Introducción a MATLAB

Matlab es un programa de cálculo creado especialmente para trabajar con matrices, aunque también sirve para muchos otros campos de las Matemáticas.

El nombre MatLab significa **Mat**rix **Lab**oratory = laboratorio de matrices. Fue desarrollado por primera vez en 1984, por la compañía **MathWorks.**

La versión de estudiante puede obtenerse gratuitamente en **www.mathworks.com** (existen versiones para Windows, Macintosh y Unix/Linux). En la Biblioteca de la ETSIIT encontrarás manuales de Matlab que vienen con CD de instalación, además de encontrarlo instalado en las salas de informática.

Matlab también funciona como lenguaje de programación, por lo que se pueden definir nuevas funciones, algoritmos, etc, aunque en este curso sólo utilizaremos las funciones que ya tiene implementadas.

ESCRIBIR INSTRUCCIONES

Cuando iniciamos Matlab aparece el *prompter*, es decir el símbolo >> , que indica que Matlab está listo para recibir instrucciones.

Cuando escribamos una **instrucción** y pulsemos la tecla **ENTER**, el programa ejecutará la instrucción y dará una resultado o salida (o si es el caso, un mensaje de error). El resultado o salida se visualiza a continuación de la instrucción.

Podemos evitar que Matlab nos muestre la salida (por ejemplo, si va a ser larga) terminando la instrucción con el símbolo **;** así la instrucción se ejecuta pero es "muda", no se muestra la salida.

Podemos poner dos o más instrucciones seguidas en la misma línea, separadas por una coma, y al pulsar ENTER se ejecutarán las dos, y se mostrarán los dos resultados. Si en lugar de coma las separamos con **;** entonces no se visualizará el resultado.

Suele ser útil escribir **comentarios**, que no se ejecutarán: se hace comenzando la línea por el signo **%.** De este modo podemos introducir texto que nos ayudará después a revisar lo que hemos hecho.

GRABAR EL TRABAJO REALIZADO

Puedes **grabar lo realizado durante la clase** de la siguiente manera: escribe la instrucción

```
>> diary nombre
```

donde *nombre* es el nombre que quieras darle. Todo lo que se haga a continuación se va grabando en tiempo real en un fichero con ese nombre.

Cuando termines escribe la instrucción `>> diary off`

y dejará de grabarse. El fichero suele guardarse por defecto en la carpeta Work dentro de la carpeta donde esté instalado Matlab. Si aparece como un fichero sin extensión, puedes darle la extensión .txt y abrirlo con cualquier procesador de textos.

OPERAR CON NÚMEROS

Podemos comenzar experimentando con números, usando Matlab como calculadora, con los operadores

+ - * / \ ^ ( )

Observar que las operaciones siguen las habituales reglas de precedencia: * tiene prioridad sobre +, etc.
Nota: los decimales se indican con punto, nunca con coma.

Tenemos tres formatos para operar: **decimal corto, decimal largo**, y **racional**.
Para operar en formato decimal corto (cuatro posiciones decimales) pondremos la instrucción

```
>> format short
```

Para operar en formato decimal largo (14 posiciones decimales) pondremos la instrucción

```
>> format long
```

Para operar en formato racional (con fracciones) pondremos la instrucción

```
>> format rational
```

Cada formato sigue funcionando mientras no activemos otro.

- **Nota**: Un truco útil de manejo consiste en **recuperar instrucciones** que hayamos escrito antes, para volver a usarlas o modificarlas, pulsando la tecla de "cursor arriba". Además, si escribimos las primeras letras de una instrucción y luego pulsamos "cursor arriba" encontraremos las instrucciones que comiencen con esas letras.

VARIABLES

Con una instrucción como esta,

```
>> a=5
```

estamos introduciendo el valor 5 en una variable llamada `a`. Desde este momento, `a` vale 5, y se puede operar con `a` como si fuese un número.

Notar que se admiten instrucciones del tipo

`>> a=a+2` donde `a` pasa a tener el mismo valor que tenía, mas 2.

Los nombres de las variables pueden contener **letras**, **números** y **guión bajo _** pero deben comenzar por letra. Es conveniente que los nombres indiquen algo sobre su contenido: por ejemplo podemos poner nombres como `matriz_inversa`, `solucion_sistema1`, etc. Notar que Matlab distingue entre **mayúsculas y minúsculas**.

Hay nombres de variables que están reservados, como `pi`, o la variable `ans` que siempre contiene el último resultado producido.

Si a una variable le damos un nuevo valor, se **sobreescribe** (el valor anterior desaparece). Por otra parte, para **borrar el valor de una variable** ponemos por ejemplo `>> clear a` que borraría el valor de a, o bien `>> clear` que borra el valor de todas las variables.

Para saber en cada momento qué variables tenemos definidas, se usa la instrucción `>> whos` .

FUNCIONES

Las funciones se escriben con minúsculas y van seguidas de un **argumento** , que siempre va **entre paréntesis**, al cual se aplica la función. Por ejemplo:

`>> sqrt(2)` para calcular la raíz cuadrada de 2.

La función `sqrt` efectúa la raíz cuadrada, y el argumento (número al que se aplica) en este caso es 2. El argumento también puede ser una variable que tenga un valor.

Otras funciones se podrán aplicar a vectores, a matrices, etc.

AYUDA

Si queremos **buscar en la ayuda** información sobre una función, por ejemplo `sqrt`, podemos poner

```
>> help sqrt
```

y aparecerá la información.

(**Nota**: en los textos de ayuda, las funciones aparecen escritas con mayúsculas, como SQRT, pero sólo es para hacerlas resaltar. Nosotros deberemos escribirlas con minúsculas).

Si no sabemos exactamente cómo se escribe una función, podemos buscar información con la instrucción `lookfor`, usando una palabra en inglés que describa lo que queremos, por ejemplo

```
>> lookfor square
```

- **Nota**: A veces el texto de ayuda es largo y tarda tiempo en salir. Se puede interrumpir el proceso en cualquier momento pulsando las teclas **Ctrl** + **C.** Estas teclas también sirven para interrumpir un cálculo muy largo o que se haya bloqueado.

DEFINIR MATRICES

En esta asignatura trabajaremos fundamentalmente con matrices.

Por ejemplo $A=\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ se escribe en Matlab como

`>> a=[1 2 3;4 5 6;7 8 9]` o bien `>> a=[1, 2, 3; 4, 5, 6; 7, 8, 9]`

Es decir, las matrices se introducen entre corchetes, por filas, separando los elementos de cada fila por un espacio en blanco o por coma. Cada fila se separa de la siguiente por **;**

- **Nota**: No hay que confundir los paréntesis redondos ( ) con los corchetes o paréntesis cuadrados [ ]. En este contexto, los corchetes prácticamente sólo se utilizarán para **definir matrices o vectores**. En todos los demás casos se usan paréntesis redondos.

ALGUNAS OPERACIONES CON MATRICES

Al igual que con los números, también funcionan con matrices los operadores + - * / \ ^ ( ) (siempre que las operaciones sean realizables).

La **matriz traspuesta** se hace con el signo de comilla simple, por ejemplo `>> a'` hace la traspuesta de `a`.

Por ejemplo, para introducir un vector columna tenemos dos opciones:

```
>> v=[1; 2; 3]
```

o bien introducirlo como fila y hacer la traspuesta:

```
>> v=[1 2 3]'
```

- **Nota**: No debe confundirse el signo de comilla simple con el acento. El signo de comilla simple, el que se usa para la matriz traspuesta, suele encontrarse en la misma tecla que la ?, en la fila superior del teclado.

Matlab permite **unir matrices** de dimensiones adecuadas, como si construyéramos una "matriz de matrices", con instrucciones como `>> [a b]` o `>> [a; b]` donde `a` y `b` son matrices.

También podemos **extraer elementos, filas o columnas** de una matriz. Por ejemplo,

`>> a(2,3)` es el elemento (2,3) de la matriz `a` (es decir, fila 2, columna 3)

`>> a(2:4,3)` son las filas 2 a 4, columna 3.

`>> a(2,3:5)` fila 2, columnas 3 a 5.

`>> a(2:4,3:5)` filas 2 a 4, columnas 3 a 5.

`>> a(:,3:5)` todas las filas, columnas 3 a 5.

`>> a(1:3,:)` filas 1 a 3, todas las columnas.

`>> a(:,:)` todas las filas y columnas, por tanto sería la misma matriz `a`.

También se pueden tomar filas o columnas salteadas, o cambiarlas de orden:

`>> a([1,3,5],[4,2])` filas 1, 3 y 5, columnas 4 y 2.

Algunas matrices están predefinidas en Matlab, por ejemplo la **matriz identidad** de tamaño n, que se construye con `eye(n)`, la matriz de unos de tamaño mxn con `ones(m,n)`, o la matriz de ceros de tamaño mxn con `zeros(m,n)`.

En adelante aprenderemos diversas funciones que se pueden aplicar a las matrices y vectores: `rank`, `det`, `lu`, `chol`, `null`, `dot`, `norm`, etc.