Capítulo 3

Programación con MATLAB

Contenido:

3.1 Operaciones de relación

MATLAB permite realizar las comparaciones entre clases de datos más frecuentes. La tabla que sigue describe los símbolos que se utilizan en esta tarea.

Operador	**Significado**
<	Menor
>	Mayor
<=	Menor o igual
>=	Mayor o igual
==	Igual
~=	No igual

Ejemplos:

```
» A=[1 0 2 4 1 9 8 9];
» B=[0 1 6 5 2 -7 2 12];
» w=A>B
w =
    1    0    0    0    0    1    1    0
» ww=A<B
ww =
    0    1    1    1    1    0    0    1

» x='helecho';
» y='almejas';
» z=x<y
z =
    0    1    1    0    1    0    1
» z=x>y
z =
    1    0    0    0    0    1    0
```

3.2 Operadores lógicos

Las relaciones pueden ser combinadas, cuantificadas o negadas por medio delos operadores lógicos.

Operador	Significado
&	And
|	Or
~	Not

La orden any aplicada a un vector devuelve un uno si el vector tiene algún elemento no nulo. Si esta orden tiene como argumento una matriz, se aplica a cada columna de la misma. Análogamente all, devuelve un uno si todos los elementos del vector que figura como argumento son diferentes de cero. En caso contrario devuelve un cero.

La orden find aplicada a una matriz devuelve el número de elementos de la misma que son diferentes de cero.

3.3 Sentencias de control de flujo

a) Lazos controlados por un vector: Utilizan la orden for y tiene la estructura siguiente:

```
for variable=vector
        (sentencias…
          …)
end
```

Todas las sentencias comprendidas entre for y end se ejecutan tantas veces como columnas tenga el vector de control.

```
for a=1:5,v=[a, a^2,a^3],end
v =
    1    1    1
v =
    2    4    8
v =
    3    9   27
v =
    4   16   64
```

```
v =
     5    25   125
```

El vector de control puede describirse directamente y el lazo se puede construir escribiendo las sentencias en líneas diferentes:

```
» for a=[1 3 7]
v=sin(a)
end
v =
   0.8415
v =
   0.1411
v =
   0.6570
```

Como es habitual en otros lenguajes de programación, se permiten los lazos anidados:

```
» for i=1:3
for j=1:5
H(i,j)=1/(i+j-1);
end
end
» H
H =
   1.0000   0.5000   0.3333   0.2500   0.2000
   0.5000   0.3333   0.2500   0.2000   0.1667
   0.3333   0.2500   0.2000   0.1667   0.1429
```

Los lazos se ejecutan para vectores de índices no enteros:

```
ind=linspace(0,1,10);
» for i=ind
seno=sin(i)
end
seno =
     0
seno =
    0.1109
seno =
    0.2204
seno =
    0.3272
seno =
    0.4300
seno =
    0.5274
seno =
    0.6184
seno =
    0.7017
seno =
    0.7764
seno =
    0.8415
```

Sin embargo, es importante recordar que la operación anterior tiene una expresión mucho más eficaz y económica en MATLAB:

```
» ind=linspace(0,1,10);
» seno=sin(ind);
» seno
seno =
```

```
Columns 1 through 7
     0    0.1109    0.2204    0.3272    0.4300    0.5274    0.6184
Columns 8 through 10
  0.7017    0.7764    0.8415
```

b) Lazos controlados por relaciones: utilizan la orden while y tienen la estructura siguiente:

```
while (relación)
(sentencias,
…)
end
```

Todas las sentencias entre while y end se ejecutan repetidamente mientras la relación de control es verdadera.

```
num=0;epsilon=1;
» while (1+epsilon)>1
epsilon=epsilon/2;
num=num+1;
end
» epsilon
epsilon =
  1.1102e-016
» num
num =
    53
```

```
n=10;
while n>0
   y=1./n
   n=n-1;
end
```

c) Ramificaciones: utilizan la orden if y tienen la estructura siguiente:

```
if (relación)
(sentencias,
…)
end
```

También existen construcciones con varias alternativas:

```
if (relación)
   (sentencias que se ejecutan si relación es verdadera)
   else
   (sentencias que se ejecutan si relación es falsa)
end
```

```
for n=5:-1:-5
   if n~=0
   y=1./n
end
end
```

3.4 Construcción de m-files

En el transcurso de una sesión de trabajo, resulta conveniente a menudo agrupar las órdenes de MATLAB sobre ficheros de texto, que se denominan 'scripts' o ficheros de comandos. Una vez confeccionados, deben guardarse con la extensión *.m.

Aprovechando las facilidades de las técnicas de ventanas en el sistema Windows, es habitual mantener en funcionamiento simultáneo el editor de textos con los m-files de la sesión (cada uno en una copia diferente del editor) y la ventana d comandos de MATLAB, pasando de uno a otro ambiente en el momento oportuno.

Una vez guardado en el disco un m-file, se ejecuta desde la ventana de comandos de MATLAB invocando su nombre de la misma forma que una orden de MATLAB. Debe tenerse en cuenta el directorio actual y el directorio donde se guardan los m-files.

Los m-files son útiles, por ejemplo, cuando se desea crear estructuras de datos para manejar con comandos u otros m-files. También se pueden usar de almacén de datos, por ejemplo, matrices de tamaños grandes.

En los ficheros m conviene incluir líneas de comentario, que van precedidas del símbolo '%'. Las líneas de comentario de cabecera serán posteriormente devueltas por el comando de MATLAB help, cuando se hace seguir del nombre del fichero m.

Es posible hacer uso del portapapeles de Windows entre el editor de textos y la ventana de comandos de MATALAB, en ambos sentidos, con la precaución de borrar el 'prompt' << cuando se copian comandos de la ventana.

3.5 Construcción de funciones con m-files

Se puede extender MATLAB creando funciones nuevas para resolver problemas específicos, lo que se consigue de forma parecida a los 'scripts' de comandos, con ayuda del editor de textos.

En este caso, la primera línea es especial. Contiene la orden function seguida del vector de parámetros de salida, signo =, nombre de la función y, entre paréntesis y separados por comas, los argumentos de la función. Hay que tener en cuenta que ahora todas las variables son locales, excepto las correspondientes a los parámetros de salida.

Ejemplos:

```
%Generación de una señal
%Cálculo de su media y su desviación típica
%Parámetros de entrada : Tamaño de la señal
%Parámetros de salida  : Media y desviación típica
%8 de diciembre de 1998
%
function [media, dt]=stats(n)
x=linspace(0,1,n);
signal=sin(2*x).*sin(5*x);
media=mean(signal);
dt=std(signal);
```

Seguidamente veremos dos maneras de construir una función para calcular factorial de un número natural n, la segunda de las cuales ilustra el concepto de recursividad.

```
%Cálculo de factorial de n
%
%10 de enero de 1999
%
function f=factorial(n)
f=n;
if n==0|n==1
   f=1;
```

```
end
   while n>1
      f=f*(n-1);
      n=n-1;
   end
  %
%Cálculo de factorial de n recursivamente
%
%10 de enero de 1999
%
function f=fact(n)
if n>1
   f=n*fact(n-1);
else
   f=1;
end
```

Ejercicios:

1) Resolver la ecuación a*x^2+b*x+c=0 escribiendo una función con los argumentos a,b,c y obteniendo como parámetros de salida x1 y x2.

2) Dado un triángulo ABC, se conocen las medidas de sus lados b y c y el ángulo de vértice A. Escribir una función que determine la longitud de lado c. Agregar la determinación de la superficie del triángulo. Recuérdese, S=sqrt(s(s-a)(s-b)(s-c)), siendo s=(a+b+c)/2.

3) Dada la sucesión de término general 1+1/2^2+…+1/n^2, escribir una función que determine u(n1)-u(n2), dados n1 y n2.

4) Constrúyase una señal aleatoria con 1024 muestras. Dedúzcase una matriz cuadrada 1024x1024 con todas sus columnas iguales y con elementos respectivos iguales a la raíz cúbica de los elementos de la señal a) Con un lazo for sin inicializar; b) Con un lazo for inicializando con una matriz de ceros; c) Sin lazo for (como se debe hacer siempre en MATLAB). En cada caso, mídase la eficacia del código.

5) Obténgase una estimación de lím sin(x)/x cuando x→0.

6) Dada la sucesión de término general u(n)=1+2^2+…+n^2, hacer uso de un lazo while para determinar el mayor valor de n (por ejemplo n0) para el que se obtiene un término menor que 1000. Imprimir n0 y u(n0).

7) Obténgase una estimación de la solución de la ecuación *x-cos(x)=0* con un error menor que 0.001.

8) Escríbanse los primeros 100 elementos de la sucesión de Fibonacci.

9) Constrúyase un programa para determinar la posición en el plano (x,y), la velocidad y el ángulo de la trayectoria de un proyectil conociendo los datos siguientes:

velocidad inicial: 10 m/s;

ángulo de disparo 50 grados;

tiempo transcurrido desde el disparo 12 segundos.

Recuérdese: *x=v0.t.cos(alfa), y=v0.t.sin(alfa)-g.t^2/2.*

10) Haciendo uso del desarrollo en serie de McLaurin, estimar ln(1+x) para x=0.5 con un error menor que eps.

3.6 Comparación de la eficacia de programas

Podemos determinar la cantidad de operaciones de coma flotante que se realizan al ejecutar un comando, un conjunto de ellos o un 'script' de comandos con la orden flops, que se puede inicializar en el momento conveniente escribiendo flops(0).

Asimismo, el tiempo de ejecución se puede obtener invocando las órdenes de arranque y parada de reloj tic y toc.

Ejemplo:

```
%Inversa de la matriz de Hilbert de orden 6
%
tic
flops(0)
inversa=inv(hilb(6));
toc
flops
elapsed_time =
    0.1700
ans =
   795
```

La orden cputime devuelve el tiempo de ejecución de un programa.

3.7 Herramientas de depuración

El editor de textos asociado con MATLAB posee varias herramientas de depuración, como puntos de parada, ejecución paso a paso, revisión de valores intermedios de variables, etc.

3.8 Resumen

Comandos nuevos comentados en este Capítulo:

any
all
cputime
end
else
find
flops
for
if
mean
std
tic
toc
while