

Fundamentos básicos de Matlab

Instrucciones y variables

Matrices

Gráficos

Flujo de control

Fuentes y funciones

13.00 Introducción la ciencia y tecnología oceánica

¿Por qué Matlab?

- Combina cálculos numéricos, gráficos y programación:
 - potente
 - fácil de usar (¿?)
- Las cajas de herramientas proporcionan acceso a cientos de rutinas útiles.
- Uso extendido en la educación de ingeniería.
- Las últimas ediciones de muchos libros de texto utilizan Matlab.
- Muchas de las materias que se imparten en el MIT utilizan Matlab.
- Matlab 5.x proporciona características de programación potentes, como por ejemplo las estructuras de datos y matrices de celdas.

Instrucciones y variables

Introducir y visualizar una matriz A

```
>> A=[1 2; 4 6]
```

```
A =
```

```
    1    2  
    4    6
```

El punto y coma suprime la salida:

```
>> A=[1 2; 4 6];
```

```
>> A=[1 2; 4 6]
```

```
A =
```

```
    1    2  
    4    6
```

```
>>
```

Instrucciones y variables

Operadores de Matlab:

+ suma
- resta
* multiplicación
/ división
^ potencia

Puede utilizar Matlab como calculadora:

```
>> 12.4/6.9
```

```
ans =
```

```
1.7971
```

Si no se da ninguna asignación, el resultado se ubica en la variable `ans` (respuesta).

Nombres de variables

Las variables de Matlab deben comenzar por una letra.

El resto de los caracteres pueden ser letras, dígitos o subrayados.

Únicamente son importantes los 19 primeros caracteres.

```
>> jleonardjleonardjleonard = 1
jleonardjleonardjle =
    1
>>
```

Matlab distingue entre mayúsculas y minúsculas.

```
>> M = [1 2];
>> m = [3 5 7];
```

M y m no son lo mismo.

Variables predefinidas

pi Inf NaN I j

```
>> z = 3 + 4*i
```

```
z =
```

```
3.0000 + 4.0000i
```

```
>> inf
```

```
ans =
```

```
Inf
```

```
>> 0/0
```

```
Advertencia: dividir por cero
```

```
ans =
```

```
NaN
```

```
>>
```

Gestión de su espacio de trabajo (*workspace*)

La función **who** realiza un listado de las variables que se encuentran en el espacio de trabajo.

```
>> who
```

```
Sus variables son:
```

```
A          M          ans          m          z
```

La función **whos** realiza un listado del tamaño y de la asignación de memoria de sus variables.

```
>> whos
```

Name	Size	Elements	Bytes	Density	Complex
A	2 by 2	4	32	Full	No
M	1 by 2	2	16	Full	No
ans	1 by 1	1	8	Full	No
m	1 by 3	3	24	Full	No
z	1 by 1	1	16	Full	Yes

```
Grand total is 11 elements using 96 bytes
```

```
>> whos
```

Nombre	Tamaño	Elementos	Bytes	Densidad	Complejo
A	2 por 2	4	32	Lleno	No
M	1 por 2	2	16	Lleno	No
Ans	1 por 1	1	8	Lleno	No
M	1 por 3	3	24	Lleno	No
Z	1 por 1	1	16	Lleno	Si

```
El total son 12 elementos utilizando 96 bytes
```

```
>>
```

Gestión de su espacio de trabajo (*workspace*)

El comando `clear` se puede utilizar para suprimir variables del espacio de trabajo.

```
>> clear A
>> who
```

```
Sus variables son:
```

```
M          ans          m          z
```

```
>>
```

Si no se añade ninguna razón al comando `clear`, éste borrará todas sus variables.

```
>> clear
>>who
```

```
Sus variables son:
```

```
>>
```

Formatos de salida

La función `format` cambia la precisión de los datos de salida

```
>> pi
ans =
    3.1416

>> format long; pi
ans =
    3.14159265358979

>> format short e; pi
ans =
    3.1416e+00

>> format long e; pi
ans =
    3.141592653589793e+00

>> format rat; pi
ans =
    355/113
```

Formatos de salida

```
>> help format
```

FORMAT Ajustar el formato de salida.

Todas las computaciones de MATLAB se realizan con doble precisión. FORMAT puede utilizarse para hacer cambios entre distintos formatos de visualización de salida, como pueden ser los siguientes:

FORMAT	Por defecto. Lo mismo que SHORT.
FORMAT SHORT	Formato de punto fijo a escala con 5 dígitos.
FORMAT LONG	Formato de punto fijo a escala con 15 dígitos.
FORMAT SHORT E	Formato de punto flotante con 5 dígitos.
FORMAT LONG E	Formato de punto flotante con 15 dígitos.
FORMAT HEX	Formato hexadecimal.
FORMAT +	Los símbolos +, - y espacio en blanco se emiten para elementos positivos, negativos y de valor cero. No se tienen en cuenta las partes imaginarias.
FORMAT BANK	Formato fijo para dólares y céntimos.
FORMAT COMPACT	Suprime suministros de línea adicionales.
FORMAT LOOSE	Vuelve a colocar los suministros de línea adicionales en su posición.
FORMAT RAT	Aproximación por ratio de números enteros pequeños.

Creación de matrices

```
>> A = [3 4; 7 6]
```

```
A =
```

```
    3    4  
    7    6
```

```
>>
```

```
>> A = [1, -4*j, sqrt(2);  
log(-1) sin(pi/2) cos(pi/3)  
asin(0.5), acos(0.8) exp(0.8)]
```

```
A =
```

```
    1.0000                0 - 4.0000i    1.4142  
          0 + 3.1416i    1.0000          0.5000  
    0.5236                0.6435          2.2255
```

```
>>
```

Operadores de matrices

```
>> A = [1 3; 5 9]; B = [4 -7; 10 0];
>> A+B
ans =
     5     -4
    15      9
>> A*B
ans =
    34     -7
   110    -35
>> b=[1;5];
>> A*b
ans =
    16
    50
>> A'
ans =
     1     5
     3     9
>>
```

Operadores de matrices elemento a elemento

```
. * multiplicación  
./ división  
. ^ potencia
```

```
>> A=[1;2;3]; B=[-6;7;10];
```

```
>> A*B
```

```
??? Error using ==> * (mensaje de error)
```

```
Inner matrix dimensions must agree. (Las dimensiones de la  
matriz interna deben concordar)
```

```
>> A.*B
```

```
ans =
```

```
    -6
```

```
    14
```

```
    30
```

```
>> A.^2
```

```
ans =
```

```
     1
```

```
     4
```

```
     9
```

```
>>
```

Notación de coma

Para crear un vector \mathbf{x} con un valor inicial \mathbf{x}_i , aumento \mathbf{dx} y el valor final \mathbf{x}_f , utilizando la notación de coma.

```
x = [xi: dx : xf];
```

Ejemplos

```
>> i = 1:5
i =
     1     2     3     4     5
>> x = 0.1:0.1:1
x =
Columns 1 through 4
    0.1000    0.2000    0.3000    0.4000
Columns 5 through 7
    0.5000    0.6000    0.7000
Columns 8 through 10
    0.8000    0.9000    1.0000
```

La comprensión del uso de las notaciones de comas es esencial para un dominio total de matlab.

Gráficos

Comandos básicos para la realización de diagramas

Tipos de línea y colores

Realces para embellecer sus diagramas

Cómo usar **hold** y **subplot**

Establecer los límites del eje: **axis** y **zoom**

Comandos básicos para la realización de diagramas

Cuatro tipos de diagramas bidimensionales:

`plot(x,y)` dibuja el vector x frente al y

`semilogx(x,y)` realiza un diagrama con eje x \log_{10} y eje y lineal

`semilogy(x,y)` realiza un diagrama con eje x lineal y eje y \log_{10}

`loglog(x,y)` realiza un diagrama con los dos ejes \log_{10}

Tipos de línea y tamaños

Se pueden obtener distintos tipos de línea, símbolos de diagramas y colores con `plot(x,y,s)`, donde `s` corresponde a una fila de 1, 2 ó 3 caracteres formada a partir de los caracteres siguientes:

<code>y</code>	<code>yellow</code>	(<i>amarillo</i>)	<code>.</code>	<code>point</code>	(<i>punto</i>)
<code>m</code>	<code>magenta</code>		<code>o</code>	<code>circle</code>	(<i>círculo</i>)
<code>c</code>	<code>cyan</code>		<code>x</code>	<code>x-mark</code>	(<i>marca de x</i>)
<code>r</code>	<code>red</code>	(<i>rojo</i>)	<code>+</code>	<code>plus</code>	(<i>más</i>)
<code>g</code>	<code>green</code>	(<i>verde</i>)	<code>-</code>	<code>solid</code>	(<i>sólido</i>)
<code>b</code>	<code>blue</code>	(<i>azul</i>)	<code>*</code>	<code>star</code>	(<i>estrella</i>)
<code>w</code>	<code>white</code>	(<i>blanco</i>)	<code>:</code>	<code>dotted</code>	(<i>de puntos</i>)
<code>k</code>	<code>black</code>	(<i>negro</i>)	<code>-.</code>	<code>dashdot</code>	(<i>guión y punto</i>)
			<code>--</code>	<code>dashed</code>	(<i>con guiones</i>)

Por ejemplo, la siguiente orden realiza un diagrama de `x` frente a `y` utilizando signos más azules

```
plot(x,y, 'b+')
```

Comandos adicionales para la realización de diagramas

- `title('text')` – añadir título.
- `xlabel('text')` – añadir xlabel.
- `ylabel('text')` – añadir ylabel.
- `text(p1, p2, 'text', 'sc')` – coloca `'text'` en `(p1, p2)` en coordenadas de pantalla donde `(0.0, 0.0)` se sitúa en la esquina inferior izquierda de ésta y `(1.0, 1.0)` en la esquina superior derecha.
- `Subplot` – subdivide la ventana.

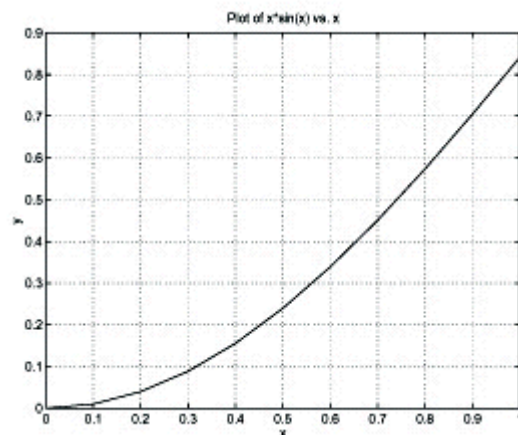
Comandos adicionales para la realización de diagramas

- **axis** – cambiar ejes.
- **axis('equal')** – igualar el ratio de aspecto.
- **grid** – añade líneas de retícula.
- **hold** – le permite realizar múltiples diagramas en el mismo subplot.
- **zoom** – permite hacer zoom (utilizando el ratón)

Nota: **grid**, **hold** y **zoom** funcionan como un “toggle” (*conmutador*), ya que repetidas llamadas conectan o desconectan la propiedad.

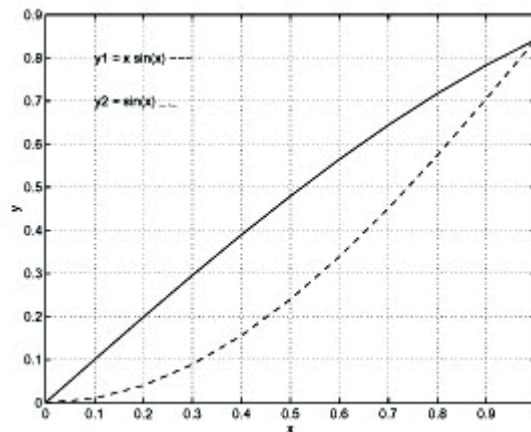
Ejemplo de un diagrama sencillo

```
>> x = [0:0.1:1]';
>> y = x.*sin(x);
>> clf      % despeja la figura actual
>> plot(x,y)
>> title('Plot of x*sin(x) vs. x');
>> xlabel('x')
>> ylabel('y')
>> grid
>> print -deps plot1.eps
>> !lp plot1.eps
request id is test-878 (1 file(s))
```



Otro diagrama sencillo

```
>> x = [0:0.1:1]';  
>> y1 = x.*sin(x);  
>> y2 = sin(x);  
>> plot(x, y1, '--', x, y2, '-');  
>> f1  
>> text(0.1,0.8,'y1 = x sin(x) ---');  
>> text(0.1,0.75,'y2 = sin(x) -.-');  
>> xlabel('x'); ylabel('y'); grid;
```

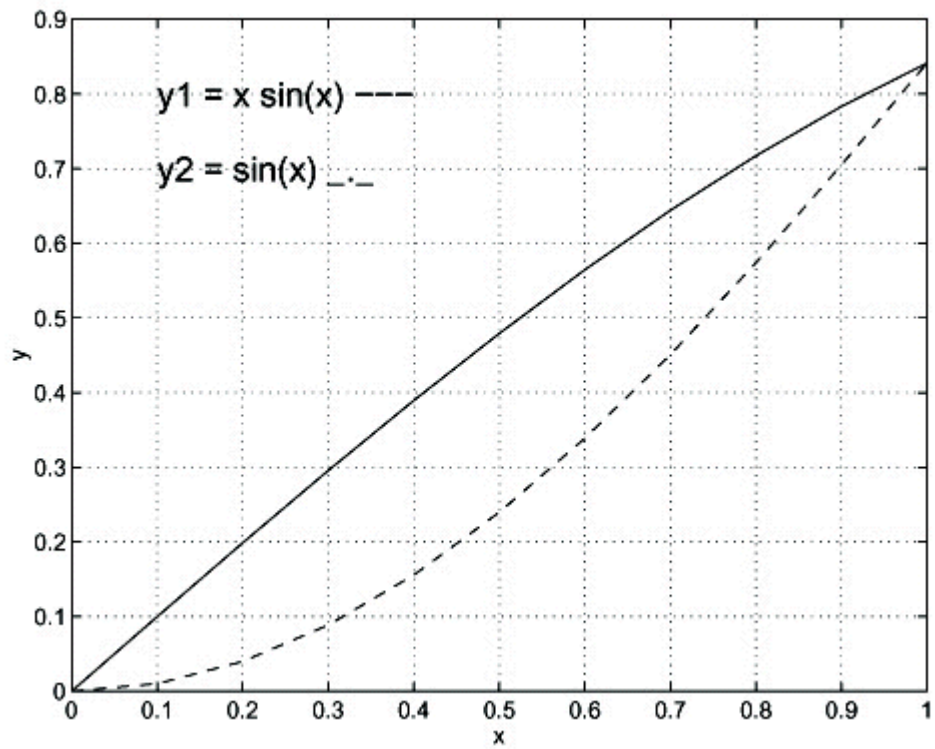


Cómo utilizar `get` (obtener) y `set` (establecer)

```
>> t1 = text(0.1,0.8,'y1 = x sin(x) ---');
>> t2 = text(0.1,0.7,'y2 = sin(x) _._');
>> get(t1)
    Color = [1 1 1]
    EraseMode = normal
    Extent = [0.0900693 0.730205 0.489607 0.117302]
    FontAngle = normal
    FontName = Helvetica
    FontSize = [18]
    FontWeight = normal
    HorizontalAlignment = left
    Position = [0.1 0.8 0]
    Rotation = [0]
    String = y1 = x sin(x) ---
    Units = data
    VerticalAlignment = middle

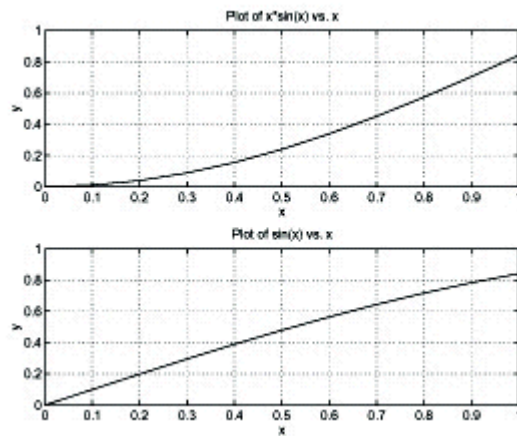
    ButtonDownFcn =
    Children = []
    Clipping = off
    Interruptible = no
    Parent = [58.0005]
    Type = text
    UserData = []
    Visible = on
>> set(t1, 'FontSize', 18)
>> set(t2, 'FontSize', 18)
```

Cómo utilizar `get` (obtener) y `set` (establecer)



Cómo utilizar subplot (subdiagrama)

```
>> subplot(211)
>> plot(x,y1);
>> xlabel('x'); ylabel('y'); grid;
>> title('Plot de x*sin(x) frente a x')
>> subplot(212)
>> plot(x,y2);
>> xlabel('x'); ylabel('y'); grid;
>> title('Plot de sin(x) frente a x')
>> print -deps plot3.eps
```



Flujo de control - Decisiones

Comandos de Matlab para decisiones:

`if, elseif, else, y end`

Ejemplo

```
d = date;
day = str2num(d(1:2));
if ((floor(day/2)*2) == day)
    disp ('hoy el día del mes es par');
else
    disp ('hoy el día del mes es impar');
end
```

Salida

```
>> date
ans =
23-Feb-97
>> d = date;
>> day = str2num(d(1:2));
>> if ((floor(day/2)*2) == day)
    disp ('hoy el día del mes es par');
else
    disp ('hoy el día del mes es impar');
end
hoy el día del mes es impar
```

Flujo de control - Bucles

Comandos de Matlab para bucles:

for y while

Ejemplos

```
% calcula un factorial con bucle for
n = 10;
factorial=1;
for i=1:n
    factorial = factorial * i;
end
factorial
```

```
% calcula un factorial con bucle while
i=1;
factorial=1;
while (i < 10)
    i = i+1;
    factorial = factorial * i;
end
factorial
```

Ficheros de comandos (*scripts*) y funciones

Los ficheros de comando y las funciones se denominan **M-files** (*ficheros M*), puesto que llevan el sufijo “.m”.

Los ficheros de comando son archivos de texto que contienen una secuencia de comandos de matlab.

Las funciones son **M-files** que devuelven valores.

La mayor diferencia entre los ficheros de comando y las funciones radica en que las variables que se crean en las funciones son variables locales, mientras que las variables que se crean en los ficheros de comando son globales.

Las cajas de herramientas de matlab son colecciones de **M-files** útiles.

Si escribe sus propios ficheros de comandos y funciones le resultará más fácil y más eficiente utilizar matlab.

Un sencillo fichero de comando de matlab

```
% simple.m - un fichero de comando sencillo de matlab.  
%  
% Este fichero de comando realiza un diagrama sencillo de la  
% función sin. Supone que alfa viene definido en el espacio de  
% trabajo antes de que solicite el archivo.
```

```
t = [0:0.01:1];  
y = sin(alpha * t);  
plot(t,y)  
xlabel('time (sec)');  
ylabel('ty(t) = sin(alpha*t)');  
title('diagrama sencillo por jleonard 23/02/97');  
grid on
```

```
>> simple  
>> help simple
```

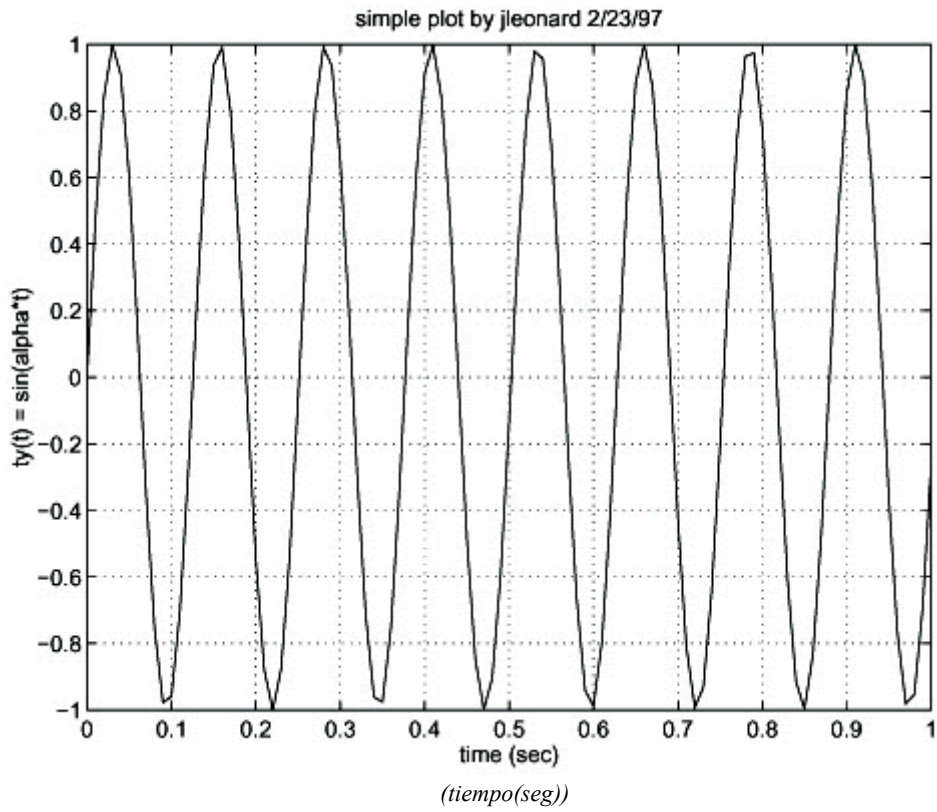
```
simple.m - un fichero de comando sencillo de matlab.
```

```
Este fichero de comando realiza un diagrama sencillo de  
la función sin. Supone que alfa viene definido en el  
espacio de trabajo antes de que solicite el archivo.
```

```
>>
```

Gráfico realizado por simple.m

(Diagrama sencillo por jleonard 23/02/97)



Ejemplo: patrón de haz del sonar de delfines

```
% archivo de fichero de comandos para realizar un patrón de
  haz para su distribución en clase
% jleonard 20/10/96
f = 120000;
lambda = 1500/f;
theta = (-90:0.01:90) * pi / 180;
k = 2 * pi/lambda;
a = 0.037/2;
theta3 = 29.3 * lambda / (2 * a);

figura(1)
clf
bp = cpbeam(theta, k, a);
bp1 = bp;
plot(theta*180/pi,10 * log10(bp))
axis([-90 90 -80 0]);
set(gca, 'Xtick', [-90:30:90]);
grid on
hold on
plot([theta3 theta3], [-80 0], '-.')
plot([-theta3 -theta3], [-80 0], '-.')
title(['f = 120 kHz  D = ', num2str(2 * a), ...
      ' m  beamwidth = +- ', num2str(theta3), ' deg']);
xlabel('\theta (grados)');
ylabel('Nivel de fuente normalizado(dB)')
```

cpbeam.m

```
función bp = cpbeam(theta,k,a)

% CPBEAM: Patrón de haz para un transductor de pistón
% circular, utilizando el modelo de función de un barco
% estándar.
%   bp = cpbeam(theta,k,a)
%   bp = cpbeam(theta,ka)
%
% CPBEAM devuelve la función del haz normalizada para el
% número k de ondas y el radio a del transductor en un ángulo
% de apertura theta (radianes).
% autor: Bradley a. Moran, programa Sea Grant del MIT, 1993.

Si el margen es < 3, a = 1; end

ReducedFreq = k*a*sin(theta)

Bp = (2*bessel1(1,abs(reducedFreq)) ./reducedFreq).^2;
```

Patrón del haz

