Tema 7: Programación con Matlab

1. Introducción

Matlab puede utilizarse como un lenguaje de programación que incluye todos los elementos necesarios. Añade la gran ventaja de poder incorporar a los programas propios del usuario todas las aplicaciones que ya tiene implementadas, lo cual facilita y simplifica en muchos casos la programación. También será de gran utilidad tener en cuenta la estructura vectorial y matricial del programa.

Como ya hemos adelantado, los programas en Matlab suelen escribirse en los ficheros .m (M-ficheros). Lo normal es que sea en ficheros Scrips que resultan los más sencillos. A veces, no tienen argumentos de entrada ni salida y están formados por un conjunto de instrucciones que se ejecutan secuencialmente. Por ejemplo, el fichero de la figura 26 representa una curva cuando se escribe su nombre, en este caso “pinta”, en la línea de comandos y se pulsa intro.

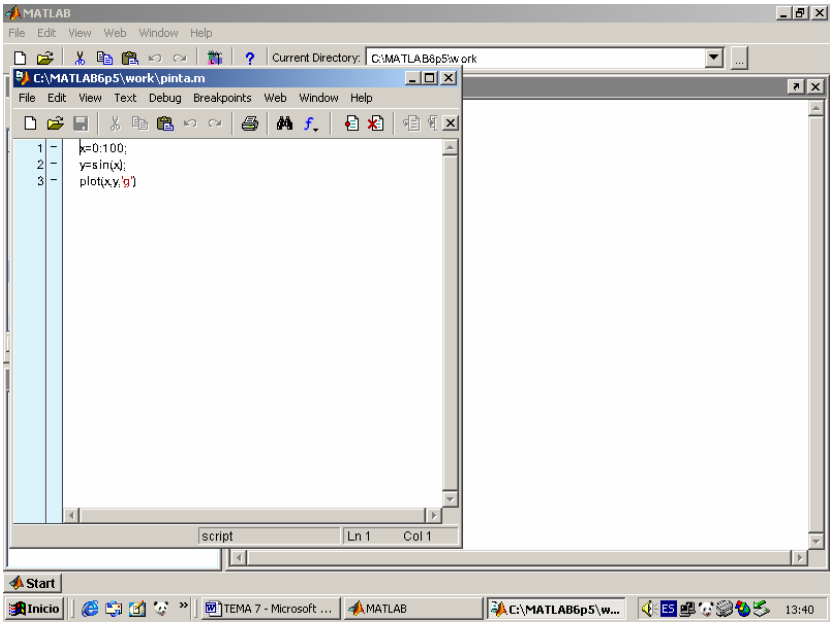

Figura 26

Se puede decir que este ya es un programa creado en Matlab.

La estructura general de un programa MATLAB es la siguiente:

1) Comentarios: inicialmente, pueden aparecer líneas comentadas en las que se da un título al programa y se realiza una breve descripción del mismo. Esta parte es opcional, pero es útil introducirla ya que se nos permite acceder directamente desde la ventana de comandos a la información comentada mediante la utilización del comando help, en la forma:

>> help nombre del programa

2) Entrada de datos si se requiere: los datos necesarios para la resolución del problema deben suministrarse al programa mediante la lectura de sus valores por teclado o desde un fichero de datos.

3) Algoritmo: desarrollo de un procedimiento que permite obtener la solución del problema en función de los datos de entrada.

4) Salida de datos: los datos obtenidos como solución del algoritmo se deben ofrecer al usuario mediante escritura en pantalla o en un fichero de datos.

2. Entrada y salida de datos

Existe un comando para introducir información en un programa cuando estamos en modo de ejecución. Este comando es: v**=input**(‘Cadena de Caracteres’)

input realiza dos tareas:

1) Imprime en pantalla la cadena de caracteres que lleva como argumento.

2) Los datos que el usuario teclea en respuesta al letrero, los introduce en la variable v.

Para que un programa en modo de ejecución pueda escribir letreros, avisos, etc. por pantalla, se utiliza el comando: **disp**(‘Cadena de Caracteres’) que escribe la cadena de caracteres que tiene como argumento en pantalla.

Para escribir el valor de una variable, se utiliza el comando: **disp**(v) que muestra en pantalla el valor de la variable v.

Para escritura de texto y/o datos en pantalla, se puede utilizar la función: **sprintf**(‘formato’, variables).

Ejemplos:

>> n=input(‘teclea el número de elementos’)

>>disp(‘este valor no es adecuado’)

3. Operadores

Ya hemos estudiado operadores de tipo aritmético. En este momento puede ser de utilidad conocer otros tipos de operadores:

Operadores relacionales:

< Menor

<= Menor o igual

>= Mayor o igual

> Mayor

== Igualdad

~= Desigualdad

find(A) Devuelve los índices de los elementos no nulos

find(A condición) Devuelve los índices de los elementos de A que cumplen la condición

Operadores lógicos

~A Negación lógica

A & B Conjunción lógica (and)

A | B Disyunción lógica (or)

xor(A,B) or exclusivo, vale 1 si A o B, pero no ambos, valen 1

Todos estos operadores actúan elemento a elemento en matrices y vectores. Las dimensiones y número de elementos de las tablas deben coincidir.

Ejemplo:

>> A=1:9; P=(A>2)&(A<6)

P= 0 0 1 1 1 0 0 0 0

4. Sentencias de control

El uso de aplicaciones recursivas y condicionales es muy habitual en matemáticas. Para ello si utilizan las bifurcaciones y los bucles.

Las bifurcaciones permiten realizar una u otra operación según se cumplan o no ciertas condiciones. Los bucles repiten operaciones sobre datos distintos.

Algunas de las sentencias de las que dispone Matlab para este tipo de trabajos son las siguientes:

Sentencia for:

Permite ejecutar de forma repetitiva un comando o grupo de comandos. La forma general de un bucle for es:

for variable=expresión

comandos

end

Por ejemplo:

```
for i=1:3;v(i)=1;end;v
    1     1     1
>>
```

Así, un bucle for siempre empieza por la sentencia for y termina con la end. En su interior incluye todo un conjunto de comandos que se separan por comas. En algunos casos es bueno poner puntos y comas para evitar repeticiones en las salidas. Por supuesto puede utilizarse en ficheros .m (figura 27).

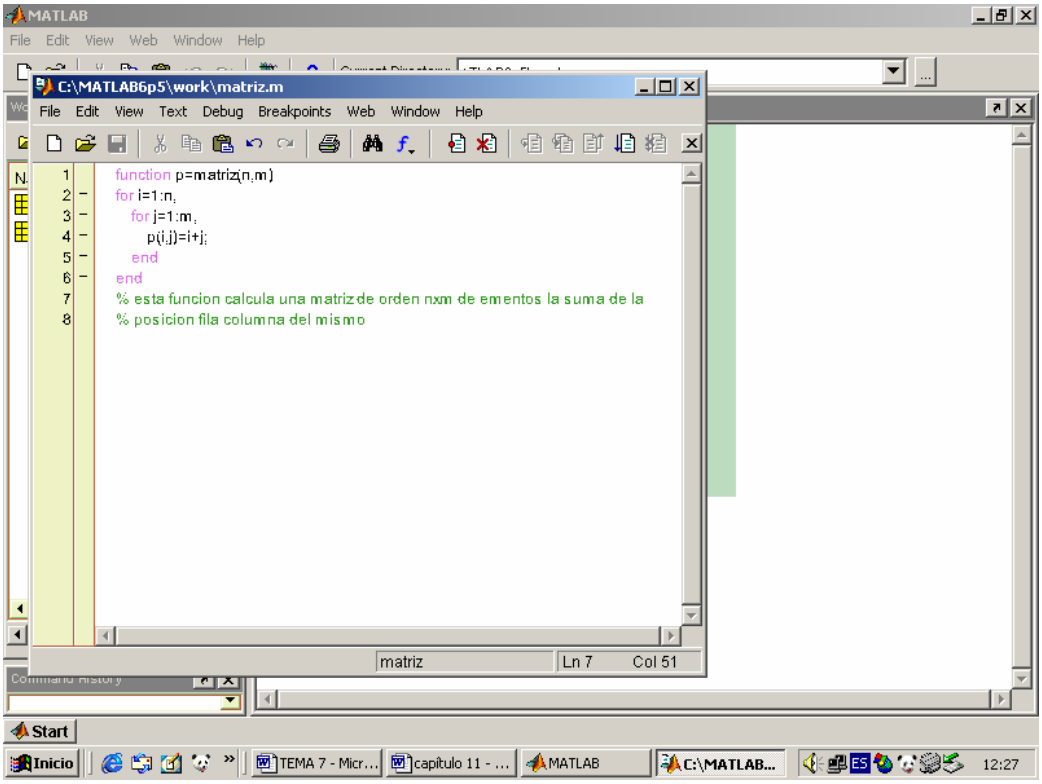

Figura 27

Sentencia if:

Mediante esta estructura se pueden ejecutar secuencias de comandos si se cumplen determinadas condiciones. Su sintaxis es:

If condición

comandos

end

De forma más general:

If condición

comandos 1

else

comandos 2

end

que ejecuta comandos 1 si la condición 1 es cierta y comandos 2 si es falsa.

Como en el caso de for, se pueden anidar sentencias if:

If condición 1

comandos 1

elseif condición 2

comandos 2

elseif condición 3

comandos 3

...

else

end

Mediante el siguiente ejemplo se imprime en pantalla una frase, de tres posibles, según sea el valor de la variable n:

```
N=input('introduce un número natural')
  If n=0,
  disp('n es cero')
  elseif rem(n,2)==0
  disp('n es par')
  else
  disp ('n es impar')
  end
```

Sentencia while:

También dispone de la sentencia “haz mientras” que ejecuta un bucle mientras una condición sea cierta. Su sintaxis es:

while condición

comandos

end

en el interior (comandos) se incluyen todo tipo de comandos que se separan por comas y que se ejecutan mientras la condición sea cierta.

Como ejemplo calcularemos el mayor número factorial que no esceda a 10^{100}:

```
n=1;
while prod(1:n)<1.e100,
        n=n+1;
end,
n
```

Este otro ejemplo tiene como salida el vector 1 2 3 4 5 6.

```
v=1:9;
 i=1;
while v(i)<7
disp(v(i))
 i=i+1;
 end
```

Continue

Esta sentencia hace que se pase inmediatamente a la siguiente iteración del bucle for o while, saltándose todas las sentencias que existan entre el continue y el final del bucle en esa iteración.

Break

Hace que se termine la ejecución de un bucle for o while.

Nota: existen otras sentencias que pueden ser de interés y que pueden consultarse en: matlab\lang - Programming language constructs.

Práctica 7: Programación con Matlab

1. Se pide:

a) Definir la función $f(x)=\begin{cases} x^2-1 & si \quad -2 \le x \le 2 \\ \frac{1}{|x|} & si \quad x<-2 \quad ó \quad x>2 \end{cases}$

b) Diseñar un programa que obtenga f(x) si x es un escalar, pero si es un intervalo (vector de dos componentes) dibuje la función en ese intervalo; en caso contrario debe aparecer un aviso en pantalla.

c) Realizar la llamada al programa para:
   1: Evaluar la función en los valores 1 y 8.
   2: Obtener la gráfica de la función en el intervalo [-15,15].

d) Arreglar el apartado a) para que f se pueda aplicar sobre vectores y nos de las salidas de la aplicación de la función sobre cada componente.

2. Construir un programa que calcule los cubos de los números naturales cuyo cuadrado sea menor que un número m (que se pide al usuario) y los introduzca en un vector v.

3. Construir una función r(a,b,c) que calcule las raíces de un ecuación de 2º grado ax^2+bx+c y que indique:
   a. "La ecuación no es de 2º grado" si a=0.
   b. Que existen 2 raíces reales distintas.
   c. Que existe una raíz real doble.
   d. Que existen dos raíces complejas.

4. Construir un programa que nos de los n primeros números de la sucesión $a(n)=n^2-4n+3$. Añadir una sentencia al programa que permita representar los términos en el plano.

5. Crear un programa que calcula la matriz A de elementos $a_{ij}=i^2+j$.

6. Dado un plano ax+by+c=0, crear un programa que nos diga si un punto (x,y) pertenece a dicho plano, está por encima o por debajo de él.

7. Crear un programa que detecte si en un determinado intervalo [a,b] existe una raíz de una función continua f(x).

8. En un comercio se venden cajas de tornillos. Su precio depende del número de cajas comprados: hasta 100 cajas el precio de la caja es de 2 euros, desde 101 a 200 cajas el precio es 1,5 euros y, a partir de 201 cajas el precio es de 1 euro. Elaborar un programa que pregunte el número de cajas demandadas y que indique el precio de la unidad y el coste total del pedido.

9. Realiza una tabla que incorpore el valor de los números enteros menores que 20, sus inversos, sus cuadrados y sus raíces cuadradas. Hacerlo primeramente con el comando while y luego con el comando for.

10. Programar el método de la bisección y el de Newton para aproximar raíces de funciones.