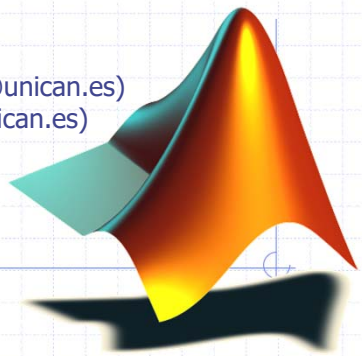
Herramientas Computacionales en el Laboratorio 2018-2019

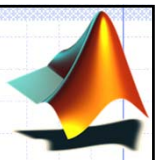
INTRODUCCIÓN A MATLAB

Profesores:
José Carlos Palencia Gutiérrez (palencij@unican.es)
Julio Largo Maeso (julio.largomaeso@unican.es)



Curso basado en el libro "Introducción a Matlab y sus aplicaciones"
Peregrina Quintela Estévez, Universidad de Santiago de Compostela

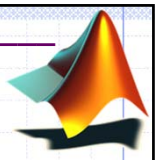
INTRODUCCIÓN



- ¿Qué es Matlab?, MATrix LABoratory

MATLAB es un programa para realizar cálculos numéricos con ***vectores*** y ***matrice***s. Como caso particular puede también trabajar con números escalares, tanto reales como complejos. Una de las capacidades más atractivas es la de realizar una amplia variedad de ***gráficos*** en dos y tres dimensiones. MATLAB.

Introducción



Los elementos básicos del Matlab, como cualquier otro lenguaje de programación, son: constantes, variables, operaciones, expresiones y funciones.

Constante numéricas:

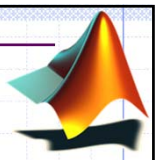
- Números enteros: 2 35 -48
- Números reales: 2. -35.2 48.45
  - Máximo de 16 cifras significativas
  - Utilizando la letra **e** a continuación de un nº con punto decimal [2.2250e-308 1.7e+308].
- Números complejos: 2+3i 4*j i,j=$(-1)^{½}$

Operaciones aritméticas elementales:

Suma: + Multiplicación: * Exponenciación: ^
Resta: - División: /

Precedencia: primero exponenciaciones, luego divisiones y multiplicaciones, por último sumas y restas.

Introducción



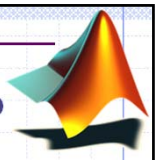
Variables: es la etiqueta que identifica una porción de memoria.

Matlab diferencia entre mayúsculas y minúsculas.

Para ver las variables definidas en un instante determinado se teclea:
>> who
o bien
>> whos
Para eliminar alguna variable se ejecuta
>> clear variable1 variable2

Expresiones numéricas: son un conjunto de números, funciones y variables previamente definidas, relacionados todos ellos por operadores aritméticos. Si una expresión es demasiado larga se indica mediante ...

Introducción



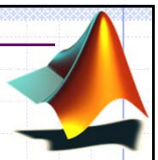
Formatos: por defecto matlab tiene formato corto pero se puede elegir entre los siguientes formatos.

- >> format long (14 cifras significativas)
- >> format short (5 cifras significativas)
- >> format short e (notación exponencial)
- >> format long e (notación exponencial)
- >> format rat (aproximación racional)

Variables predefinidas en Matlab:

i = $(-1)^{½}$ pi = π Inf= ∞ NaN= cálculos indefinidos
date = valor de la fecha actual
rand = genera números aleatorios [0 1]
realmin = menor nº positivo
realmax= mayor nº positivo
eps = 2.22e-16 (relativo a la mínima precisión)

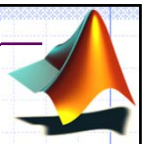
Introducción



Funciones de matlab: nombre(argumento)

- sqrt (x) raiz cuadrada.
- abs(x) módulo de x.
- conj(z) conjugado de un complejo.
- real(z), imag(z) parte real e imaginaria de z.
- abs(z), angle (z) módulo y fase de un complejo.
- exp(x) calcula e^x, siendo x real o complejo.
- sin(x) cos(x) tan(x) (ángulo en radianes).
- asin(x) [$-\pi/2$ $\pi/2$] acos(x) [0 π] atan(x) [$-\pi/2$ $\pi/2$]
- log(x) (en base e) log10(x) (en base 10)
- rats(x) expresa un número en forma racional.
- rem(x,y) resto de x/y.
- round(x) redondeo.
- sign(x) signo de x (1 si x positivo, -1 si negativo)

etc…



Comandos de ayuda:

- help
- lookfor
- Seleccionando cualquier palabra escrita y pulsando F1.

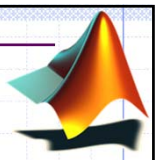
- what lista los ficheros .m y .mat del directorio actual.
- dir lista todos los ficheros del directorio actual.
- type nombre_fichero Muestra el contenido del fichero
- delete nombre_fichero Borra el fichero
- cd Cambia de directorio
- pwd Indica el directorio actual
- which nombre_fichero Indica el directorio donde está

Para guardar en un fichero los comandos que se ejecutan en una sesión se pone

>> diary nombre_fichero

...

>> diary off



Ejercicio 1.1 Calcular el valor de la expresión:

$$J = \frac{42.1768 + 234}{2^{10} - 10247}$$

Ejercicio 1.2 Calcular el valor de la expresión:

$$H = \frac{9.8 * 10^{14} + 5.876 * 10^{-5}}{9.987 * 10^{5} - 10^{6}}$$

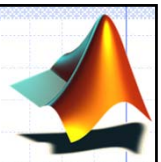
Y escribir el resultado en, al menos, 2 formatos.

Ejercicio 1.3 Calcular $I = \sqrt[7]{\frac{3sen(32º15')}{42.1^3}}$

Ejercicio 1.4 Según Hill y Lounasmaa, la ecuación de la curva de inversión del helio es $P = -21.0 + 5.44T - 0.132T^2$ donde la presión viene dada en atmósferas y la temperatura T en Kelvin. Calcular el valor de la presión a una temperatura de 23 K. Calcular el valor de la temperatura para una presión de 1N/m^2.

Nota: 1N/m^2=9.265*10^{-6}atm.

PROGRAMACIÓN

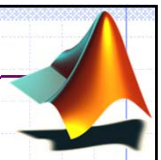


Estructuras de control condicionadas

if condición
 % instrucciones a ejecutar si se cumple la condición
end

if condición
 % instrucciones a ejecutar si se cumple la condición
else
 % instrucciones a ejecutar si no se cumple
end

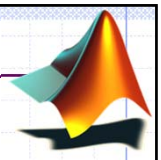
A su vez, dentro de cada conjunto de instrucciones a ejecutar se pueden poner más estructuras if anidadas.



Ejemplo El siguiente programa pide al usuario que introduzca un número por teclado e indica si ese número es positivo o no

```
x=input('Introduce un numero: ');
if x>0
    disp('El numero introducido es positivo');
else
    disp('El numero introducido no es positivo');
end;
```

Programación



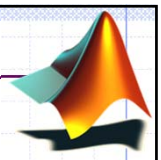
También se admiten otras variantes de estructuras if anidadas para facilitar su lectura

```
if condición1
  ...
else
    if condición2
      ...
    else
        if condición3
          ...
        else
          ...
        end
    end
end
```

⇔

```
if condición1
  ...
elseif condición2
      ...
elseif condición3
        ...
else
        ...
end
```


Programación



Ejemplo: Igual al ejemplo anterior, distinguiendo el caso del valor cero

```
x=input('Introduce un numero:');
if x==0
  disp ('El numero es 0');
else
  if x>0
    disp ('El numero es positivo');
  else
    disp ('El numero es negativo');
  end
end
```

⇔

```
x=input('Introduce un numero:');
if x==0
    disp ('El numero es 0');
elseif x>0
    disp ('El numero es positivo');
else
    disp ('El numero es negativo');
end
```

Programación

Bucles con condición de finalización

Repite un conjunto de instrucciones mientras se cumpla una condición de control:

```
while condición
    % instrucciones a ejecutar mientras se cumple la condición
end
```

Normalmente, en las instrucciones dentro del bucle debe haber alguna que modifique la condición de control (o salir con break).

Ejemplo: Sumar todos los números enteros entre 1 y 100, incluidos.

```
suma=0;
numero=1;
while numero<=100
        suma=suma+numero;
        numero=numero+1;
end
```


Programación

Bucles for

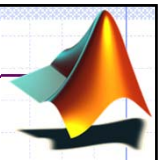
Repite un conjunto de instrucciones un cierto número de veces, indicado por una variable de control:

```
for k=n1:m:n2
    % instrucciones a ejecutar
end
```

Repite las instrucciones para cada valor de k indicado (desde n1 hasta n2 en saltos de valor m). El incremento se realiza automáticamente al final de cada iteración.

Ejemplo: Sumar todos los números enteros entre 1 y 100, incluidos.

```
suma=0;
for numero=1:1:100;
        suma=suma+numero;
end
```

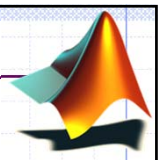


Ejercicio 2.1 Escribir un programa que pida al usuario que introduzca los tres coeficientes de una ecuación de 2º grado

$$ax^2+bx+c=0$$

y después calcule sus soluciones y las pinte en pantalla.

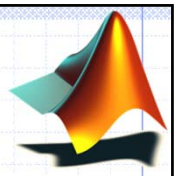
Ejercicio 2.2 Modificar el programa anterior para que se indique, además, si las soluciones son reales o complejas.



Ejercicio 2.3 Escribir un programa que vaya leyendo números positivos introducidos por el usuario y calcule la suma de todos ellos hasta que el número introducido sea negativo.

Ejercicio 2.4 Escribir un programa que pida al usuario que introduzca 10 números y calcule el valor máximo, el mínimo y la suma de todos ellos .

VECTORES Y MATRICES



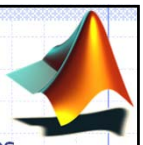
- Las matrices son el tipo fundamental de dato en Matlab.

```
» A=[1 3 5; 6 9 2; 4 8 7]

A =

     1     3     5
     6     9     2
     4     8     7

» det(A)

ans =

     5
```

```
» A^2+3*A

ans =

    42    79    61
    86   142    68
    92   164   106
```

MATrix LABoratory
-- datos son matrices
-- reglas del álgebra lineal

Vectores

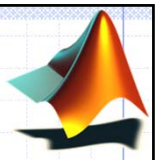


Los vectores pueden ser vectores fila o vectores columna:
Vectores fila: los elementos de una misma fila están separados por ***blancos*** o ***comas***.
>> v =[-2 13 4]
Vectores columna: los elementos de una misma columna están separados por ***intro*** o por caracteres ***punto y coma*** (;).
>> w =[2;3;4;7;9;8]

Se puede acceder a un elemento escribiendo su índice entre paréntesis. Ejemplo: v(2) devolvería el valor 13.

Otras formas de generar vectores:
- Especificando el incremento de sus componentes v=a:h:b;
- Especificando su dimensión linspace(a,b,n). Si se omite n, toma 100 por defecto; el incremento es k=(b-a)/(n-1).
- Con componentes logarítmicamente espaciadas logspace(a,b,n) genera un vector fila de n puntos logarítmicamente espaciados entre 10^a y 10^b. Si se omite el valor de n se toma 50.
- Subvector v=w(i:k:j) Copia desde i hasta j cada k elementos.

Vectores



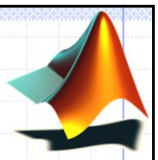
Operaciones con escalares:

v+k	adición o suma
v-k	sustracción o resta
v*k	multiplica cada elemento de v por k
v/k	divide cada elemento de v por k
k./v	divide k por cada elemento de v
v.^k	potenciación: cada componente de v esta elevado a k
k.^v	potenciación: k elevado cada componente de v

Operaciones entre vectores:

v+w	adición o suma
v-w	sustracción o resta
v.*w	multiplica cada elemento de v por el correspondiente de w
v./w	divide cada elemento de v por el correspondiente de w
v.^w	potenciación cada componente de v elevado al correspondiente de w
v*w	producto escalar de vectores (v vector fila y w vector columna)

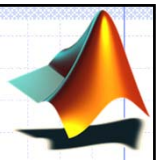
Vectores



Funciones de matlab específicas para vectores:

length(v)	dimensión del vector.
norm(v)	módulo del vector.
sum(v)	suma todos los elementos del vector.
prod(v)	multiplica todos los elementos del vector.
v'	transposición de vectores (filas ↔ columnas)
dot(v,w)	producto escalar de vectores.
cross(v,w)	producto vectorial de vectores.
[y,k]=max(v)	valor máximo de las componentes de un vector. k indica la posición de ese valor.
[y,k]=min(v)	similar al anterior, para el valor mínimo.

Vectores



Ejemplo: Cálculo de errores relativos

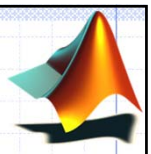
Supongamos que para resolver una ecuación diferencial ordinaria hemos utilizado dos métodos:

- Un método analítico mediante el cual sabemos que su solución en el intervalo [0,1] es $y(x)=x^2+\cos(x)$.
- Un método numérico para aproximar la solución en el intervalo [0,1] con parámetro de discretización 0.2

Ángulo (radianes)	Solución aproximada
0	1.0030
0.2	1.0234
0.4	1.0825
0.6	1.1869
0.8	1.3342
1	1.5415

Calcular los errores relativos cometidos en el método numérico

Vectores



Ejercicio 3.1 Dados los vectores definidos por

x=(1,4.5,7.8) y=(sen(90º),cos(45º),0)

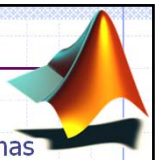
Calcular: x+y, el producto escalar de x e y, calcular el ángulo que forman ambos vectores.

Ejercicio 3.2 Para un laboratorio se compran los materiales especificados en la tabla siguiente:

Referencia artículo	Precio/unidad	Cantidad (uds.)
1520	1146	200
1621	3450	250
1428	6225	150
1429	7100	150
1628	8500	100

Utilizar vectores y operaciones con vectores para calcular el importe a pagar de cada artículo y el importe total (la tabla de precios no incluye IVA, debiendo por tanto aplicarse un 21% de incremento).

Matrices



Para definir una matriz *no hace falta establecer de antemano su tamaño*. MATLAB determina el número de filas y de columnas según el número de elementos que se proporcionan (o se utilizan).

Las matrices se definen por filas; los elementos de una misma fila están separados por ***blancos*** o ***comas***, mientras que las filas están separadas por ***intro*** o por caracteres ***punto y coma*** (;).

Por ejemplo, para definir una matriz **A** de dimensión (3x3):

» A=[1 2 3; 4 5 6; 7 8 9]

La respuesta del programa es la siguiente:

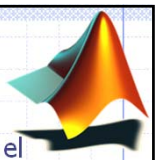
```
A =     1 2 3
        4 5 6
        7 8 9
```

matriz traspuesta: En MATLAB, el apóstrofo (') es el símbolo de trasposición matricial.

matriz inversa: la inversa de **A** se calcula con la función **inv( )**

B=inv(A)

Matrices

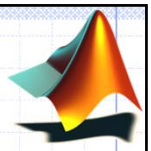


En MATLAB se accede a los elementos de un vector poniendo el índice entre paréntesis, por ejemplo *x(3)* ó *x(i)*. En una matriz, se accede a cada elemento poniendo los dos índices entre paréntesis, separados por una coma, por ejemplo *A(1,2)* ó *A(i,j)*.

Operadores matriciales de MATLAB son los siguientes:

+	adición o suma
–	sustracción o resta
*	multiplicación
'	adjunta (transpuesta o transpuesta conjugada)
^	potenciación
/	división-derecha. A/B equivale a A*inv(B)
\	división-izquierda. A\B equivale a inv(A)*B
.*	producto elemento a elemento
./ y .\	división elemento a elemento
.^	elevar a una potencia elemento a elemento

Matrices

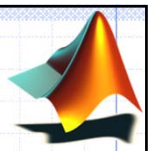


det(A)	Calcula el determinante de la matriz.
diag(A)	Obtención de la diagonal de una matriz.
sum(diag(A))	Calcula la traza de la matriz A.
diag(A,k)	Busca la k-ésima diagonal.
norm(A)	Norma de una matriz.

Generación de matrices:

- Generación de una matriz de ceros, zeros(n,m)
- Generación de una matriz de unos, ones(n,m)
- Generación de una matriz identidad eye(n,m)
- Generación de una matriz de elementos aleatorios rand(n,m)
- Matrices con diagonal dada: diag(v), diag(v,k)
- A partir de vectores: [X Y] columnas, [X; Y] filas

Matrices



Al igual que con vectores, se puede obtener una submatriz a partir de otra matriz, indicando las filas y las columnas:

Por ejemplo, si tenemos la siguente matriz A:

```
A =  1   2   3   4
     5   6   7   8
     9  10  11  12
```

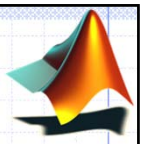
```
» B=A(1:2, 2:4)
B =  2   3   4
     6   7   8
```

También se admiten variables subindicadas multidimensionalmente:

Por ejemplo, si tenemos la matriz a=[1 2; 3 4]

```
» a(1,:)= 1 2        » a(2,:)= 3 4

» a(:,1)= 1          » a(:,2)= 2
          3                    4
```

Matrices



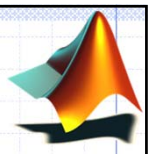
Ejercicio 3.3 Dadas las matrices a y b calcular: a+b, a+0, bI. Comprobar que el producto de matrices no es conmutativo. Elegir la submatriz de a formada por la primera y la segunda filas y la submatriz de b formada por la segunda y tercera columnas y calcular el producto

$$a = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 4 & 2 & 5 \end{pmatrix} \quad b = \begin{pmatrix} 4 & 2 & 1 \\ 2 & 0 & 4 \\ 1 & -2 & -5 \end{pmatrix}$$

Ejercicio 3.4 Utilizar matrices para construir una tabla que contenga: En la 1ª columna la variable grados Celsius en el intervalo [0 100] con un paso de 2. En la 2ª columna su valor en grados Fahrenheit y en la 3ª en Kelvin.

$$\frac{9 * celsius}{5} + 32$$

Matrices



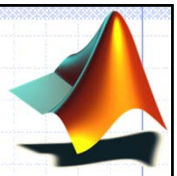
Ejercicio 3.5 Utilizando matrices, resolver el siguiente sistema de ecuaciones:

$$\begin{cases} x_1 + x_2 + x_3 + 1 = 0 \\ 2x_1 - x_2 + 4x_3 - 1 = 0 \\ -x_1 + 3x_2 - x_3 = 0 \end{cases}$$

Comprobar que los valores obtenidos son soluciones de ese sistema.

Nota: También se puede utilizar el comando linsolve (ver ayuda)

POLINOMIOS



Los polinomios se representan en matlab por un vector fila de dimensión n+1 siendo n el grado del polinomio.

Dado un polinomio

x^3+2x

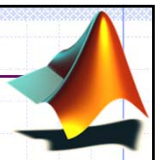
se representa por

```
>> pol1=[1 0 2 0]
```

Para el cálculo de las raíces de un polinomio existe el comando roots.

```
>>raices=roots(pol1) (da un vector columna,
    aunque pol1 sea un vector fila)
```


Polinomios



Un polinomio puede ser reconstruido a partir de sus raíces con el comando poly

```
>> p=poly(raices) (da un vector fila) **
```

En caso de que el argumento de poly fuera una matriz, obtendríamos como resultado el polinomio característico de la matriz. Así mismo si queremos calcular los autovalores de la matriz bastaría con calcular las raíces del polinomio característico.

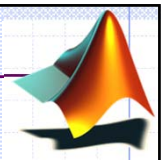
Ejemplo:

```
pol2=[2 4 0 1];          % definición del polinomio 2x^3+4x^2+1
raices=roots(pol2)       % cálculo de sus raíces
pol2_n=poly(raices)      % reconstrucción del polinomio
```

Ejemplo:

```
A=[1 2 3 ; 2 3 4; 4 2 5];
p=poly(A)        % pol. característico
roots(p)         % autovalores de A
```

Polinomios



Para calcular el valor de un polinomio p en un punto dado x se puede utilizar el comando polyval

```
>>y=polyval(p,x)
```

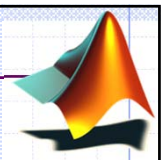
Ejemplo:

```
p=[1 −1 −1 1] % definición del polinomio x^3-x^2-x+1
polyval(p,2)
```

Ejemplo:

```
x=-2:0.1:2;
y=polyval(p,x);
```


Polinomios



Para multiplicar y dividir polinomios tenemos los comandos especiales conv(p1,p2) y deconv(p1,p2)

Ejemplo:

```
>>p1=[1  -2  1]; p2=[1 1];
>>p3=conv(p1,p2)
>>p=deconv(p3,p2)
```

Si queremos conocer, además, el resto de la división de polinomios podemos usar el mismo comando

```
>>[p,r]=deconv(p1,p2) % resto de la división
```

$$\frac{p1(x)}{p2(x)} = p(x) + \frac{r(x)}{p2(x)}$$

Polinomios

El comando residue permite el cálculo del desarrollo en suma de fracciones simples del cociente p1/p2 (p2 debe tener raíces reales)

El formato del comando es: >>[m,r,k] =residue(p1,p2)
donde:
m = vector columna con los numeradores
r = vector columna con las raices del denominador
k = vector fila con los coeficientes del polinomio independiente.

$$\frac{p1(x)}{p2(x)} = \frac{m(1)}{x - r(1)} + \cdots + \frac{m(n)}{x - r(n)} + k(x)$$

*En caso de que existan raices múltiples, la expresión anterior cambia (mirar en la ayuda: **help residue**)

Polinomios

>>[p1,p2]=residue(m,r,k) hace la operación inversa

Ejemplo 6:
Descomponer en fracciones simples el cociente:

$$\frac{x^3 + x^2 + 1}{x^3 - 3x^2 + 4}$$

```
p1=[1 1 0 1]
p2=[1 –3 0 4]
[m,r,k]=residue(p1,p2)
rats(m) % Expresamos los coeficientes como fracciones simples
```

$$\frac{x^3 + x^2 + 1}{x^3 - 3x^2 + 4} = \frac{35}{9(x-2)} + \frac{13}{3(x-2)^2} + \frac{1}{9(x+1)} + 1$$

```
[pol1,pol2]=residue(r,p,k)  % Operación inversa
```

Polinomios

>>p=polyfit(x,y,n)

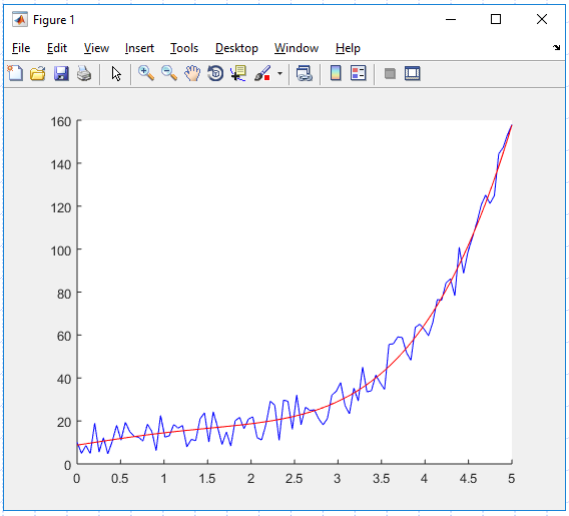
Ajuste por mínimos cuadrados: El comando polyfit calcula los coeficientes de un polinomio p de grado n que se ajusta a la nube de puntos x-y de forma que el error cuadrático medio sea mínimo.

Ejemplo:

```
% Creamos algo parecido a una nube de puntos: 100 puntos para x
% entre 0 y 5 y una función exponencial con ruido aleatorio para y
x=linspace(0,5,100);
y=exp(x)+ 20*rand(1,100);
% Buscamos un polinomio de grado 4 que ajuste los puntos
p=polyfit(x,y,4)
% Evaluamos el polinomio en los puntos x para poder pintar una
% grafica comparando la nube con el ajuste.
yp=polyval(p,x);
hold on;
plot(x,y,'b');
plot(x,yp,'r');
```


Polinomios

El resultado será parecido a:



Polinomios

Para calcular la derivada de un polinomio tenemos el comando,

```
>>polyder(p)
```

Ejemplo 7:
Dado el polinomio x^3+6x^2+1 su derivada es

```
>>p=[1, 6,0,1];
>>d=polyder(p)
 d=
        3   12   0    % es decir 3x^2+12x
```


Polinomios

Ejercicio 4.1 Sea el polinomio $p(x)=x-1$. Calcular $p(x)^3$ e identificar el polinomio obtenido y calcular sus raíces.

Ejercicio 4.2 Según Hill y Lounasmaa, la ecuación de la curva de inversión del helio es $P=-21+5.44T-0.132T^2$ donde P viene dada en atmósferas y T en Kelvin.

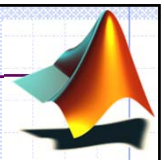
a) Calcular el valor de la presión a una temperatura de –268.25 ºC.
b) Calcular el valor de la temperatura para una presión de $3N/m^2$. (Nota: $1N/m^2=9.265e-6$ atm)

Ejercicio 4.3 Calcular la solución general de la EDO

$$y^{(5)}-y^{(4)}+2y'''-2y''+y'-y=0$$

Ejercicio 4.4 Calcular al integral

$$I=\int \frac{x^4+2x+1}{x-1}dx$$



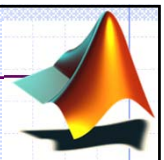
Solución Ejercicio 4.3

```
>>format long
>>p=[1 –1  2 –2  1 –1];
>>raices=roots(p)

>>der_p=polyder(p)
>>polyval(der_p,round(raices(2)))
```

La solución exacta es:
$y(x)=c_1e^x+c_2sen(x)+c_3cos(x)+c_4xsen(x)+c_5xcox(x)$

Comando Matlab:

```
>>dsolve('D5y-D4y+2*D3y-2*D2y+Dy-y=0','x')
```

Ejercicio 4.4

```
>>numerador=[1 0 0 2 1];
>>denominador=[1 –1];
>>[cociente,resto]=deconv(numerador,denominador)
```

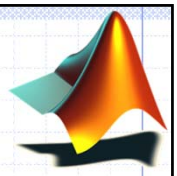
$$I=\int\frac{x^4+2x+1}{x-1}dx=\int\left(x^3+x^2+x+3+\frac{4}{x-1}\right)dx=$$

$$=\frac{x^4}{4}+\frac{x^3}{3}+\frac{x^2}{2}+3x+4\ln(x-1)$$

Comando Matlab:

```
>>int('(x^4+2*x+1)/(x-1)')
```

GRÁFICOS: 2D Y 3D

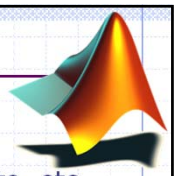


Funciones gráficas 2D elementales:

MATLAB dispone de 4 funciones básicas para crear gráficos 2-D. Estas se diferencian principalmente por el *tipo de escala* que utilizan en los ejes Estas cuatro funciones son las siguientes:

- plot() crea un gráfico a partir de vectores y/o columnas de matrices, con escalas lineales sobre ambos ejes.
- loglog() ídem con escala logarítmica en ambos ejes.
- semilogx() ídem con escala lineal en el eje de ordenadas y logarítmica en el eje de abscisas.
- semilogy() ídem con escala lineal en el eje de abscisas y logarítmica en el eje de ordenadas.

GRÁFICOS: 2D Y 3D



Existen funciones orientadas a añadir títulos al gráfico, a los ejes, dibujar una cuadrícula auxiliar, introducir texto, etc.

- title('título') añade un título al dibujo
- xlabel('tal') añade una etiqueta al eje de abscisas. Con ***xlabel off*** desaparece
- ylabel('cual') idem al eje de ordenadas.
- text(x,y,'texto') introduce 'texto' en el lugar especificado por las coordenadas **x** e y. Si **x** e **y** son vectores, el texto se repite por cada par de elementos.
- gtext('texto') introduce **texto** con ayuda del ratón:
- legend() define rótulos para las distintas líneas o ejes utilizados en la figura.
- grid activa una cuadrícula en el dibujo. Con g***rid off*** desaparece la cuadrícula

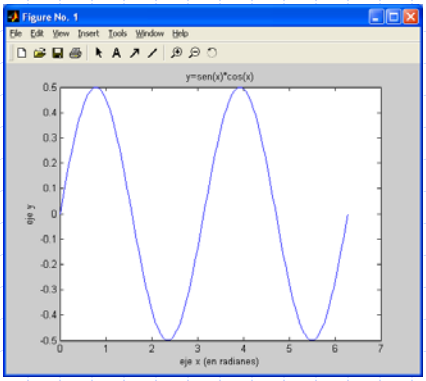
GRÁFICOS: 2D y 3D

FUNCIÓN *PLOT*

plot es la función clave de todos los gráficos 2-D en MATLAB

Ejemplo 1

```
x=0:pi/90:2*pi;
y=sin(x).*cos(x);
plot(x,y)

grid on
grid off
xlabel('eje x (en radianes)')
ylabel('eje y')
title('y=sen(x)*cos(x)')
```



GRÁFICOS: 2D Y 3D

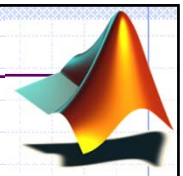
FUNCIÓN *PLOT*

Es posible incluir en el título o en la etiqueta de los ejes el valor de una variable numérica. Ya que el argumento de los comandos title, xlabel e ylabel es una variable de texto, es preciso transformar las variables numéricas

int2str(n) convierte el valor de la variable entera n en carácter

num2str(x) convierte el valor de la variable real o compleja x en carácter

GRÁFICOS: 2D Y 3D



Texto sobre la gráfica

gtext('texto')
text(x,y,'texto a imprimir')

Calcular las coordenadas de puntos sobre la curva

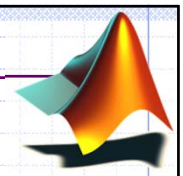
ginput(n)
[x,y]=ginput(n)

Elección del trazo y color de la curva

plot(x,y,'opcion')

hold on
hold off

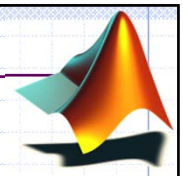
GRÁFICOS: 2D Y 3D



Opciones de plot

y	yellow	.	Point	-	solid
m	magenta	o	circle	:	dotted
c	cyan	x	x-mark	-.	dashdot
r	red	+	plus	--	dashed
g	green	*	star		
b	blue	s	square		
w	white	d	diamond		
k	black	v	triangle (down)		
^	triangle (up)	p	pentagram		
<	triangle (left)	h	hexagram		
>	triangle (right)				

GRÁFICOS: 2D Y 3D

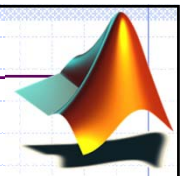


Ejemplo 2: Calcular gráficamente las soluciones de la ecuación:

$$\frac{2x - \cos(2x)}{2} = 0.4$$

```
teta=0:pi/360:pi/4;
f1=(2*teta-cos(2*teta))/2;
f2=0.4*ones(size(f1));
figure
plot(teta,f1,'g--',teta,f2,'r')
axis square
xlabel('Angulo (radianes)')
gtext('2x-cos(2x))/2')
text(0.2,0.43,'y=0.4')
[teta0,y0]=ginput(1)
title(['Raiz aproximada=',num2str(teta0)])
```

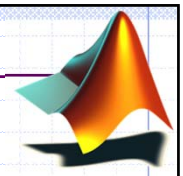

GRÁFICOS: 2D Y 3D



Elección de la escala de los ejes

axis([x0 x1 y0 y1])
axis auto: devuelve la escala a la de defecto
axis off: desactiva los etiquetados de los ejes desapareciendo los ejes sus etiquetas y la grid.
axis on: lo activa de nuevo
axis xy: sistema de coordenas cartesianas origen en el ángulo inferior izquierdo, eje ox de izqda. A dcha. y oy de abajo a arriba.
axis equal: los mismos factores de escala para los dos ejes
axis square: cierra con un cuadrado la región delimitada por los ejes de coordenadas actuales.

GRÁFICOS: 2D Y 3D



Se puede crear una nueva figura en blanco llamando **figure** o referirnos a una figura ya hecha **figure(n)**

clf Borra la figura actual
close all borra todas las figuras
close(figure(n)) borra la figura n

Impresión de gráficas

Podemos guardas gráficos en ficheros:

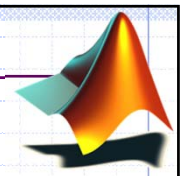
print -djpeg<nn> % JPEG imagen, nn nivel de calidad
% nn es 75 por defecto)

Ejemplo:

print -djpeg90 figura1

Nos genera un fichero "figura1.jpg" con la imagen

GRÁFICOS: 2D Y 3D



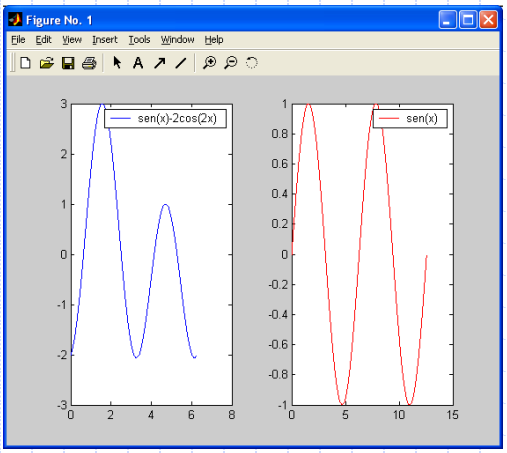
FUNCIÓN *SUBPLOT*

Una ventana gráfica se puede dividir en **m** particiones horizontales y **n** verticales, con objeto de representar múltiples gráficos en ella. Cada una de estas subventanas tiene sus propios ejes, aunque otras propiedades son comunes a toda la figura. La forma general de este comando es:
subplot(m,n,i) donde **m** y **n** son el número de subdivisiones en filas y columnas, e **i** es la subdivisión que se convierte en activa. Las subdiviones se numeran consecutivamente empezando por las de la primera fila, siguiendo por las de la segunda, etc.

GRÁFICOS: 2D Y 3D

Ejemplo 3:

```
subplot(1,2,1)
x=0 : 0.001 : 2*pi;
f=sin(x)-2*cos(2*x);
plot(x,f)
legend('sen(x)-2cos(2x)')

subplot(1,2,2)
x=0 : 0.001 : 4*pi;
g=sin(x);
plot(x,g,'r')
legend('sen(x)')
```



GRÁFICOS: 2D Y 3D

Otras funciones gráficas 2-D

- bar() crea diagramas de barras.
- barh() diagramas de barras horizontales.
- bar3() diagramas de barras con aspecto 3-D.
- bar3h() diagramas de barras horizontales con aspecto 3-D.
- pie() gráficos con forma de "tarta".
- pie3() gráficos con forma de "tarta" y aspecto 3-D.
- area() similar ***plot()***, pero rellenando en ordenadas de 0 a y.
- stairs() función análoga a ***bar()*** sin líneas internas.
- errorbar() representa sobre una gráfica –mediante barras– valores de errores.
- compass() dibuja los elementos de un vector complejo como un conjunto de vectores partiendo de un origen común.
- feather() dibuja los elementos de un vector complejo como un conjunto de vectores partiendo de orígenes uniformemente espaciados sobre el eje de abscisas.
- hist() dibuja histogramas de un vector.

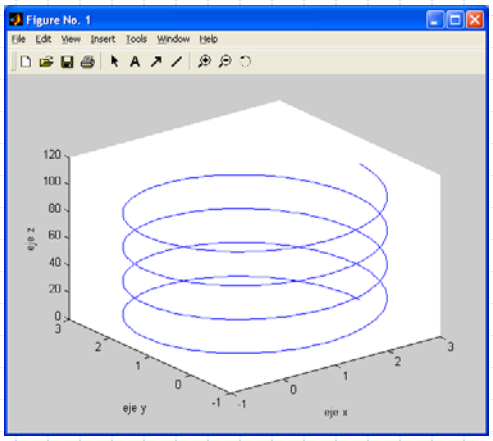
Funciones gráficas 3D elementales:

La función **plot3** es análoga a su homóloga bidimensional **plot**. Su forma más sencilla es la siguiente:

» plot3(x,y,z)

Ejemplo 4:

```
teta=0:pi/80:8*pi;
x=1+2*cos(teta);
y=1+2*sin(teta);
z=4*teta;
plot3(x,y,z)
axis([-1 3 -1 3 0 120]);
xlabel('eje x')
ylabel('eje y')
zlabel('eje z')
```

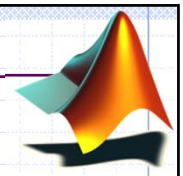


Manipulación de gráficos

view: view(azimut, elev), view([xd,yd,zd]).
view(2)
view(3)
rotate(h,d,a) o rotate(h,d,a,o) h es el objeto, d es un vector que indica la dirección y a un ángulo, o el origen de rotación

En el dibujo de funciones tridimensionales, a veces también son útiles los *NaN*. Cuando una parte de los elementos de la matriz de valores **Z** son *NaN*, esa parte de la superficie no se dibuja, permitiendo ver el resto de la superficie.

GRÁFICOS: 2D Y 3D



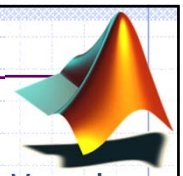
Transformación de coordenadas

```
[ang,rad]=cart2pol(x,y) %De cartesianas a polares
[ang,rad,z]=cart2pol(x,y,z) %De cartesianas a cilindricas

[x,y]=pol2cart(ang,rad) %De polares a cartesianas
[x,y,z]=pol2cart(ang,rad,z) %De cilindricas a cartesianas

[angx,angz,rad]=cart2sph(x,y,z) %De cartesianas a esfericas

[x,y,z]=aph2cart(angx,angz,rad) %De esfericas a cartesianas
```

Ejemplo 5:

```
%cilindricas
[ang,rad,z]=cart2pol(sqrt(3),1,2)
%esfericas
[ang1,ang2,rad1]=cart2sph(sqrt(3),1,2)
```


GRÁFICOS: 2D Y 3D



Ejercicio 5.1: Consideremos la ecuación de Van der Waals. Se considera el benceno para el cual a=18.78 atml^2/mol^2, b=0.1208 l/mol. Representar sobre una misma gráfica las dos subgráficas correspondientes a:

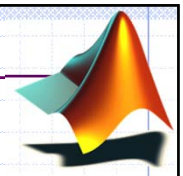
Isotermas de 100, 200, 300 y 400 ºC

Isobaras de 25, 35, 45 y 55 atm

Cada curva debe ir con trazo diferenciado, con el texto que indique la isolínea que se ha representado, así como el título de la gráfica y la etiqueta de los ejes. R=0.0821 y V=[2:100];

$$(P + \frac{a}{V^2})(V - b) = RT$$

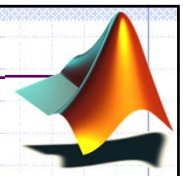
GRÁFICOS: 2D Y 3D



Solución Ejercicio 5.1:

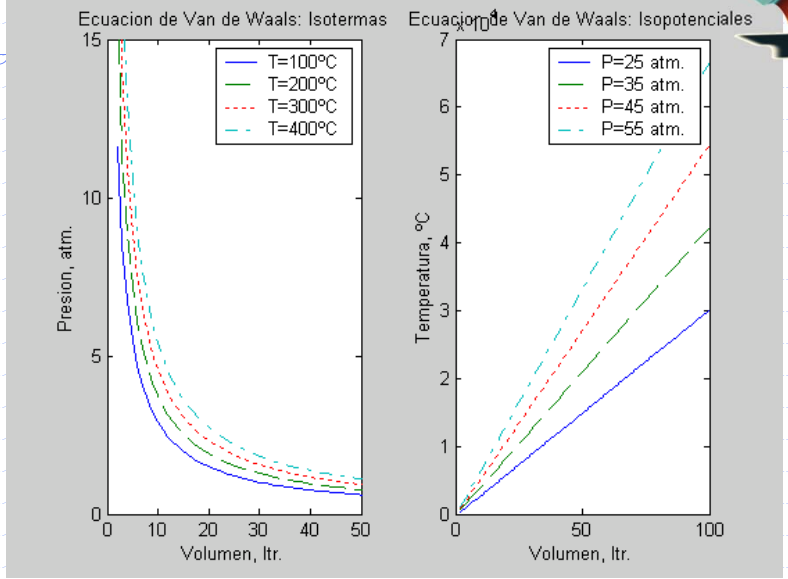
```
a=18.78;
b=0.1208;
R=0.0821;
subplot(1,2,1)
T=[373:100:673];
V=[2:100];
fac1=R./(V-b);
fac2=a./V.^2;
P=zeros(4,length(V));
P(1,:)=T(1)*fac1-fac2;
P(2,:)=T(2)*fac1-fac2;
P(3,:)=T(3)*fac1-fac2;
P(4,:)=T(4)*fac1-fac2;
plot(V,P(1,:),'-',V,P(2,:),'--',V,P(3,:),':',V,P(4,:),'-.')
title('Ecuacion de Van der Waals: Isotermas')
xlabel('Volumen, ltr.')
```


GRÁFICOS: 2D Y 3D



```
ylabel('Presion, atm.')
axis([0, 50,0,15])
legend('T=100ºC','T=200ºC','T=300ºC','T=400ºC')
subplot(1,2,2)
P=[25:10:55];
T=zeros(4,length(V));
fac1=(V-b)/R;
T(1,:)=((P(1)+fac2).*fac1)-273.15;
T(2,:)=((P(2)+fac2).*fac1)-273.15;
T(3,:)=((P(3)+fac2).*fac1)-273.15;
T(4,:)=((P(4)+fac2).*fac1)-273.15;
plot(V,T(1,:),'-',V,T(2,:),'--',V,T(3,:),':',V,T(4,:),'-.')
title('Ecuacion de Van der Waals: Isobaras')
xlabel('Volumen, ltr.')
ylabel('Temperatura, ºC')
legend('P=25 atm.','P=35 atm.','P=45 atm.','P=55 atm.')
```

GRÁFICOS: 2D Y 3D



GRÁFICOS: 2D Y 3D

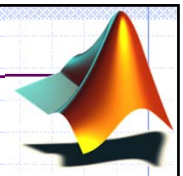
Ejercicio 5.2 Dada la función f(x,y)=xy, obtener sobre una ventana gráfica las representaciones siguientes:

-La superficie definida por la función sobre el dominio [-10,10]*[-10,10].

-Las líneas de contorno sobre la superficie

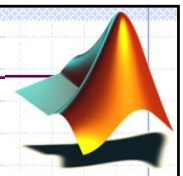
-La proyección de las líneas de contorno sobre el domino de definición

-La proyección de las líneas de contorno sobre el plano xy correspondientes a los valores –4,-1, 1 y 4.



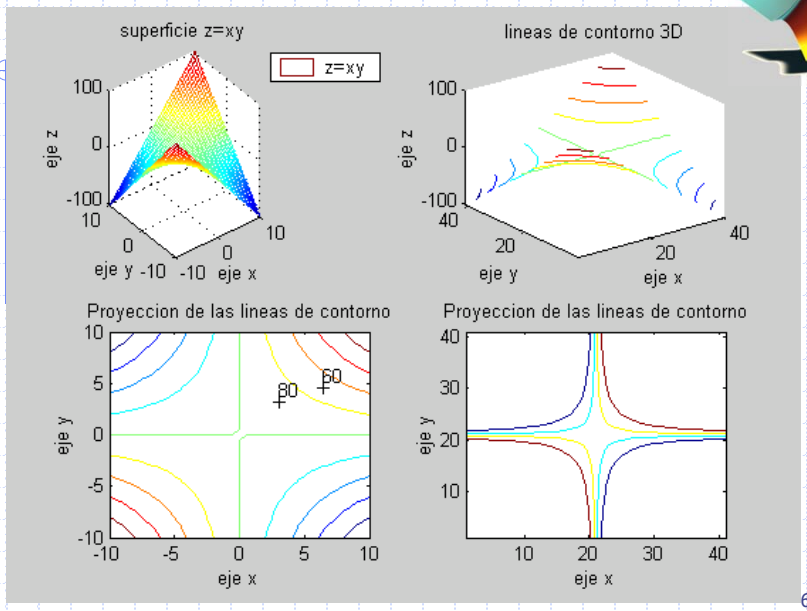
Solución Ejercicio 5.2

```
x=[-10:0.5:10]; y=x;
[X,Y]=meshgrid(x,y);
Z=X.*Y;
subplot(221)
mesh(X,Y,Z)
legend('z=xy')
xlabel('eje x')
ylabel('eje y')
zlabel('eje z')
title('superficie z=xy')
subplot(222)
contour3(Z)
grid off
xlabel('eje x')
ylabel('eje y')
```

```
zlabel('eje z')
title('lineas de contorno 3D')
subplot(223)
cs=contour(Z);
contour(x,y,Z)
grid off
clabel(cs)
xlabel('eje x')
ylabel('eje y')
title('Proyeccion de las lineas de contorno')
subplot(224)
contour(Z,[-4,-1,1,4])
grid off
xlabel('eje x')
ylabel('eje y')
title('Proyeccion de las lineas de contorno')
```

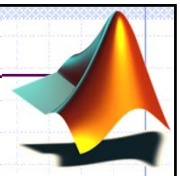
GRÁFICOS: 2D Y 3D



GRÁFICOS: 2D Y 3D

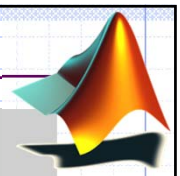
Ejercicio 5.3 Representar la superficie de revolución obtenida al girar la curva $y=x^2+1$ alrededor del eje ox

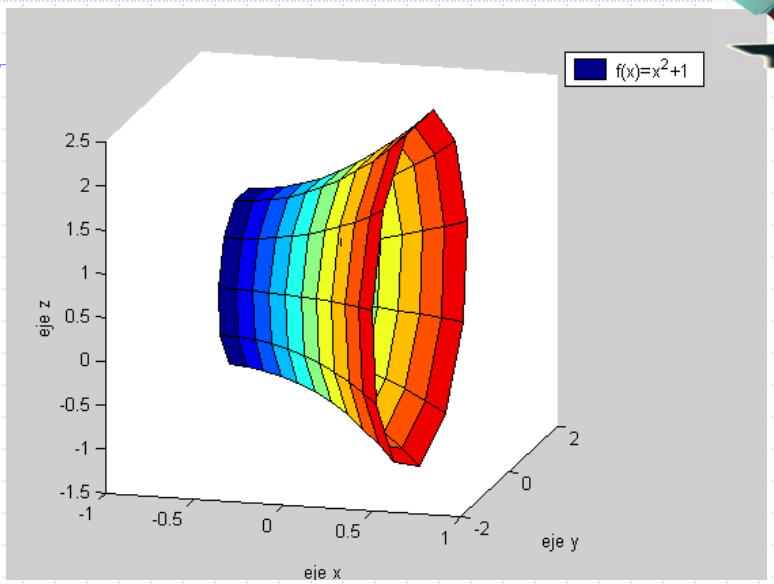
x=[0:0.1:1];



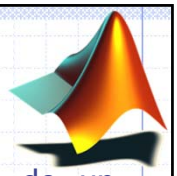
Solución Ejercicio 5.3

```
x=[0:0.1:1]; %puntos de discretizacion del eje ox
rad=x.^2+1; %vector de radios
n=length(rad); %numero de radios
cylinder(rad,n) %representacion del cilindro
xlabel('eje x')
ylabel('eje y')
zlabel('eje z')
[X,Y,Z]=cylinder(rad,n);
h=surf(X,Y,Z); %calculo del objeto
rotate(h,[0,1,0],90)
%Al rotar desaparecen las etiquetas de los ejes
xlabel('eje x')
ylabel('eje y')
zlabel('eje z')
view(15,15) %cambiamos el punto de observacion
grid off
legend('f(x)=x^2+1')
```



ANÁLISIS DE DATOS



Para realizar con Matlab análisis estadísticos de un conjunto de datos estos deben ser almacenados utilizando matrices. Cada columna de la matriz representará una variable medida y cada fila los valores que toman las variables consideradas en un determinado punto de medida.

Veamos algunos comandos interesantes:

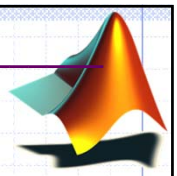
Cálculo del mínimo, máximo, media y mediana.

[y,k]=min(x)

[y,k]=max(x)

m=mean(y)

me=median(y) (*la mediana de un conjunto de números ordenados en magnitud es o el valor central o la media de los 2 valores centrales*).



Desviación típica.

s=std(y)

Nos da la dispersión o variación de los datos.

$$std(y) = \sqrt{\left(\frac{\sum_i (y(i) - \bar{y})^2}{N} \right)}$$

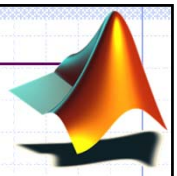
Coeficientes de correlación

r=corrcoef([x,y])

Nos da el grado de relación entre x e y.

$$r = \frac{\sum_{i=1}^{N} x(i)y(i)}{\sqrt{\left(\sum_{i=1}^{N} x(i)^2\right)\left(\sum_{i=1}^{N} y(i)^2\right)}}$$

ANÁLISIS DE DATOS



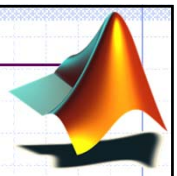
Matriz de covarianza

s=cov([x,y])

$$s = \frac{\sum_{i=1}^{N} x(i)y(i)}{N-1}$$

En el caso de matrices S=cov(X) nos daría la matriz de covarianza y su diagonal es el vector de varianzas (desviaciones típicas al cuadrado).

ANÁLISIS DE DATOS



Gráficos estadísticos.

Histograma:

>>hist(y)

Ejemplo:

```
y=rand(40,1);
hist(y)
```

Variantes: hist(y,n) (con n subintervalos de clases regulares); hist(y,x); [n,p]=hist(y)

Gráfico de barras:

bar(y)

Perfil de muestra:

stairs(y)

stem(x,y)

ANÁLISIS DE DATOS

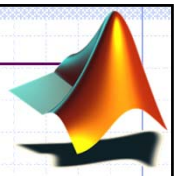


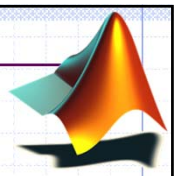
Gráfico de errores:
errorbar(x,y,e)

Histograma angular:
rose similar a hist pero realiza un histograma angular los valores de la muestra es de ángulos en radianes.

Curvas de regresión
p=polyfit(x,y,n)
Calcula el polinomio de regresión de grado n; es decir el polinomio p de grado n que minimiza. (ver ejercicio 7.2)

$$\sum_{i=1}^{N}\left(p(x_i)-y_i\right)^2$$

ANÁLISIS DE DATOS



Interpolación uni y bidimensional
.-1D

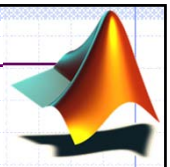
vector_y=interp1(x,y,vector_x,opcion)

opcion:
-'linear': interpolación lineal
-'cubic' :interpolación cúbica
-'spline': interpolación spline cúbica (ptos de interpolación igualmente espaciados).

.-2D

matriz_Z=interp2(X,Y,Z,matriz_X,matriz_Y,opcion)

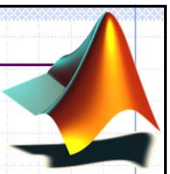
opcion:
-'bilinear': interpolación lineal
-'bicubic' :interpolación cúbica
-'nearest'



Ejercicio 6.1:

La tabla siguiente recoge el peso de 30 estudiantes.
Construir sobre una misma ventana las 4 figuras siguientes:
1.-Un histograma de frecuencias con el ox peso y oy frecuencia de valores.
2.-El polígono de frecuencias, curva obtenida entre los puntos definidos por las marcas de clase y la frecuencia.
3.-El perfil de muestra mediante stairs
4.-El perfil de muestra mediante stem.
Calcular el máximo y mínimo peso y la media de pesos.

71	82	65	75	77	91	59	84	89	81
73	91	82	75	96	85	69	76	81	92
84	79	77	95	81	79	84	85	76	82



Solución Ejercicio 6.1:

```
peso=[71 82 65 75 77 91 59 84 89 81 ...
      73 91 82 75 96 85 69 76 81 92 ...
      84 79 77 95 81 79 84 85 76 82];
figure
subplot(2,2,1)
hist(peso)
title('Histograma')
xlabel('Peso')
ylabel('Frecuencia')
subplot(2,2,2)
[n,p]=hist(peso);
plot(p,n)
title('poligono de frecuencias')
xlabel('Peso')
ylabel('Frecuencia')
```

ANÁLISIS DE DATOS

```
subplot(2,2,3)
stairs(peso)
title('Stairs')
ylabel('Peso')
xlabel('Individuo')
subplot(224)
stem(peso)
title('Stem')
ylabel('Peso')
xlabel('Individuo')
maximo=max(peso)
minimo=min(peso)
media=mean(peso)
```

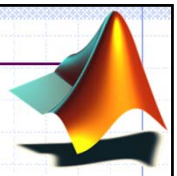

ANÁLISIS DE DATOS

Ejercicio 6.2

Se ha medido experimentalmente la conductividad eléctrica del acero a distintas temperaturas, recogiéndose los siguientes valores:

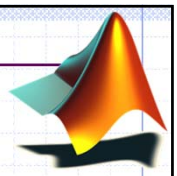
T(ºC)	K$(\Omega cm)^{-1}$
100	51813
300	28571
500	17483
700	11696
900	9116

Obtener los polinomios de regresión de primero y segundo grado. Calcular para los polinomios obtenidos cuál sería el valor estimado de la conductividad eléctrica a los 600ºC y a los 1000ºC.



Solución Ejercicio 6.2

```
temp=100:200:900;
conduc=[51813 28571 17483 11696 9116];
pol1=polyfit(temp,conduc,1)
temp1=0:50:1000;
conduc1=polyval(pol1,temp1)
conduc1e=polyval(pol1,temp);
plot(temp1,conduc1,temp,conduc1e,'o',temp,conduc,'*')
legend('recta de regresion','conductividades estimadas',...
'conductividades medidas')
xlabel('temperatura ºC')
ylabel('Conductividad electrica (ohm cm)^{-1}')
cond1_600=polyval(pol1,600)
cond1_1000=polyval(pol1,1000)
```

```
pol2=polyfit(temp,conduc,2)
temp2=0:50:1000;
conduc2=polyval(pol2,temp2)
conduc2e=polyval(pol2,temp);
figure
plot(temp2,conduc2,temp,conduc2e,'o',temp,conduc,'*')
legend('recta de regresion','conductividades estimadas',...
'conductividades medidas')
xlabel('temperatura ºC')
ylabel('Conductividad electrica (ohm cm)^{-1}')
cond2_600=polyval(pol2,600)
cond2_1000=polyval(pol2,1000)
corre1=corrcoef([conduc',conduc1e'])
corre2=corrcoef([conduc',conduc2e'])
```

AYUDA

La ayuda es muy importante en matlab:
- help nombre_comando
- help nombre_toolbox

Algunas importantes son:
- help graph2d
- help graph3d
- help specgraph
- help save

Podemos ver ejemplos hechos con matlab poniendo
- demo

Para saber más: http://www.mathworks.com/
http://www.mathworks.com/access/helpdesk/help/toolbox/compiler/compiler.shtml
Esta última hace referencia a los compiladores de c de matlab, los mex files