancos o por comas y las distintas columnas se separan por punto y coma. La matriz queda delimitada por corchetes:

$$A = [2\ \ -1\ \ 0\,;\ \ -1\ \ 2\ \ -1\,;\ \ 0\ \ -1\ \ 2].$$

[1] **Página web de la asignatura:** *http://orion.ciencias.uniovi.es/∼salim/*

- Con la orden $A(i:j\ ,\ k:m)$ extraemos la submatriz de A formada por las filas desde la i hasta la j, y columnas desde la k hasta la m. Probar con

$$A(1:2\ ,\ 1:2),\qquad A(3\ ,\ :)\qquad A(:\ ,\ 2:3)\qquad A(:\ ,\ [1\ \ 3])\quad \text{y}\quad A(:).$$

- La expresión $1:4$ representa el vector fila [1 2 3 4]. Los números no tienen que ser necesariamente enteros ni el incremento siempre igual a uno. Probar por ejemplo con:

$$0.2:0.2:1.3\qquad \text{y}\qquad 5:-1:1.$$

- Las matrices grandes pueden construirse a partir de otras más pequeñas. Si quisiera añadir la fila $b = [5:-1:3]$ a la última fila de A escribiría $[A\,;\ b]$. Si quisiera añadir b a la ultima columna debería escribir $[A\ b']$ ya que las matrices que se vayan pegando han de tener dimensiones coherentes.

4 Operaciones con matrices

Las operaciones entre matrices son las habituales, y solamente habrá que cuidar las dimensiones.

suma	resta	multiplicación	división	potencia	traspuesta
$A+B$	$A-B$	$A*B$	$A\backslash B$	$A\hat{}n$	A'

Si b es un vector columna

$$A\backslash b\qquad \text{resuelve el sistema}\qquad Ax = b.$$

La división $A\backslash B$ es equivalente en cuanto a resultado a $\text{inv}(A)*B$, pero no en cuanto al número de operaciones por ser distintos procesos.

5 Operaciones con *Arrays*

Se refieren a operaciones que se realizan elemento a elemento en lugar de las operaciones matriciales usuales definidas en las sección anterior. En todos los casos los arrays implicados (matrices o vectores) han de tener las mismas dimensiones. Para indicar que se trata de una operación entre *arrays* se precede el operador de un punto, por ejemplo $(.*)$, $(.\backslash)$, $(./)$ o $(.\hat{})$. Por ejemplo:

`a.*b`	=	`[a1*b1,a2*b2,...,an*bn]`	*producto* de dos vectores.
`a./b`	=	`[a1/b1,a2/b2,...,an/bn]`	*cociente a la derecha* de dos vectores.
`a.\b`	=	`b./a`	*cociente a la izquierda* de dos vectores.
`a.^c`	=	`[a1^c,a2^c,...,an^c]`	vector *elevado a* escalar.
`c.^a`	=	`[c^a1,c^a2,...,c^an]`	escalar *elevado a* vector.
`a.^b`	=	`[a1^b1,a2^b2,...,an^bn]`	vector *elevado a* vector.

6 Funciones que operan elemento a elemento sobre matrices

La librería MATLAB dispone de una gama muy completa de funciones predefinidas que se corresponden con las funciones matemáticas más utilizadas. Algunos ejemplos de estas funciones son:

`abs(x)`	valor absoluto de x	`sqrt(x)`	raíz cuadrada de x
`sin(x)`	seno de x	`cos(x)`	coseno de x
`asin(x)`	arcoseno de x	`acos(x)`	arcocoseno de x
`tan(x)`	tangente de x	`atan(x)`	arcotangente de x
`exp(x)`	exponencial de x	`log(x)`	logaritmo en base e de x

7 Funciones que operan sobre vectores

La siguientes funciones se aplican esencialmente a vectores (fila o columna), pero actúan también sobre una $m \times n$ matriz columna por columna produciendo un vector con el resultado de su aplicación a cada columna. Por ejemplo `max(max(A))` devuelve el mayor elemento de la matriz A.

`max`	máximo elemento	`min`	mínimo elemento
`sum`	suma de elementos	`norm`	norma de vectores y matrices
`sort`	ordenación de elementos	`reshape`	reestructuración de matrices
`mean`	valor medio	`tril`	parte triangular inferior

8 Funciones que operan sobre matrices

`eig`	autovalores y autovectores	`chol`	factorización de Cholesky
`inv`	inversa	`lu`	factorización LU
`det`	determinante	`size`	tamaño
`cond`	número de condición	`expm`	matriz exponencial

9 Matrices especiales

`diag`	matriz diagonal
`eye`	matriz identidad
`ones`	matriz de unos
`zeros`	matriz nula
`linspace`	genera un vector de componentes linealmente espaciadas
`meshgrid`	genera las coordenadas de una malla bidimensional

10 Los ficheros en MATLAB

Los ficheros de instrucciones MATLAB llevan la extensión `.m`. Se distinguen dos tipos:

- **Ficheros de programas.** Son *m-ficheros* que no constituyen funciones y que se construyen mediante una secuencia de instrucciones. El contenido de un fichero de programas MATLAB nombre.m se ejecuta tecleando simplemente su nombre.
- **Ficheros de función.** Son aquellos cuya primera línea ejecutable (no de comentario) comienza con la palabra `function`.

  Una función se define con un m-fichero, cuyo nombre coincide con el de la función. La primera línea ejecutable es:

  ```
  function argumentos_salida=nombre_función (argumentos_entrada)
  ```

  seguida de las instrucciones necesarias. Cuando hay más de un argumento de salida, éstos deben ir entre corchetes y separados por comas. Por ejemplo:

  ```
  function y=f(x)
  function [a,b,c]=g(x,y)
  ```

  Es conveniente comenzar las primeras líneas del fichero con un comentario (iniciándolas con el símbolo `%`), explicando cómo debe usarse la función y sus argumentos (tanto de entrada como de salida). De esta manera, dicha explicación será visible mediante la instrucción `help nombre_función`.

  La función puede finalizarse en cualquier momento utilizando la instrucción `return`.

11 Instrucciones de entrada y salida

- `x=input('mensaje'[,'s'])`. Permite la introducción de datos por pantalla. La opción `'s'` se emplea para leer una variable de tipo carácter (`'string'`), evitando los apóstrofos.

  Ejemplos:

  ```
  a=input('Número de filas:  ').
  m=input('Nombre del fichero:  ','s').
  ```

- `disp('mensaje')` ó `disp('texto')`. Muestra un texto o una matriz de texto por pantalla. Para combinar información numérica y texto en un comando `disp` se puede utilizar la instrucción `num2str`.
- `pause`. Detiene la ejecución del programa (véase la ayuda).

12 Lectura y escritura en ficheros externos

Los ficheros externos tienen *nombre* y *extensión*. Si la extensión no se indica, MATLAB considera que es `.mat`.

- La orden `save`
  - `save nombre`: guarda en el fichero binario `nombre.mat` todas las variables en uso.
  - `save nombre X Y Z`: guarda en el fichero binario `nombre.mat` las variables `X`, `Y` y `Z`.
  - `save nombre.ext X Y Z -ascii`: guarda en el fichero ASCII `nombre.ext` las variables `X`, `Y` y `Z` con precisión simple.
  - `save nombre.ext X Y Z -ascii -double`: guarda en el fichero ASCII `nombre.ext` las variables `X`, `Y` y `Z` con precisión doble.
- La orden `load`
  - `load archivo`: lee las variables del fichero binario `archivo.mat`, que ha sido previamente generado con MATLAB.
  - `load archivo.ext`: lee el contenido del fichero ASCII `archivo.ext` y lo almacena como una única variable de nombre `archivo`.

13 Control de flujo

- **Bucle `for`**

  Permite que una sentencia, o grupo de sentencias, pueda ser repetida un numero fijo y predeterminado de veces. Pueden incluirse bucles anidados (unos dentro de otros). También debemos recordar que toda instrucción `for` debe ir acompañada de un `end`. Por ejemplo:

  ```
  for i=1:3
  for j=1:3
  A(i,j)=1/(i+j-1);
  end
  end
  ```

- **Bucle `while`**

  El bucle `while` permite que una sentencia o grupo de sentencias sean ejecutadas un número indefinido de veces mientras una expresión lógica sea verdadera. Si, por ejemplo queremos averiguar cuál es el mayor entero cuyo factorial es menor que 100:

```
n=1;
while prod(1:n)<100
n=n+1;
end
```

- **Estructuras `if-elseif-end`**

  En ocasiones se quiere ejecutar un conjunto de órdenes solo en el caso de que verifique cierta condición. Por ejemplo, se puede definir la función

$$f(x)=\begin{cases}1 & \text{si} \quad x<-1\\ x^2 & \text{si} \quad -1\le x\le 1\\ x & \text{si} \quad x>1\end{cases}$$

  mediante un M-fichero escribimos en un fichero llamado `f.m` la siguientes instrucciones:

```
function y=f(x)
if x<-1
y=1;
elseif x <= 1
y=x^ 2;
else
y=x;
end
```

- Las condiciones se escriben mediante los siguientes operadores relacionales y lógicos:

$<$	menor	$<=$	menor o igual	&	conjunción
$>$	mayor	$>=$	mayor o igual	|	disyunción
$==$	igual	$\sim=$	distinto	$\sim$	negación

14 Representaciones gráficas

14.1 Gráficos 2D

- Si $x=(x_1,\cdots,x_n)$, $y=(y_1,\cdots,y_n)$ son dos vectores la orden `plot`(x,y) dibuja el conjunto de puntos $\{(x_i,y_i)\}$ y los enlaza con segmentos. Utilizar la orden de ayuda `help` para ver las distintas opciones de `plot` para controlar el color, la marca y el tipo de trazo de la gráfica. Por ejemplo,

$$x=0:0.1:2*\texttt{pi};\quad \texttt{plot}(x,\texttt{sin}(x),\,'r--')$$

  representa la función `sen`(x) en color rojo y con trazo discontinuo.

- La orden `hold on` mantiene activa la ventana gráfica actual. Es útil para superponer varios dibujos en una misma ventana. Con `hold off` se realiza cada gráfico en una ventana diferente. Es la opción por defecto.

- Se puede subdividir una ventana gráfica mediante la orden `subplot`. `subplot`(m,n,p) divide la ventana gráfica en $m\times n$ subventanas distribuidas en m filas y n columnas y coloca el gráfico actual en la ventana p-ésima, contando de izquierda a derecha y de arriba abajo.

- `Matlab` dispone de las siguientes órdenes para poner texto en un gráfico y controlar la escala del dibujo:

`title`('texto')	sitúa el texto como título
`xlabel`('texto')	sitúa el texto como al lado del eje x
`ylabel`('texto')	sitúa el texto como al lado del eje y
`text`$(x,y,$ 'texto')	sitúa el texto en el punto (x,y) del gráfico
`grid on`	dibuja una malla sobre el gráfico
`axis`$[xmin,\ xmax,\ ymin,\ ymax]$	establece los valores máximos y mínimos para los ejes
`axis axis`	fija la escala de los ejes en los valores actuales (con `hold` en *on*)
`axis off`	elimina la malla y los ejes
`zoom`	permite ampliar un gráfico

14.2 Gráficos 3D

Para representar una superficie $z=f(x,y)$ es necesario comprender el funcionamiento de la orden `meshgrid` que genera el soporte del dibujo.

Dados los vectores $x=(x_1,\cdots,x_n)$, $y=(y_1,\cdots,y_m)$ la orden

$$[X\ Y]=\texttt{meshgrid}(x,y)$$

genera una matriz X de dimensión $m\times n$ cuyas filas son m copias del vector x, y una matriz Y de dimensión $m\times n$ cuyas columnas son n copias del vector y:

$$X=\begin{pmatrix}x_1 & \dots & x_n\\ \vdots & \ddots & \vdots\\ x_1 & \dots & x_n\end{pmatrix}\qquad Y=\begin{pmatrix}y_1 & \dots & y_1\\ \vdots & \ddots & \vdots\\ y_m & \dots & y_m\end{pmatrix}.$$

A partir de X e Y se genera la matriz

$$Z=f(X,Y)=\begin{pmatrix}f(x_1,y_1) & \dots & f(x_n,y_1)\\ \vdots & \ddots & \vdots\\ f(x_1,y_m) & \dots & f(x_n,y_m)\end{pmatrix}$$

y la superficie se representa la superficie mediante una de las siguientes órdenes:

`mesh`$(X,\ Y,\ Z)$, `surf`$(X,\ Y,\ Z)$

`contour`$(X,\ Y,\ Z)$, `pcolor`$(X,\ Y,\ Z)$.