Laboratorio 0 MA-43-B:

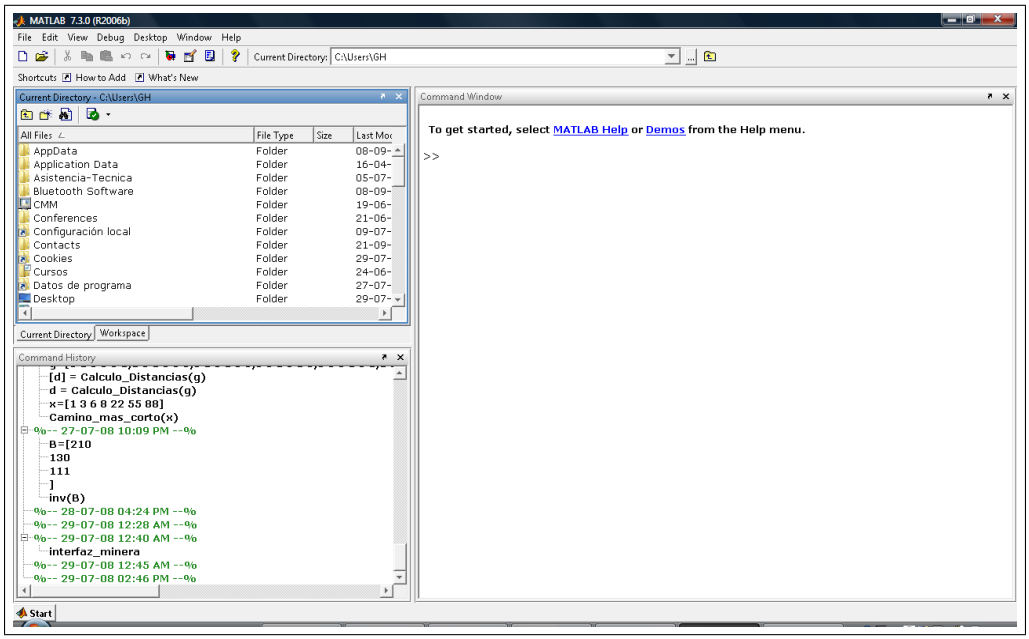
Comandos Básicos de Matlab

Gonzalo Hernández

UCHILE - Departamento de Ingeniería Matemática

1 Interfaz de Matlab

Al ejecutar el programa Matlab, se abrirá una ventana con la siguiente interfaz gráfica:



Esta es la distribución de ventanas por default. Expliquemos cada una de las subventanas:

1.1 Command

La ventana llamada Command es la ventana principal de Matlab. En esta ventana es donde se definen las variables, se escriben pequeños programas,se ejecutan funciones y se muestran los resultados.

1.2 Current Directory

Muestra la dirección del directorio de trabajo, es decir, donde se guardan las funciones programadas para luego ocuparlas. Por defecto, la dirección del directorio es "C:\Matlab\work", pero se puede cambiar a cualquier otra especificando la dirección. Aquí aparecerá una lista de las funciones que hayamos guardado en esa carpeta.

1.3 Command History

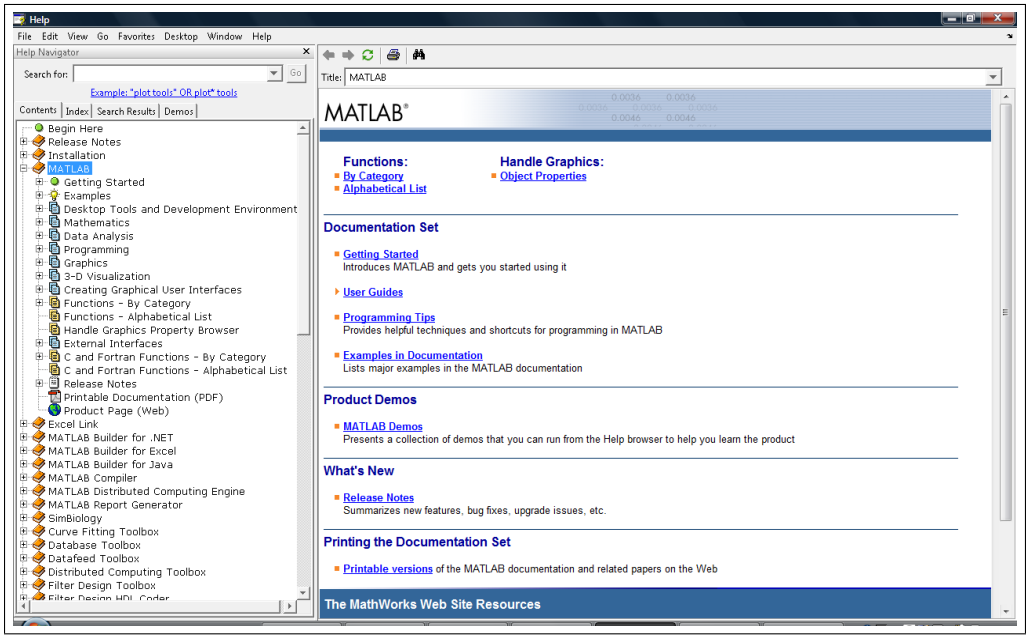
En esta ventana se registran todos los comandos introducidos en la ventana Command Windows en esta sesión y las sesiones anteriores. Para limpiar el historial, basta hacer click con el boton derecho del mouse y elegir "Clear Entire History".

1.4 Workspace

Aparece un lista con el valor y la información de todas las variables definidas. El comando **clear** borra todas las variables que se han creado.

1.5 Help

La ventana Help contiene información de ayuda sobre Matlab y sus funciones. Cualquier duda que se tenga, basta apretar la tecla **F1** y aparecerá la ventana Help.



ACT1: Operaciones básicas en la interfaz Matlab:

i) Cambie el "Current Directory" al directorio donde trabajará en esta oportunidad.

ii) Cambie la distribución "Default" de ventanas a una de su agrado. Guarde esta distribución para uso futuro. Active la ventana "Editor".

iii) Busque en el Help alguna información que necesite. Por ejemplo, más adelante en este laboratorio necesitaremos graficar una función. Esto se hace mediante el comando "plot".

2 Trabajando en Command

1) Luego de escribir un comando, al apretar la tecla **Enter** este se ejecuta y se muestra en pantalla todas las salidas que generó.

2) Si al final de escribir un comando se termina con el **punto y coma** (**;**), el comando se ejecuta pero no se muestra en pantalla ninguna salida.

```
>> cos(pi)

ans =

 -1

>> cos(pi);
```

3) Si al comienzo de una linea se escribe el simbolo **porcentaje** (%) , la linea es tomada como un comentario y no se ejecuta.

4) El comando **clc** limpia todos los comandos en la ventana Command, pero las variables predefinidas no son modificadas.

3 Operaciones Aritméticas

Operación	**Simbolo**	**Ejemplo**
Suma	+	5+3=8
Resta	-	5-3=2
Multiplicacion	*	5*3=15
División por la derecha	/	5/3
División por la izquierda	\	5\3=3/5
Exponenciación	^	5^3=$5^3 = 125$

3.1 Orden de precedencia

Al evaluar una expresión con varios tipos de operaciones aritméticas, unas tiene mayor prioridad que otras:

Precedencia	**Operacion**
Primera	Paréntesis **( )**
Segunda	Exponenciación **^**
Tercera	Multiplicación ***** y división **/**
Cuarta	Suma **+** y Resta **-**

4 Formatos Numéricos

En Matlab se puede cambiar el formato numérico de las expresiones. Por defecto, Matlab tiene *format short*, que se puede cambiar por otro, como por ejemplo:

Comando	**Descripción**	**Ejemplo:** >>290/7
format short	Reales de 4 decimales con redondeo	41.4286
format long	Reales de 14 decimales con redondeo	41.42857142857143
format short e	Notación científica de 4 decimales	4.1429e+001
format long e	Notación científica de 15 decimales	4.142857142857143e+001
format short g	Punto flotante de 5 digitos con redondeo	41.429
format long g	Punto flotante de 15 digitos con redondeo	41.4285714285714
format bank	Reales de 2 decimales con redondeo.	41.43
format rat	Cuociente de enteros pequeños	290/7

5 Variables

Matlab trabaja principalmente con variables para poder hacer cálculos y guardar resultados. Veamos como trabajar:

1) Toda variable por default es un número real floating point. Es posible también declarar variables enteras o lógicas.

2) Las variables no es necesario declararlas, solo es necesario asignar un valor, llamado expresión.

3) Una expresión puede ser simplemente un número o una fórmula aritmética usando números y otras variables previamente asignadas, por ejemplo:

$$2\hat{}3 + 46 + (5 * 10) + x$$

4) Para asignar una variables se usa el signo igual (=) de la siguiente forma:

$$\begin{aligned} Nombre\_variable &= Expresion \\ x &= \cos(pi) \end{aligned}$$

5.1 Reglas sobre los nombres de las variables

1) Los nombres de las variables pueden tener hasta 63 carácteres alfanuméricos incluyendo la barra abajo (_), pero siempre deben empezar con una letra. Matlab es sensible, es decir, las minúsculas son diferentes a las mayúsculas.

2) Si se usa como nombre de variable el nombre de una función predefinida, como cos, sin, exp, entre otros, las funciones predefinidas no funcionarán, pero si las variables. Evitar usar estos nombres para evitar confusión al programar.

3) Existen variables predefinidas, que se pueden redefinir en algunos casos especiales. Algunas de estas variables son:

   (a) pi : El número π
   (b) eps : La diferencia más pequeña entre dos números: 2^{-52}
   (c) inf : Usado para representar el infinito
   (d) i : La unidad imaginaria: $\sqrt{-1}$
   (e) j : La unidad imaginaria: $\sqrt{-1}$
   (f) NaN : Representa el resultado de una operación no válida, como por ejemplo $0/0$, $inf * 0$. Significa "Not a number".

5.2 Manejo de Variables

1) Como se mencionó, en la pantalla Workspace aparece una lista de todas las variables definidas y su información.

2) Para liberar memoria usada por Matlab, se puede eliminar variables definidas con algunos comandos básicos. Algunos de éstos son:

   (a) **clear** : Remueve todas las variables de la memoria
   (b) **clear x y z** : Remueve solo las variables x, y y z de la memoria

3) Para obtener información sobre las variables, pueden ser de gran utilidad los comandos:

   (a) **who** : Muestra una lista con todas las variables en la memoria

   (b) **whos** : Muestra una lista de todas las variables en la memoria con la información sobre sus bytes y clases.

ACT2: La ecuación cúbica:

$$ax^3 + bx^2 + c + d = 0 \qquad (*)$$

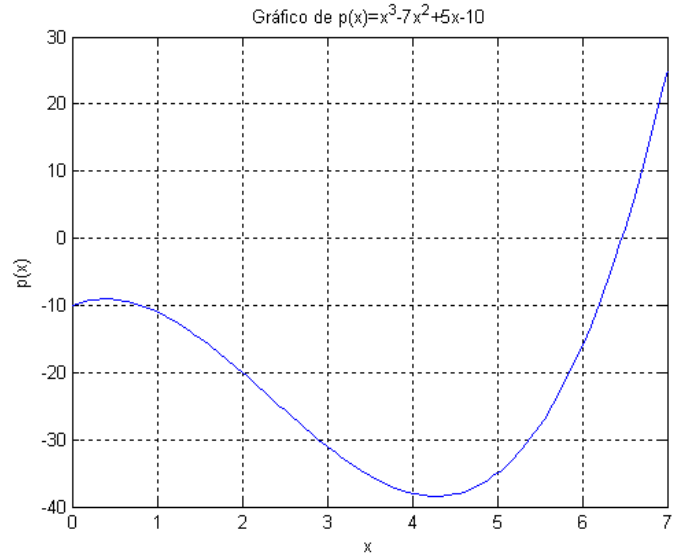
tiene 3 raíces x_1, x_2, x_3.

$$ax^3 + bx^2 + cx + d = a(x - x_1)(x - x_2)(x - x_3) = 0$$

Una de las cuales es real. Por ejemplo, si definimos la función cuadrática:

$$p(x) = x^3 - 7x^2 + 5x - 10$$

Su gráfico es:



Y sus raíces están dadas por: $x_1 = 6.4659, x_2 = 0.2670 + 1.2146i, x_3 = 0.2670 - 1.2146i$

Estas raíces se calculan de la siguiente forma:

$$\begin{aligned}
q &= \frac{9abc - 27a^2d - 2b^3}{54a^3} \\
r &= \sqrt{\left(\frac{3ac - b^2}{9a^2}\right)^3 + q^2} \\
s &= (q + r)^{\frac{1}{3}} \qquad t = (q - r)^{\frac{1}{3}} \\
x_1 &= s + t - \frac{b}{3a} \\
x_2 &= -\frac{(s+t)}{2} - \frac{b}{3a} + \frac{\sqrt{3}}{2}(s - t)i \\
x_3 &= -\frac{(s+t)}{2} - \frac{b}{3a} - \frac{\sqrt{3}}{2}(s - t)i
\end{aligned}$$

para $r \in \mathbb{R}$.

Defina 4 variables asignándole valores a elección. Suponga que estas variables son los parámetros a, b, c, d que definen la ecuación cúbica $(*)$. Calcule las raíces x_1, x_2, x_3.

Hint: El comando $sqrt(x)$ calcula raíz cuadrada del número x. Busque en el Help como se calculan raíces cúbicas.

6 Lógica

A veces es necesario usar lógica para algunos programas, como veremos más adelante, por lo cual necesitamos saber si una sentencia es verdadera o falsa.

6.1 Representación del True y False

Matlab no tiene las variables True y False. En vez de eso, los representa con números:

True	1 ó distinto de 0
False	0

Obs: Cuando Matlab entrega el valor True, siempre entrega el valor 1, pero reconoce cualquier número distinto de 0 como True.

6.2 Operadores Relacionales

Los operadores relacionales en Matlab son:

Operador	Descripción
$<$	Menor
$>$	Mayor
$<=$	Menor o Igual
$>=$	Mayor o Igual
$==$	Igual
$\tilde{}=$	Distinto

Estos operadores se pueden aplicar a dos números o a dos vectores de igual dimensión, que compara elemento por elemento. Ejemplo:

```
>> x=[0 1 2 3 4]

x =

     0     1     2     3     4

>> y=[4 3 2 1 0]

y =

     4     3     2     1     0

>> z=(x<=y)

z =

     1     1     1     0     0
```

7 Funciones matemáticas

En Matlab están predefinidas una gran variedad de funciones matemáticas. Algunas de ellas son:

Función	Descripción	Ejemplo
sqrt(x)	Raiz cuadrada	>> sqrt(81) ans = 9
exp(x)	Exponencial	>> exp(5) ans = 148.4132
abs(x)	Módulo absoluto	>> abs(-5) ans = 5
log(x)	Logaritmo natural	>> log(1000) ans = 6.9078
log10(x)	Logaritmo decimal	>> log10(1000) ans = 3
log2(x)	Logaritmo base 2	>> log2(512) ans = 9
factorial(x)	Factorial	>> factorial(10) ans = 3628800

Función	Descripción	Ejemplo
sin(x)	Seno	>> sin(pi/6) ans = 0.5000
cos(x)	Coseno	>> cos(pi) ans = -1
tan(x)	Tangente	>> tan(pi/6) ans = 0.5774
atan(x)	Arco-tangente	>> atan(1) ans = 0.7854
sinh(x)	Seno hiperbólico	>> sinh(0) ans = 0
cosh(x)	Coseno hiperbólico	>> cosh(0) ans = 1
tanh(x)	Tangente hiperbólica	>> tanh(1) ans = 0.7616

7.1 Funciones evaluadas en un intervalo

En muchas ocasiones es necesario evaluar una misma función en muchos puntos. Esto se puede hacer en Matlab definiendo el argumento x como un vector fila o columna. Veamos un ejemplo:

```
>> x=[1 2 3 4 5 6 7 8 9]

x =

     1     2     3     4     5     6     7     8     9

>> sqrt(x)

ans =

    1.0000    1.4142    1.7321    2.0000    2.2361    2.4495    2.6458    2.8284    3.0000
```

Tambien se pueden evaluar las funciones típicas en matrices.

ACT3: Busque en el Help de Matlab que otras funciones se pueden evaluar en matrices. Además de las funciones matemáticas básicas: ¿Qué otras funciones más avanzadas están implementadas?

7.2 Paso dado

En muchas ocasiones necesitaremos que un vector comience en un determinado número, y en cada posición del vector sea igual a la posición anterior mas una constante llamada paso. En Matlab se pueden crear estos vectores de la siguiente forma:

$$nombre\_variable = [m : q : n]$$

m : El primer término del vector. q : El paso entre dos términos. n : El último término del vector.

Observaciones

i) Los corchetes son opcionales.

ii) El paso q puede ser un real positivo o negativo.

iii) Si el paso q es omitido, por defecto vale 1, es decir $[m : n] \Longleftrightarrow [m : 1 : n]$

iv) Si $m = n$, entonces el paso no es coinciderado y guarda en la variable la constante m.

v) Si los valores de m, q, n no calzan, entonces:

(a) Si q es positivo, el último término del vector es el mayor número menor que n.

(b) Si q es negativo, el último término del vector es el menor número mayor que n.

Ejemplos

```
>> x=[0:-3:-10]          >> y=[0:3:10]          >> z=[0:0.25:1]
x =                      y =                    z =
0    -3    -6    -9      0 3 6 9                0 0.2500 0.5000 0.7500 1.000
```

7.3 Numero de puntos dados

En otros casos necesitaremos un vector de tamaño fijo, donde el primer y último término sean los extremos de un intervalo, y los otros términos sean puntos intermedios equiespaceados. Para esto, en Matlab existe una funcion predefinida:

$$nombre\_variable = linspace(xi, xf, n)$$

xi : Primer término del vector xf : Último término del vector n : Dimensión del vector

Observaciones

i) n debe ser un número natural.

ii) Si n es omitido, por defecto se concidera $n = 100$.

iii) Si xi es menor que xf, entonces el vector es creciente.

iv) Si xi es mayor que xf, entonces el vector es decreciente.

Ejemplos

```
>> x=linspace(0,8,6)

x =

         0    1.6000    3.2000    4.8000    6.4000    8.0000
```

```
>> x=linspace(-1,-10,7)

x =

    -1.0000   -2.5000   -4.0000   -5.5000   -7.0000   -8.5000  -10.0000
```

ACT4: Para calcular númericamente la derivada de una función clase $\mathcal{C}^1$ en x_0 se puede ocupar la fórmula de los 3 puntos :

$$\frac{df}{dx}(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{h^2}{6}\frac{d^3f}{dx^3}(\eta) \qquad \eta \in [x_0-h, x_0+h] \qquad \text{(3 Puntos)}$$

Si la función presenta una alta variabilidad (cambios de curvatura muy frecuentes), es recomendable utilizar una fórmula más precisa, como por ejemplo la de los 5 puntos:

$$\frac{df}{dx}(x_0) = \frac{f(x_0-2h) - 8f(x_0-h) + 8f(x_0+h) - f(x_0+2h)}{12h} + \frac{h^4}{30}\frac{d^5f}{dx^5}(\eta) \qquad \eta \in [x_0-2h, x_0+2h]$$

(5 Puntos)

Estas aproximaciones se obtienen al derivar el polinomio de Lagrange que interpola $f(x)$ en los puntos x_0-h, x_0, x_0+h y $x_0-2h, x_0-h, x_0, x_0+h, x_0+2h$ respectivamente.

Utilizando la fórmula de los 3 puntos, calcule la derivada numérica de la función:

$$f(x) = x\cos(x^2)$$

en el intervalo $[0, 6]$. Para esto defina:

i) Un vector x_0 equiespaciado a $q = 0.05$ en $[0, 6]$

ii) Dos valores para h cercanos 0

Calcule el error de esta aproximación para cada h comparando con el valor exacto de la derivada.

8 Graficar Funciones

En esta sección aprenderemos a graficar funciones en Matlab con algunas especificaciones: color y tipo de linea y marcas de los puntos.

8.1 Especificación del gráfico

Al comando plot se le puede agregar otros input:

$$plot(x, y, 'especificacion\ de\ linea', 'propiedad', 'valor')$$

Se puede elegir el tipo de linea, color de la linea y tipo de marcas de los puntos:

Tipo de linea	Especificación		Color de linea			Tipo de marcas	
Solida (Default)	- (Linea)		Roja	r		Ninguna (Default)	(Nada)
Segmentada	- - (Dos lineas)		Verde	g		Signo más	+
Punteada	: (Dos puntos)		Azul (Default)	b		Circulos	o
Segmento Punto	-. (Linea y punto)		Calipso	c		Asteriscos	*
			Magenta	m		Puntos	.
			Amarilla	y		Cuadrados	s
			Negra	k		Diamantes	d
			Blanca	w		Estrella de 5 puntas	p

Observaciones

i) Se elige hasta 1 tipo de linea, 1 color de linea y 1 tipo de marcas, y se ponen las 3 representaciones juntas,sin separaciones, en $'especificacion\ de\ linea'$.Ejemplo $' : r*'$

ii) Si alguna de las propiedades no se elige, entonces se graficará con la propiedad Default.

iii) El orden de las propiedades no importa.

iv) Para evitar ambiguedades como $' - .'$ que puede ser linea solida y marcar los puntos con puntos, o la linea segmento-punto, entonces es mejor siempre especificar las 3 propiedades.

8.1.1 Propiedades y valores

Aparte de las especificaciones ya mencionadas, también existen otras propiedades, que se deben poner aparte de la $'especificacion\ de\ linea'$. Se debe indicar la $'propiedad'$ y su respectivo $'valor'$. Se pueden poner cuantas uno quiera.

Propiedad	Descripción	Valores posibles
linewidth	Ancho de linea	Un número en unidades de puntos (Default 0.5)
markersize	Tamaño de las marcas	Un número en unidades de puntos
markeredgecolor	Color del borde de las marcas	Un color de la tabla 'Color de linea'
markerfacecolor	Color del relleno de las marcas	Un color de la tabla 'Color de linea'

8.1.2 Titulo y Etiquetas

Al crear un gráfico, se puede escribir un texto como título o un detalle en las coordenadas, con los comandos siguientes:

Comando	Descripción
title($'$texto$'$)	Escribe texto arriba del gráfico
xlabel($'$texto$'$)	Escribe texto en el eje x
ylabel($'$texto$'$)	Escribe texto en el eje y

ACT5: Grafique la función de la Actividad 4 en el intervalo [0,6] usando 60 puntos, con linea sólida de ancho 0.5, de color azul, marcas del tipo círculo. Ponga algún título en el gráfico. Guarde el gráfico como imagen jpg, bmp y archivo tipo .fig. Abra el archivo tipo .fig y cambie alguno de los parámetros del gráfico: línea sólida, color de línea, tipo de marcas. Vuelva a grabar el gráfico.

8.2 Gráficos Múltiples

Para poder gráficar 2 o más funciones en un mismo gráfico, se utiliza el mismo comando plot agregando nuevos vectores después de las especificaciones del primer gráfico:

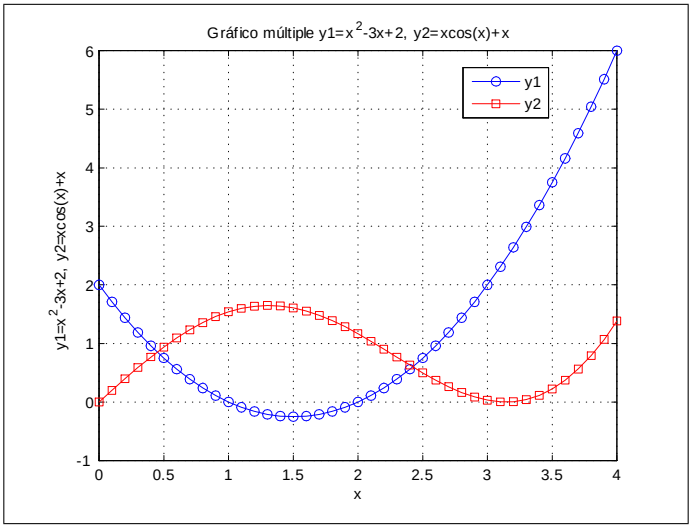
$$plot(x1, y1,'\ especificacion\ de\ linea1','\ propiedad1','\ valor1', x2, y2,'\ especificacion\ de\ linea2','\ propiedad2','\ valor2')$$

Otra forma de graficar 2 o más funciones de un mismo gráfico es con el siguiente comando:

$$hold\ on$$

Al graficar una funcion, y luego graficar la otra, el primer gráfico se borra y solo queda el último. Para evitar esto, antes de graficar la segunda función, se debe escribir el comando *hold on*, que sobrepone ambos gráficos. Por ejemplo:

```
>> x=[0:0.1:4]; y1=x.^2-3*x+2;
>> plot(x,y1,'-bo')
>> y2=x.*cos(x)+x;
>> hold on
>> plot(x,y2,'-rs')
```



Otra forma de crear gráficos múltiples es mediante el comando:

$$subplot(m, n, p), plot(x, y)$$

En este caso se genera una figura con $m \times n$ gráficos y en el plot p (numerado desde arriba hacia abajo y desde la izquierda a la derecha) se grafica los vectores x, y. Por ejemplo, la siguiente secuencia de comandos genera la figura:

```
>> x=[0:0.05:6];
>> y1=x.*sin(x); y2=x.*cos(x); y3=x.*sin(x.^2); y4=x.*cos(x.^2);
>> subplot(2,2,1), plot(x,y1,'-b')
>> subplot(2,2,2), plot(x,y2,'-r')
>> subplot(2,2,3), plot(x,y3,'-g')
>> subplot(2,2,4), plot(x,y4,'-k')
```