Tutorial con los comandos básicos de Matlab

```
%Cálculos simples
        cos(2+sqrt(3))
        exp(3/log(5))

%Ojo: hay ciertos problemas con la configuración del teclado español en
%Matlab. A veces hay teclas que no reconoce. Para evitarlo basta con abrir un
%editor cualquiera (Bloc de notas, Word), teclear el comando y copiarlo en
%Matlab.
%Por ejemplo, la potencia se escribe con el ángulo ^ que está sobre la tilde
%francesa pero a mí no me funciona en el ordenador, así que lo copio y lo
%pego del Bloc de notas
        32^4

%Como vemos, el último valor calculado lo almacena en la variable ans
        cos(ans)

%CREACIÓN DE VARIABLES:
        x=1.3
        y=-3*x+sqrt(x)

%Si se pone ; al final se ejecuta pero no se ve
        y=4*x-5*log(x);  %se machaca el valor anterior
        y

%se pueden usar para hacer recursiones
        y=2*y

%Importante: Matlab distingue mayúsculas y minúsculas, es decir, x no es lo
%mismo que X
        Y=3*x

%Para limpiar todas las variables (es como si volviesemos a empezar)
        clear       %Cuidado! a veces no interesa limpiarlas todas

%para resolver una duda con la ayuda online
        help sqrt

%Matlab SIEMPRE TRABAJA EN DOBLE PRECISIÓN, es decir, con 15-16
%cifras decimales exactas. Otra cosa es como lo muestra en pantalla.
%Por defecto muestra solo 4 cifras decimales. Para que muestre las 16 cifras
%se pone
        format long
        x=1.3
        y=-3*x+sqrt(x)

%Para continuar un comando en la línea inferior hay que poner ...
        s=1+2+3+4+5+...
                +6+7
```

```
%ojo: no vale para cadenas de caracteres: hay que cerrarlas en la misma
%línea.

%Para recuperar órdenes hay que usar las flechas arriba/abajo. Así podemos
%hacer recursivamente
        x=0.2
        x=cos(x)
        x=cos(x)      %....(Repetir varias veces con las flechas)


%VECTORES Y MATRICES

%Para introducir vectores se puede hacer componente a componente
        v(1)=0, v(2)=1, v(3)=4, v(4)=5

%o en conjunto entre [] y separados por comas o espacios en blanco
        v=[0,1,4,5]
        v=[0 1 4 5]

%Comando ':'  A:INC:B  varía de A a B con incremento INC.
%Se usa para definir vectores, pero también bucles como veremos más tarde
        v=[0:0.1:1]             %Va de 0 a 1 con incremento 0.1.
%Puede ser negativo
        w=[6:-0.2:4]

        z=[0:0.3:1]

%Se pueden hacer operaciones entre componentes
        w(1)*w(2)*z(4)

%Importante: en Matlab no existe la componente 0-ésima. Con esto tendremos
%algún problema cuando intentemos pasar las formulas de interpolación.
%Habrá que reajustar los subíndices en las fórmulas.

%Las matrices se pueden introducir por filas separadas por ;
        A=[1,2,-3; 3,4,5; 6,7,8]
%o dándole al Enter después de cada fila
        A=[1,2,-3
          3,4,5
          6,7,8]
%Y también componente a componente
        A(1,1)=1, A(1,2)=2, A(1,3)=-3       %,....

%Operaciones con vectores y matrices
%La suma (resta) y producto siguen las reglas del álgebra lineal
        B=[1,4,-5; 0,3,6;-1,-2,9]
        z=[-3,4,5]
        A+B
        A*B
        A*z
```

```
        A*w  %da error pues las dimensiones no cuadran

%Traspuesta: apostrofe '
        C=A'
        w=v'

%Operaciones componente a componente:
%Las operaciones básicas se pueden hacer poniendo un . delante
        A.*B
        B./A
%Las funciones intrínsecas lo hacen componente a componente siempre, y
%también escalar*matriz
        D=cos(A)
        3*D

%Otras funciones matriciales
        d=det(A)      %determinante de A
        E=inv(A)      %matriz inversa numérica
        F=diag([1,2,3])     %matriz diagonal donde [1,2,3] es la diagonal principal
        I1=zeros(3)     %matriz nula de dimensión 3
        I2=eye(4)       %matriz identidad de dimensión 4

%Resolución de sistemas lineales Ax=r: lo hace mediante
%descomposiciones LU, ahorrando operaciones en el caso de matrices
%especiales como las diagonales simétricas
        r=[5,6,-4]'       %Ojo: hay que trasponer pues es un vector columna
        x=A\r


%GRÁFICAS

%Matlab dibuja puntos (x,y) que tenemos que darle y luego los une con
%líneas
        clear
        x=[1,2,3,4], y=[3,6,7,0]
        plot(x,y)

%Si queremos dibujar una función tenemos que darle muchos puntos:
        x=[0:0.1:10];         %ponemos ; para que no salga toda la lista
        y=cos(x)*sin(x);
        plot(x,y)

%También esta la alternativa de linspace para x (para graficas suele ser
%más cómoda)
        x=linspace(0,10,100);      %crea 100 puntos de 0 a 10
        y1=cos(x).*sin(x);  %se pone el punto pues es producto matricial
        y2=cos(3*x)/4;
        plot(x,y1,x,y2) %mezcla de dos o mas curvas
```

%Opciones:
plot(x,y1,'r',x,y2,'mo'), title('Funciones'), text(5,0.3,'Puntos'), xlabel('x'), ylabel('y=f(x)'), axis('equal'), legend('Funcion 1','Funcion 2')

%Esta línea dibuja la función y1 en línea continua roja, la y2 en línea
%punteada de color magenta ('m') con el símbolo 'o'. Le pone el título
%'Funciones', en el punto (5,0.3) pone la etiqueta 'Puntos', en el eje de
%las x pone la etiqueta 'x', en el eje de las y la etiqueta 'y=f(x)', y la
%orden axis('equal') hace que la escala en el eje x y en el y sean iguales,
%o sea, una escala 1:1. Además pone una leyenda en el borde superior
%izquierdo, identificando cada curva. Para opciones, usar help plot, help title,...

%Hay una opción para dibujar funciones (fplot), que es ligeramente peor que
%esta y más costosa. No la veremos por ahora.

%Para imprimir las gráficas, se hace desde la ventana de graficas. Se
%exporta en la opción **File->Export**. Se puede pasar a los siguientes formatos:
% -PostScript (encapsulado): .eps (recomendado, es el mejor)
% -Bitmap .bmp
% -JPEG .jpg
% -TIFF .tif