Docencia 2007-2008
Máster en Matemáticas y Aplicaciones
Métodos numéricos
Prácticas con MATLAB

Ficha de trabajo en clase 1
Introducción a MATLAB
04 de octubre de 2007

A) Escalares, vectores, matrices

En MATLAB, el símbolo "=" se llama operador de asignación. Asigna a una variable cuyo nombre está especificado en la parte izquierda del operador un valor numérico especificado en la parte derecha del operador.

Ejercicio 1: Definir dos variables a y b cuyos valores asignados sean 4 y 5. Calcular $a+b$, $a-b$, $b-a$, ab, a/b, a^b y b^a. Definir una variable x que tome el valor $2a+5b$. Redefinir x como $3(x+6(b-a))$.

En MATLAB, el número iracional π toma la forma de una variable predefinida denotada por `pi`*. El formato numérico por defecto es* `short` *(punto fijo con 4 dígitos decimales). Otros formatos numéricos:* `long` *(punto fijo con 14 dígitos decimales),* `short e`*,* `long e`*,* `short g`*,* `long g`*,* `rational` *(se aproxima un número real por un cociente de dos enteros). Para cambiar el formato numérico, se usa el comando* `format`*, seguido por el nombre del formato numérico al que queremos pasar (por ejemplo,* `format long e`*).*

Ejercicio 2: Calcular π en cada uno de los formatos numéricos antes mencionados. Definir dos variables c y d cuyos valores sean $\sin(\pi/6)$ y $\tan(\pi/6)$ y calcular $c+d$, c^2+d^2, $1-c^2$, $\sqrt{c}$, e, e^d, $|-c-1|$, $\cos(\pi/2)/\sin(\pi)$ y $e/\sin(\pi)$.

Para crear una variable vectorial a partir de una lista de números o expresiones numéricas conocidas, se teclean sus componentes $e_1, \cdots, e_n$ separadas por coma (,) o por espacio dentro de un par de corchetes ([ ]):

$$vect = [e_1 \ e_2 \ \cdots \ e_n].$$

Para introducir un vector de paso constante h a partir de los extremos a y b se escribe entre corchetes o sin corchetes $a:h:b$

$$vect = [a:h:b] \qquad o \qquad vect = a:h:b.$$

Por $a:b$ se denota el vector de paso constante $h=1$.
Para definir una matriz, se teclean entre corchetes los vectores filas separados por punto y coma (los vectores fila se pueden dar en cualquiera de las formas antes mencionadas, es decir, por sus componentes o como vectores de paso constante):

$$matriz = [fila_1; fila_2; \cdots; fila_m].$$

El símbolo ' se usa para denotar la traspuesta de una matriz. Cuando entre dos matrices $A = (a_{ij})_{i=\overline{1,m},j=\overline{1,n}}$ y $B = (b_{ij})_{i=\overline{1,m},j=\overline{1,n}}$ hay una operación aritmetica precedida por punto (.), el resultado es una matriz $C = (c_{ij})_{i=\overline{1,m},j=\overline{1,n}}$, cuyas componentes c_{ij} se obtienen realizando dicha operación entre los correspondientes componentes a_{ij} y b_{ij}.
Para encontrar los autovalores y los autovectores de una matriz A se usa el comando `eig`*:*

$$[eigenvectors, eigenvalues] = eig(A),$$

donde eigenvectors es una matriz cuyas columnas son los autovectores de A y eigenvalues es una matriz diagonal, los componentes de la diagonal siendo los autovalores de la matriz A.

Ejercicio 3: Definir los vectores $vect_1 = (1,\ 7,\ 60/5-1,\ 7*3-5,\ 4^2-5)$, $vect_2 = [2:0{,}2:3]$, $vect_3 = [2:10]$, $vect_4 = [40:-2:30]$, $vect_5 = 55:-3:36$, $vect_6 = 3:12$, $vect_7 = 1:11$.
Definir $vect_8 = vect_6'$. Calcular $vect_6 + vect_8$, observar y corregir.
Calcular $vect_7+3$, $vect_6.*vect_7$, $vect_6./vect_7$, $(vect_6-vect_7).^3+9$, $vect_6.^2$, $vect_6-4$, $vect_6*vect_6$ y corregir los eventuales errores que aparecen.

Definir la matriz $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$. Calcular A', $A*A'$ y $A.*A'$. Definir el vector $b = (1,\ 3,\ 5)$.
Calcular $[b;A]$ (*añade una fila dada por el vector* b *a la matriz* A) y $[b'\ A]$ (*añade una columna dada por* b' *a la matriz* A).

Redefinir la matriz $A = \begin{pmatrix} 1 & 4 & -3 \\ 2 & 1 & 5 \\ -2 & 5 & 3 \end{pmatrix}$. Definir B como la inversa de A (*a través del comando inv(A)*) y calcular $A*B$, $A-B$, $A+B$, B', B^3, $B.^3$. Resolver el sistema $A*x = b'$. Encontrar los autovalores y los autovectores de la matriz A.

Consideremos m, n dos enteros positivos. A través del comando `eye(n)` *se introduce la matriz identidad de tamaño n. Mediante los comandos* `ones(m,n)`, `zeros(m,n)` *se introducen matrices de m filas y n columnas con todos los elementos 1, respectivamente 0; cuando estos comandos sólo tienen un argumento (*`ones(n)`, `zeros(n)`*), se obtienen matrices cuadráticas de tamaño n con todos los elementos 1 o 0.*
Si x es un vector, el comando `diag(x)` *produce una matriz diagonal con el vector x en la diagonal principal. El comando* `diag(x,k)`*, donde k es un entero, produce una matriz cuadrada diagonal, con el vector x colocado en la k-esima diagonal inferior si k es negativo, en la k-esima diagonal superior si la k es positivo y en la diagonal principal si $k = 0$. Si A es una matriz, el comando $diag(A)$ produce un vector columna que almacena la diagonal principal de dicha matriz.*
El comando `size(A)`*, donde A es una matriz $m \times n$, tiene como salida el vector $[m, n]$. El comando* `length(x)`*, donde x es un vector de n componentes, tiene a n como salida.*
Los comandos `triu(A)` *y* `tril(A)` *generan una matriz triangular superior, respectivamente inferior, a partir de una matriz cuadrada A.*

Ejercicio 4: Calcular $eye(5)$, $ones(2,3)$, $ones(5)$, $zeros(4,2)$, $zeros(4)$, $diag(vect_6)$, $diag(A)$, $diag(diag(A))$, $size(A)$, $length(vector_7)$, $size(vector_6)$, $triu(A)$, $tril(A)$, $size(vector_6')$, $diag(vect_6,-3)$. Definir una matriz $M = diag(ones(5,1)) + diag(ones(3,1),-2) + diag(ones(4,1),1)$.
Usar el comando `spy(M)` para visualizar en una figura los elementos no-triviales de la matriz M.

Si x es un vector de n componentes, el comando $x(j)$, $j = \overline{1,n}$, tiene como salida la j-esima componente del vector. Si A es una matriz $m \times n$, el comando $A(i,j)$, $i = \overline{1,m}$, $j = \overline{1,n}$, tiene como salida el elemento (i,j) de dicha matriz.
Denotemos por `index_1`, `index_2` *dos vectores de enteros positivos en el rango $\overline{1,m}$, respectivamente $\overline{1,n}$. El comando* `A(index_1, index_2)` *tiene como salida la matriz resultada al intersectar las filas de indices contenidos en* `index_1` *con las columnas de indices contenidos en* `index_2`.
`A(:,index_2)` *tiene como salida la matriz resultada al intersectar todas las filas con las columnas de indices contenidos en el vector* `index_2`.
`A(end,index_2)` *tiene como salida el vector resultado al intersectar la última fila con las columnas de indices contenidos en* `index_2`.

Ejercicio 5: Definir la matriz

$$A = \begin{pmatrix} 35 & 1 & 6 & 26 & 19 & 24 \\ 3 & 32 & 7 & 21 & 23 & 25 \\ 31 & 9 & 2 & 22 & 27 & 20 \\ 8 & 28 & 33 & 17 & 10 & 15 \end{pmatrix}$$

y el vector $x = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix}$.

Calcular $x(4)$, $x(end)$, $x(2:5)$, $x(1:2:7)$, $x([2,4:end])$, $A(2,3)$, $A(3,1:4)$, $A(3,:)$, $A(:,4)$, $A(end,:)$, $A(end,3:5)$, $A(:,[1,\ 3:6])$, $A(1:3,3:6)$, $A([4,2],[3,1,5])$.

B) Representaciones gráficas.

Para realizar gráficos 2-dimensionales se usa el comando `plot(x,y,option)`*, donde x e y son vectores de la misma longitud y* `option` *es una opción que indica el color y el estilo de la línea.*

Ejercicio 6: Teclear el siguiente conjunto de comandos:

```
clear
z=[-3*pi:0.1:3*pi]
plot(z,sin(z),'r*')
hold on
```
(*este comando mantiene los dos dibujos en la misma figura*)
```
plot(z,cos(z),'bo').
```

Se pueden dar más detalles sobre una figura usando comandos del tipo `title('')` *- añade título al gráfico,* `legend` *- añade una leyenda al gráfico,* `grid` *- añade una cuadricula al gráfico,* `grid off` *- hace desaparecer la cuadricula.*

Para realizar peliculas, se usan los comandos `M=moviein(n)` *- reserva memoria para almacenar n frames,* `movie(M,i,j)` *- hace que visualicemos la pelicula almacenada en* `M` *i veces a la velocidad de j imagenes por segundo.*

Ejercicio 7: Teclear el siguiente conjunto de comandos:

```
clear
z=[-2*pi:0.1:2*pi];
M=moviein(17);
for j=1:17
f=sin(z+j*pi/17);
plot(z,f,'b*')
M(:,j)=getframe;
end
movie(M,10,7).
```

Los siguientes dos programas tienen como objetivos la inicialización al uso del Editor de MATLAB, a la definición de funciones en MATLAB y a las representaciones gráficas 3-dimensionales.

Ejercicio 8: Primero, vamos a definir una función `test3d` en un fichero llamado `test3d.m`:

```
function Z=test3d(X,Y)
Z=3*(1-X).^2.*exp(-X.^2-(1+Y).^2)-10*(X/5-X.^3-Y.^5).*exp(-X.^2-Y.^2)
        -1/3*exp(-(X+1).^2-Y.^2)
```

Observar que el nombre de la función debe coincidir con el nombre del fichero la guardamos.

Vamos a escribir un otro fichero tipo .m que realiza la vizualización 3-d de la función `test3d` antes definida:

```
clear  all
u=[0:pi/20:6*pi];
figure(1)
```
(*se abre una ventana gráfica dividida en cuatro sub-ventanas dispuestas en dos lineas y dos columnas*)
```
subplot(2,2,1)
plot3(u,sin(u),cos(u),'b*')
```
(*el comando* `plot3` *facilita la visualización de curvas parametrizadas en tres dimensiones*)
```
x=[-3:0.1:3];
y=[-4:0.1:4];
[X,Y]=meshgrid(x,y);
```
(*el comando* `meshgrid`*, cuyos argumentos son dos vectores x e y de tamaño m y n, produce dos matrices $n \times m$, X, cuyas n filas son copias de x, e Y, cuyas m columnas son copias de y.*)

`Z=test3d(X,Y);` (*se aplica la función* `test3d` *a las matrices* X *e* Y*, obteniendose una matriz* $n \times m$*,* Z*.*)
`subplot(2,2,2)`
`meshc(x,y,Z)` (*los comandos* `mesh` *o* `meshc` *hace que la superficie sea visualizada como una malla; el segundo comando tiene la ventaja de que se puede visualizar tambien la forma de las curvas de nivel. El comando* `waterfall` *hace visualizar la superficie como una malla unidireccional.*)
`subplot(2,2,3)`
`surfc(Z)`(*los comandos* `surf` *o* `surfc` *hace que visualicemos el gráfico como una superficie continua; el segundo comando tiene la ventaja de que se puede visualizar tambien la forma de las curvas de nivel.*)
`subplot(2,2,4)`
`contour3(x,y,Z,16)` (*los comandos* `contour` *y* `contour3` *ayudan a visualizar las curvas de nivel de la superficie como proyección en el plano* xOy*, respectivamente en tres dimensiones.*)

Referencias

[1] Amos Gilat, Matlab, *Una introducción con ejemplos prácticos*, Editorial Reverté, Barcelona

[2] Javier García de Jalón, José Ignacio Rodríguez, Jesús Vidal, *Aprenda Matlab 7.0 como si estuviera en primero*, Universidad Politécnica de Madrid, Escuela Técnica Superior de Ingenieros Industriales, http://mat21.etsii.upm.es/ayudainf/aprendainf/Matlab70/matlab70primero.pdf

[3] David Griffiths, *An introduction to Matlab, Version 2.3*, http://www.maths.dundee.ac.uk/∼ftp/na-reports/MatlabNotes.pdf

[4] Desmond J. Highham, Nicholas Highham, *Matlab guide*, www.uam.es/personal_pdi/ciencias/fquiros/Numerico2_03_04/mg_final.pdf

[5] Maria Dolores Cárdenas, *Matlab como ayuda al estudiante en Ciencias Matemáticas*, http://www.uam.es/personal_pdi/ciencias/fquiros

[6] *Getting Started with Matlab 7*, http://www.mathworks.com/acces/helpdesk/help/pdf_doc/matlab/getstart.pdf

[7] Juan Antonio Infante, José Maria Rey, *Introducción a Matlab*, http://www.mat.ucm.es/∼jair/matlab/notas.pdf