MAESTRÍA EN INGENIERÍA Y GESTIÓN AMBIENTAL

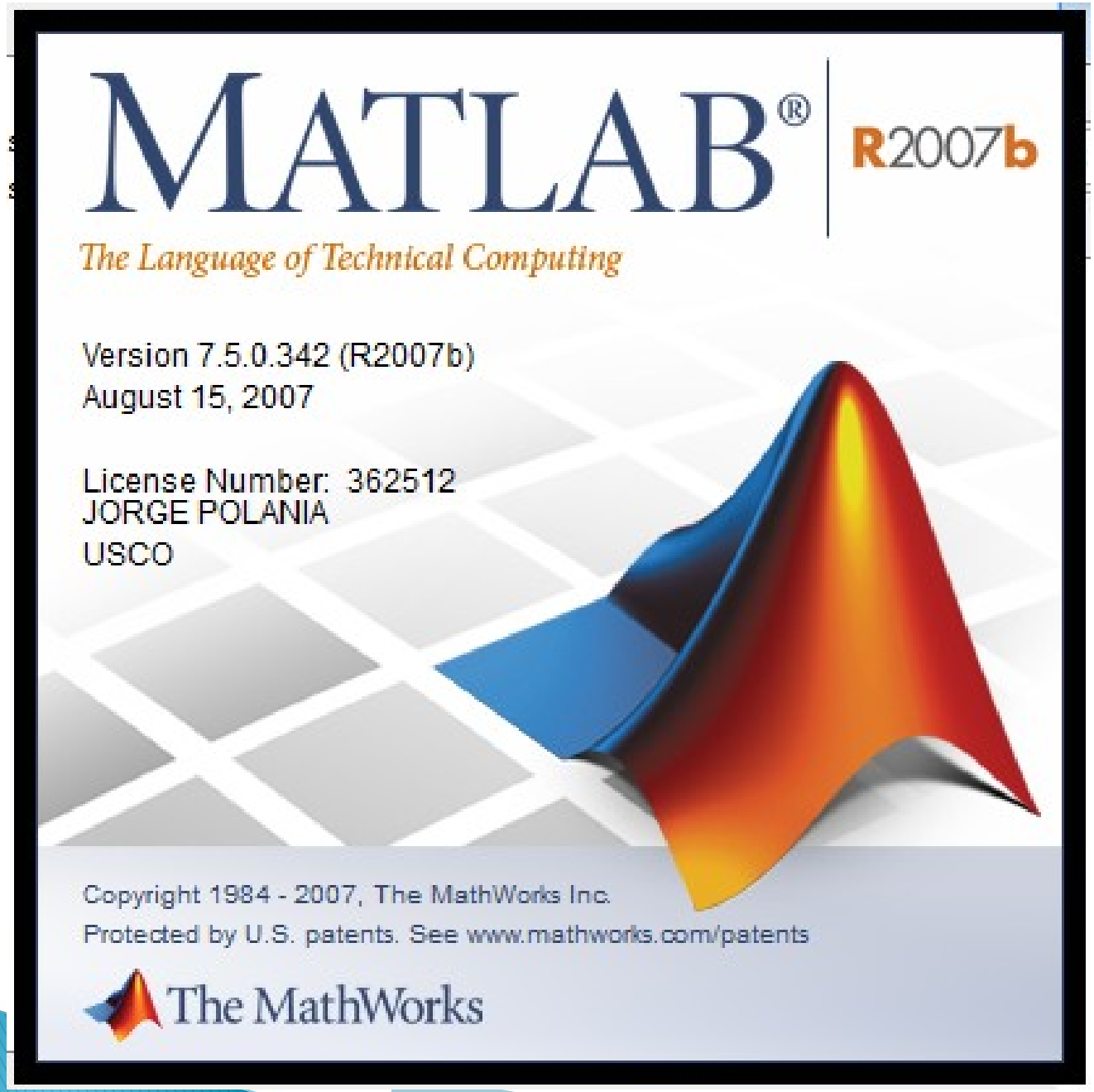
CURSO: SIMULACIÓN DE SISTEMAS AMBIENTALES

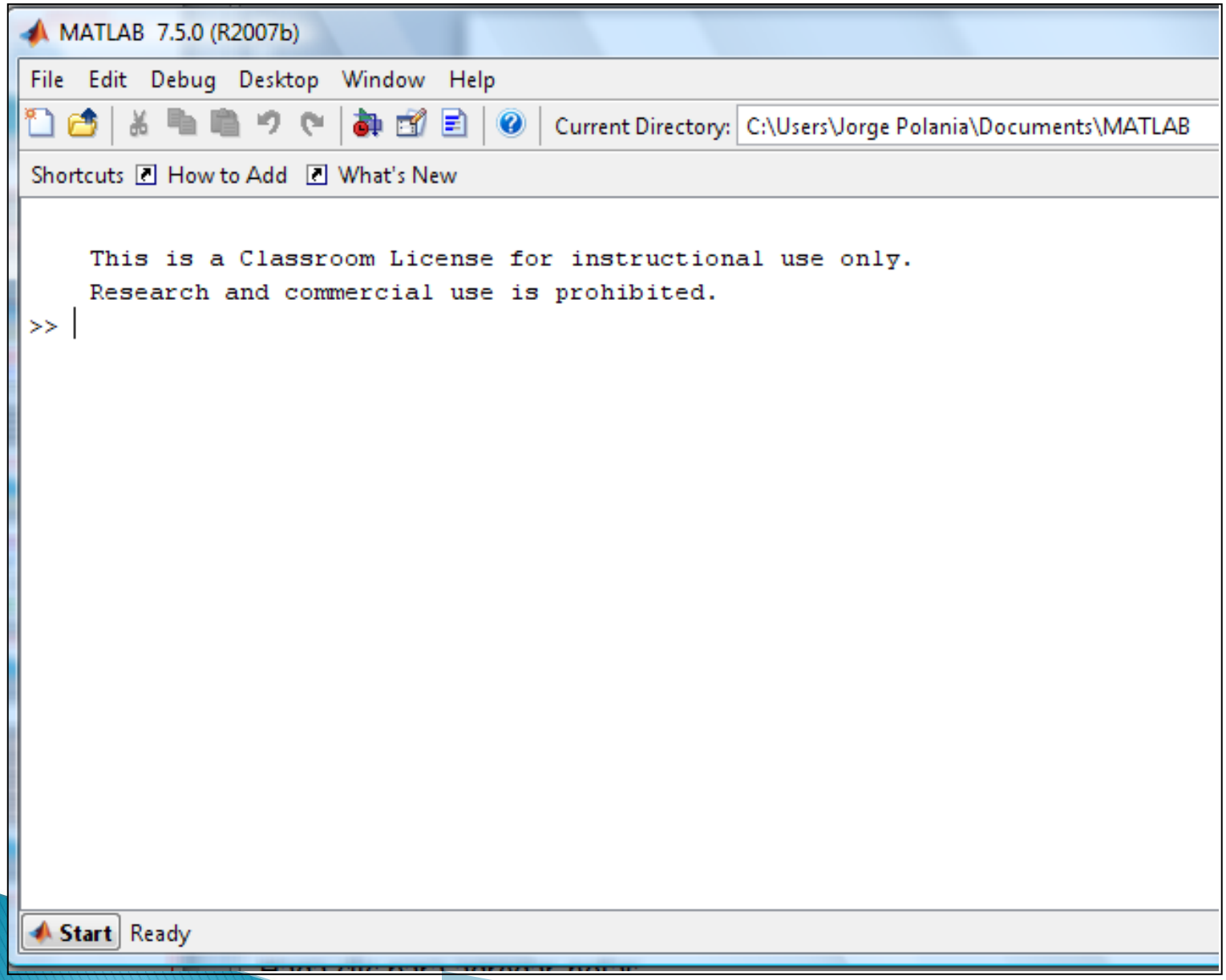
PROFESOR: M.I. Jorge Antonio Polanía P.

CONTENIDO

MÓDULO 2:
PROCESOS CON MATLAB

1. POLINOMIOS
2. GRAFICACIÓN
3. CÁLCULO NUMÉRICO
4. DINÁMICA DE SISTEMAS


MATLAB®
R2007b
The Language of Technical Computing
Version 7.5.0.342 (R2007b)
August 15, 2007
License Number: 362512
JORGE POLANIA
USCO
Copyright 1984 - 2007, The MathWorks Inc.
Protected by U.S. patents. See www.mathworks.com/patents
The MathWorks


MATLAB 7.5.0 (R2007b)
File Edit Debug Desktop Window Help
Current Directory: C:\Users\Jorge Polania\Documents\MATLAB
Shortcuts How to Add What's New
This is a Classroom License for instructional use only.
Research and commercial use is prohibited.
>>
Start
Ready

1. POLINOMIOS

La integración de las Tecnologías de Información y comunicación (TIC) en las asignaturas de un currículo puede realizarse de varias formas. Una de ellas es el uso de las simulaciones. Estas se han convertido en una excelente herramienta para mejorar la compresión y el aprendizaje en áreas como las matemáticas, física, estadística, finanzas, etc. La simulación permite probar, analizar y descubrir cómo funciona o cómo se comporta un fenómeno.

Matlab es un programa interactivo de cálculo numérico y de visualización de datos basado en software de matrices, en un entorno de desarrollo totalmente integrado y orientado a proyectos que requieren un elevado cálculo numérico y visualización gráfica.

En las universidades Matlab se ha convertido en una herramienta básica tanto para estudiantes, como para docentes e investigadores por su amplio abanico de programas especializados llamados Toolboxes que cubren casi todas las áreas del conocimiento. Dispone de un programa SIMULINK que es un entorno gráfico interactivo con el que se puede analizar, modelar y simular sistemas.

1.1 VARIABLES Y FUNCIONES

OPERADORES

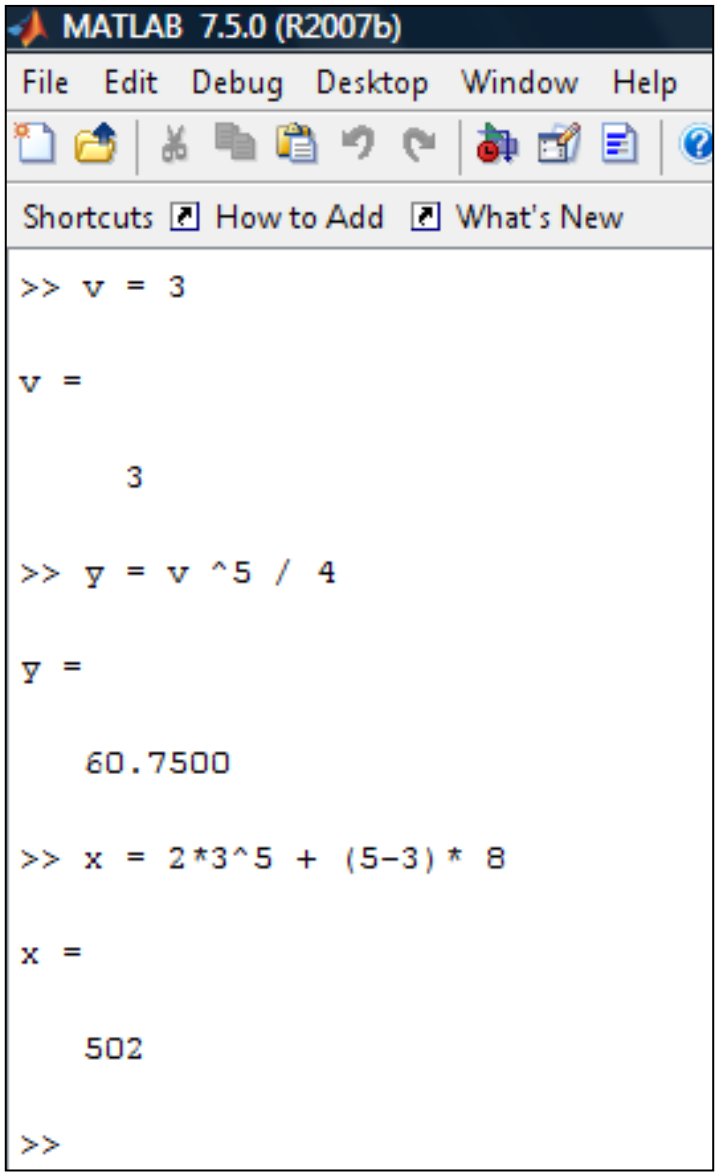
Una variable se crea por asignación.

Los operadores básicos son:

x +y	Suma
x – y	Diferencia
x * y	Producto
x / y	División
x ^y	Potencia

Ejemplos:

En la ventana de comandos de Matlab, ejecutar:



VECTORES

Un *vector fila* de n elementos se puede representar de dos formas:

V = [v1,v2,v3,…..vn] % con coma entre ellos, o
V = [v1 v2 v3 …..vn] % con espacios entre ellos

Ejemplo:

Vector = [1 1.2 3.4 4/5 2.25]

Un vector se puede representar sin necesidad de explicitar todos los elementos,

EXPRESIÓN MATLAB	SIGNIFICADO
Vector = [a : b]	a y b son el primero y último elemento. Los elementos intermedios se diferencian en una unidad
Vector = [a : s : b]	a y b son el primero y último elemento. Los elementos intermedios se diferencian en la cantidad s
Vector = linspace[a,b,n]	a y b son el primero y último elemento. Hay n elementos uniformemente espaciados entre sí
Vector = logspace[a,b,n]	a y b son el primero y último elemento. Hay n elementos logarítmicamente espaciados entre sí

Ejemplos:

```
>>Vector1 = [5:5:30]                % elementos de 5 a 30 en pasos de 5
Vector1 = 5   10   15   20   25   30

>>Vector2 = [5:10]
Vector2 = 5  6  7  8  9  10        % elementos de 5 a 10 en pasos de 1 (por defecto)
```

Un *vector columna* se representa con sus elementos separados por punto y coma.

Ejemplo:

```
>>Vector = [2; 3; 2.5; 4.5; 8]

Vector =
            2
            3
            2.5
            4.5
            8
```

1.2 MATRICES

Las matrices se representan en Matlab introduciendo entre corchetes los vectores fila separados por punto y coma.

Ejemplo:

```
>>A = [1 3 5; 4 7 9; 4 2 10]
A =

1   3   5
4   7   9
4   2   10
```

Algunas definiciones de variables matriciales:

A(m,n)	Define el elemento (m,n) de la matriz A
B = A'	Define la transpuesta de A
A(a:b,c:d)	Define una submatriz formada por las filas que hay entre la a-ésima y la b-ésima y por las columnas que hay entre la c-ésima y la d-ésima
A(:,c:d)	Submatriz formada por las filas de A y las columnas que hay entre la c-ésima y d-ésima
A(a:b,:)	Submatriz formada por las columnas de A y las filas que hay entre la a-ésima y b-ésima
size(A)	Devuelve el tamño u orden de la matriz A
inv(A)	Calcula la inversa de la matriz A (cuadrada)
det(A)	Determinante de A (cuadrada)

```
>> B = A'

B =

    1    4    4
    3    7    2
    5    9   10
```

```
>> eye(3)

ans =

    1    0    0
    0    1    0
    0    0    1
```

```
>> C=B(:,2:3)

C =

    4    4
    7    2
    9   10
```

```
>> D = B(1:2,:)

D =

    1    4    4
    3    7    2
```

```
>> size(D)

ans =

    2    3
```

OPERACIONES CON MATRICES

Las operaciones básicas son: suma, resta, multiplicación y división. La suma, resta y división sólo se puede hacer si ambas matrices tienen la misma dimensión. Por ejemplo,

```
>> A=[3 -6 2;4 7 -2]
B=[2 -5  12;5 3 1]
C=A+B
D=A-B
```

```
A =

     3    -6     2
     4     7    -2
```

```
B =

     2    -5    12
     5     3     1
```

```
C =

     5   -11    14
     9    10    -1
```

```
D =

     1    -1   -10
    -1     4    -3
```

La multiplicación de matrices se realiza si el número de columnas de la primera es igual al número de filas de la segunda. Ejemplo,

```
>> E=[2 -7;3 4;5 7]
C=A*E
D=B*E

E =

     2    -7
     3     4
     5     7

D =

    49    50
    24   -16

>> A*B
??? Error using ==> mtimes
Inner matrix dimensions must agree.
```

```
C =

    -2   -31
    19   -14

>> A.*B

ans =

     6    30    24
    20    21    -2
```

En división las matrices deben tener la misma dimensión. Ejemplo,

```
A =

     3    -6     2
     4     7    -2
```

```
>> B

B =

     2    -5    12
     5     3     1
```

```
>> A/B

ans =

    0.3508   -0.0987
   -0.3427    1.1828
```

```
>> A./B

ans =

    1.5000    1.2000    0.1667
    0.8000    2.3333   -2.0000
```

Multiplicación y división por un escalar.

```
A =

     3    -6     2
     4     7    -2
```

```
>> c

c =

     3
```

```
>> A/c

ans =

    1.0000   -2.0000    0.6667
    1.3333    2.3333   -0.6667
```

```
>> A*c

ans =

     9   -18     6
    12    21    -6
```

1.3 FUNCIONES

FUNCIONES TRIGONOMÉTRICAS	
Directas	**Inversas**
sin(x)	**asin(x)**
cos(x)	**acos(x)**
tan(x)	**atan(x)**
csc(x)	**acsc(x)**
sec(x)	**asec(x)**
cot(x)	**acot(x)**

FUNCIONES HIPERBÓLICAS	
sinh(x)	**asinh(x)**
cosh(x)	**acosh(x)**
tanh(x)	**atanh(x)**
csch(x)	**acsch(x)**
sech(x)	**asech(x)**
coth(x)	**acoth(x)**

FUNCIONES EXPONENCIALES Y LOGARÍTMICAS	
exp(x)	Función exponencial base e
log10(x)	Logaritmo decimal
log(x)	Logaritmo natural
sqrt(x)	Raíz cuadrada
abs(x)	Valor absoluto

NÚMEROS COMPLEJOS	
abs(z)	Módulo del complejo z
angle(z)	Argumento del complejo z
conj(z)	Conjugado del complejo z
real(z)	Parte real del complejo z
imag(z)	Parte imaginaria del complejo z
factorial(n)	n! = n(n-1)(n-2)(n-3)…..3.2.1

Ejemplos:

Calcular las siguientes expresiones en Matlab

a) $y = e^{\sqrt{x^2+2x-5}}$ para x = 2.5

```
>> x=2.5;
>> y = exp(sqrt(x^2+2*x-5))

y =
    12.1825
```

b) y = 2sen(5x) + 3cos(2x) para x = 30º

```
>> x = 30*pi/180;

>> y = 2*sin(5*x) + 3*cos(2*x)

y =
    2.5000
```

c) $y = \log \sqrt[3]{x+5} + \ln(x^2)$

Para x=1.5

```
>> x=1.5;
>> y = log10(x + 5)^(1/3) + log(x^2)
y =
    1.7442
```

1.4 POLINOMIOS

Los comandos usados por Matlab para trabajar con polinomios son:

p = poly(r)	Da los coeficientes del polinomio P cuyas raíces son el vector r
y = polyval(p,x)	Evalúa el polinomio p en el valor de x
r = roots(c)	Encuentra las raíces del polinomio c
p = polyfit(x,y,n)	Polinomio de orden n que ajusta los puntos (x,y)
s = solve('ecuacion1','ecuacion2')	Resuelve las ecuaciones
d = det(A)	Calcula determinante de A

Ejemplos:

a) >> p=poly([2 3 4])

p =

1 -9 26 -24

% El polinomio es $x^3 – 9x^2 +26x – 24$

b) Para x = 2.5 calcular
$y = x^4 – 3x^2 + 5x -2.8$

```
>> x = 2.5;
>> p = [1 0 -3 5 -2.8];
>> y = polyval(p,x)

y =
    4.0750
```

c) Encontrar las raíces de:
$x^5 – 3x^3 + x^2 -5x + 2$

```
>> c = [1 0 -3 1 -5  2];
>> r = roots(c)

r =
  -2.1716
   1.8905
  -0.0575 + 1.1076i
  -0.0575 - 1.1076i
   0.3960
```

d) Calcular el polinomio interpolador de segundo orden que pasa por los puntos (-1,4), (0,2) y (1,6)

```
>> x = [-1,0,1]; y = [4,2,6];
>> p = polyfit(x,y,2)
 p =
    3.0000    1.0000    2.0000
```

El polinomio interpolador que más se ajusta es $3x^2 + x + 2$

e) Calcular $\sqrt{1-x} + \sqrt{1+x} = 4$

```
>> s = solve('sqrt(1-x)+sqrt(1+x)=4')

s =
   4*i*3^(1/2)
  -4*i*3^(1/2)

>> eval(s)

s(1)=6.9281i          s(2)=-6.9281
```

f) Resolver el sistema de ecuaciones:

$$2x + 3y = 5$$
$$x - 2y = -2$$

```
>> [x,y] = solve('2*x + 3*y = 5','x - 2*y = -2')

%  x = 4/7,  y =  9/7
```

g) Calcular el determinante de la matriz:

$$\begin{bmatrix} 2 & 4 & -1 \\ 3 & -2 & 5 \\ -1 & 3 & 6 \end{bmatrix}$$

```
>> A = [2 4 -1; 3 -2 5; -1 3 6];
>> d = det(A)

%  d =  -153
```

2. GRAFICACIÓN

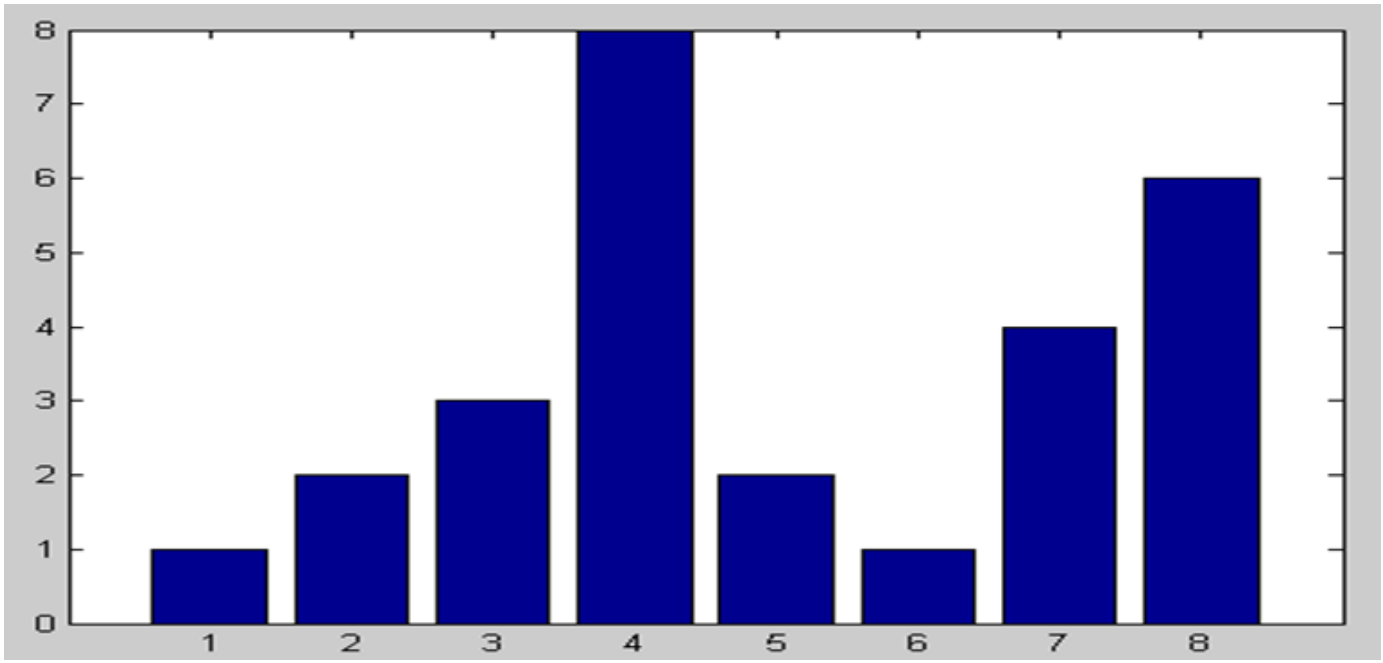
Matlab ofrece diversas formas de representación gráfica.

COMANDO MATLAB	DESCRIPCIÓN
bar(y)	Gráfica de barras relativo al vector y
bar(x,y)	Gráfica de barras al vector y; x define el eje x
barh(….)	Gráfica de barras horizontales
bar(….;'color')	Color = r, g, y, c, m, k
bar(y,'estilo')	Estilo=grouped (agrupado), stacked (anidado)
bar3(y,…)	Barras en tres dimensiones
plot(x,y)	Grafica y en función de x
plot(x,y,'bo')	Grafica y en función de x on color y caracter

fplot('f',[x1 x2],'y*')	Grafica función f entre x1 y x2
fplot('[f1,f2,..]',[x1 x2])	Grafica las funciones en el intervalo dado
tittle('texto')	Título de la gráfica
xlabel('texto'), ylabel('texto')	Rótulos en el eje x y en el eje y
grid	Pone rejilla en la gráfica
axis([x1 x2 y1 y2])	Define límite de los ejes
legend('rotulo1','rotulo2',….)	Coloca legenda en la gráfica
text(x,y,'texto')	Coloca texto en coordenadas (x,y)
subplot(m,n,p)	Subgráficas de m filas, n columnas

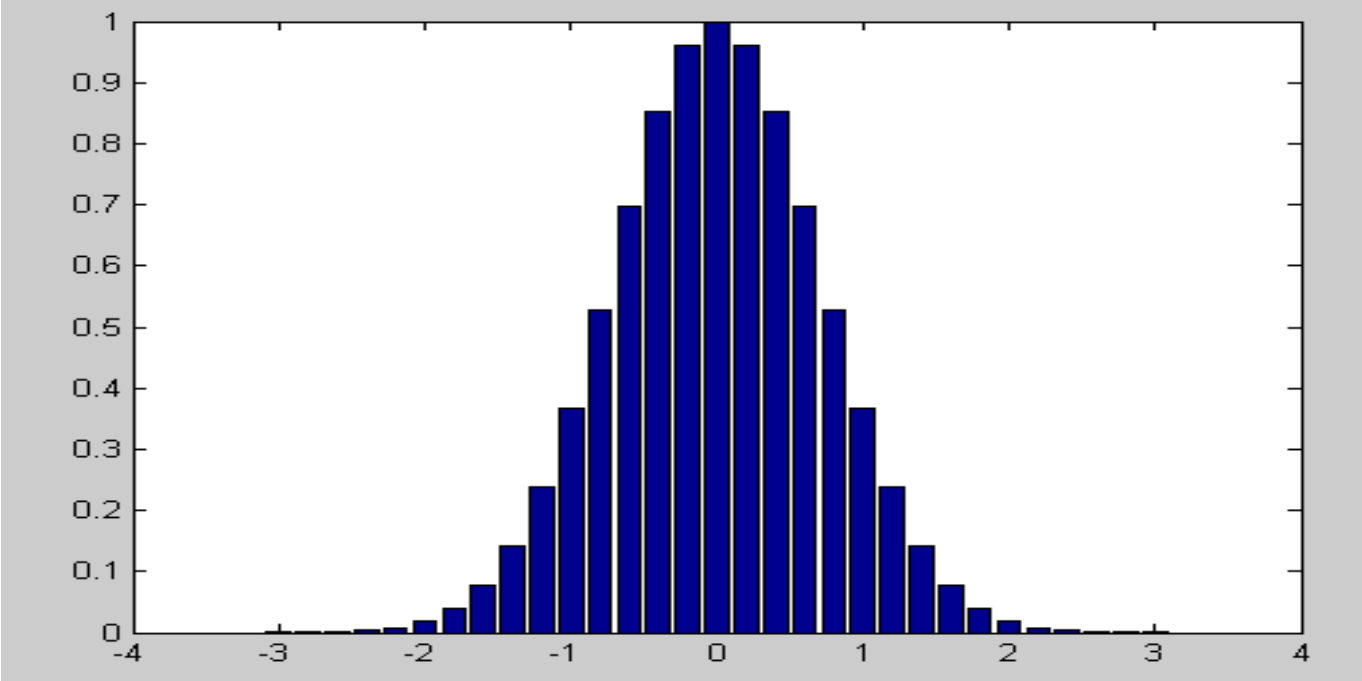
Ejemplos:

a) >> y=[1 2 3 8 2 1 4 6];
>> bar(y)

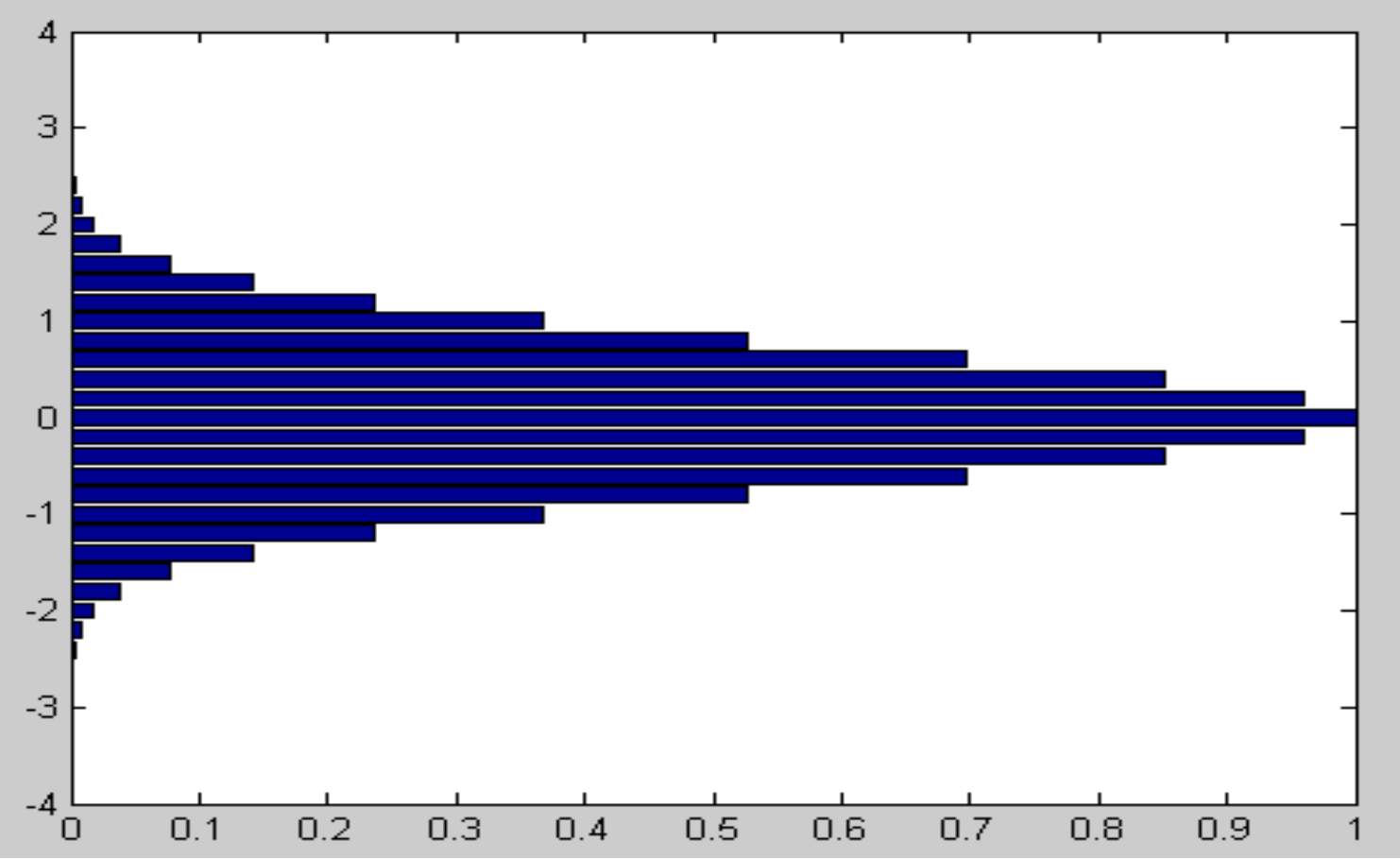


b) Gráfico de barras para la función $y = e^{x^2}$ cuando x varía de -3 a 3 en pasos de 0.2

```
>> x = -3:0.2:3;
>> y = exp(-x.*x);
>> bar(x,y)
```

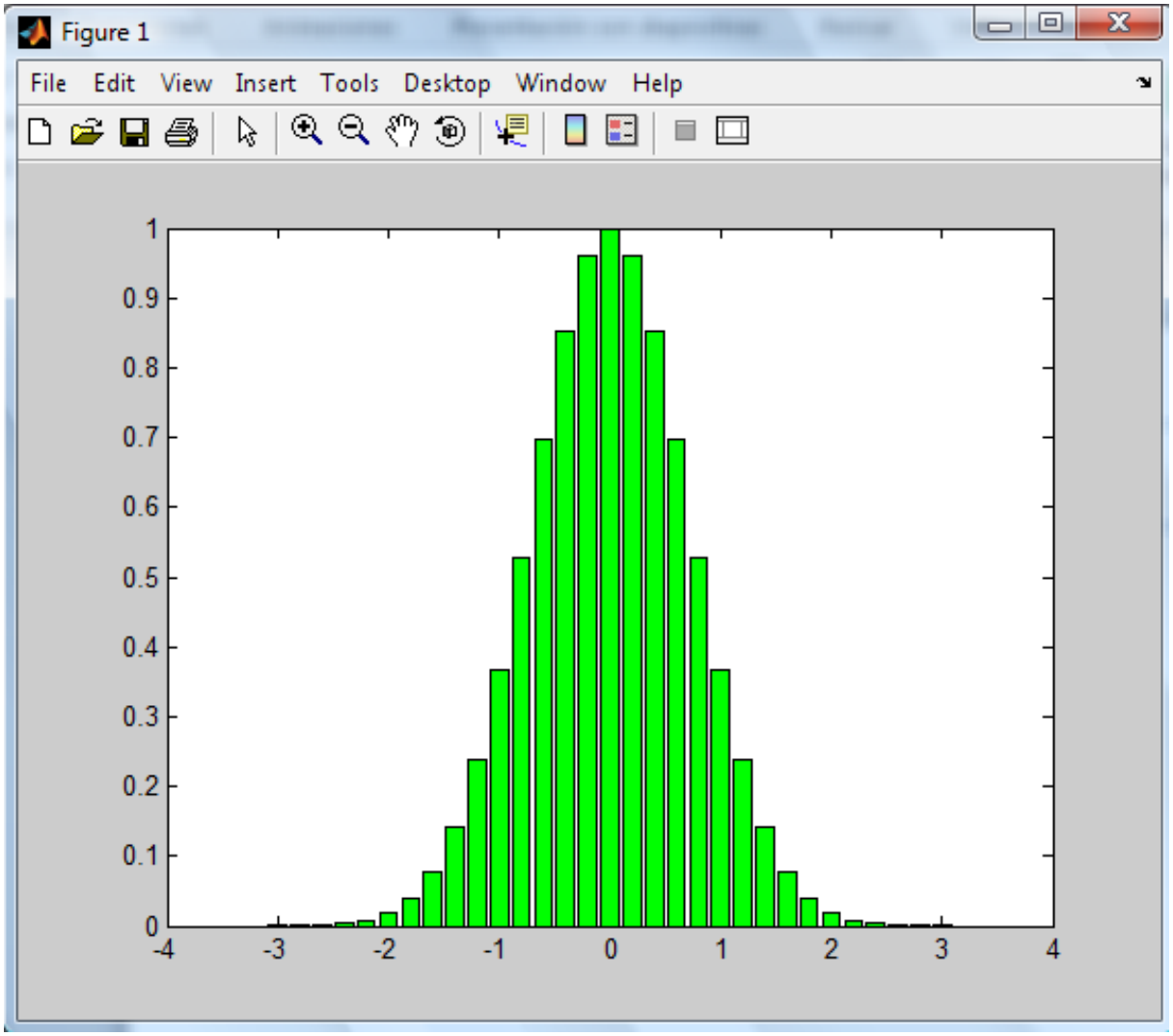


c) barh(x,y)



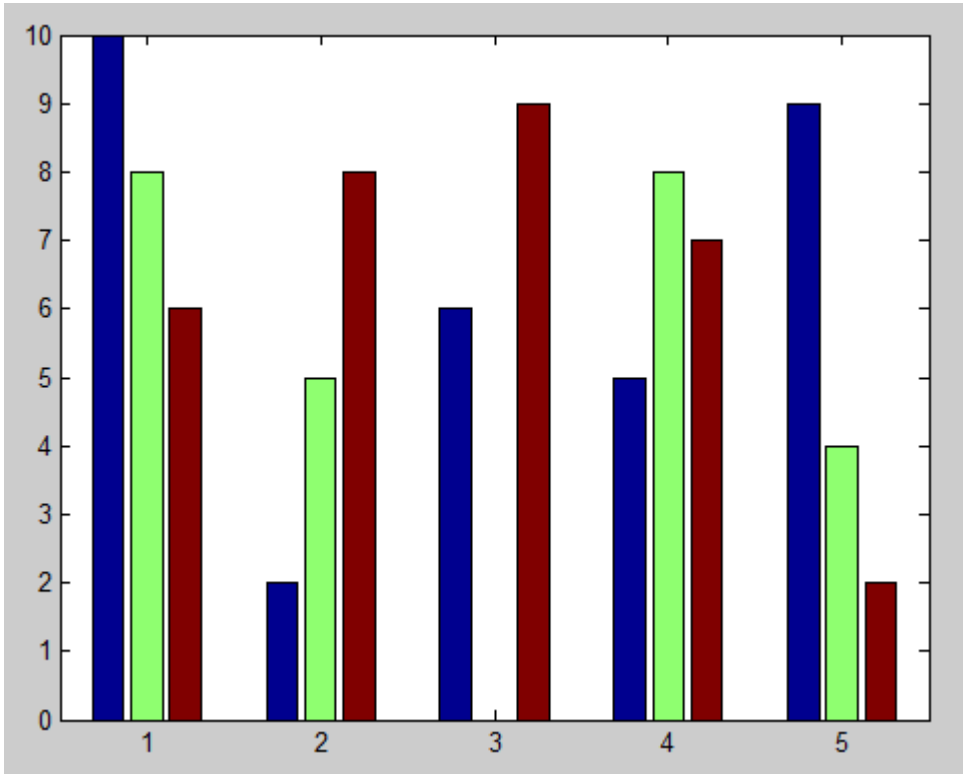
d) Ejecutar

>> bar(x,y,’g’)

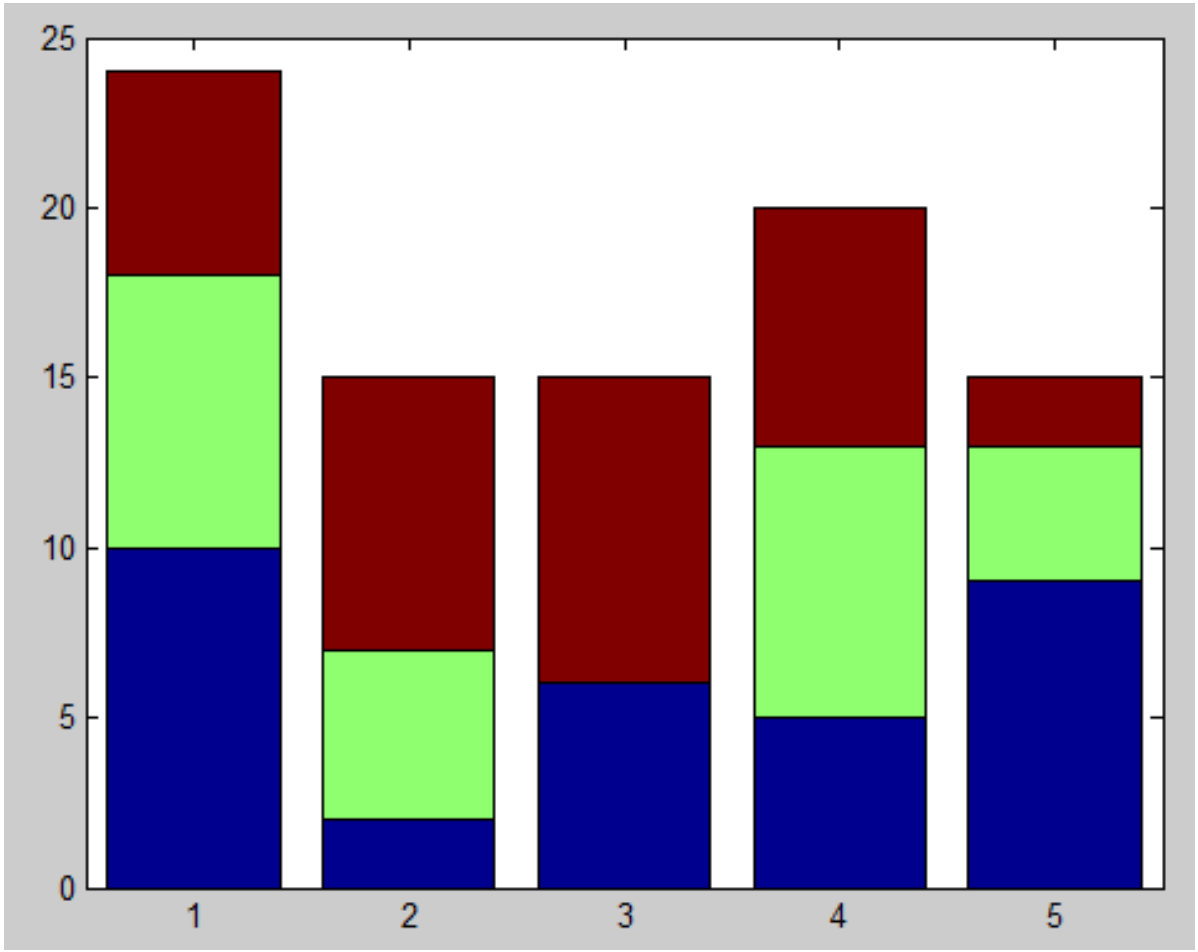


y =[10 8 6; 2 5 8; 6 0 9; 5 8 7; 9 4 2]

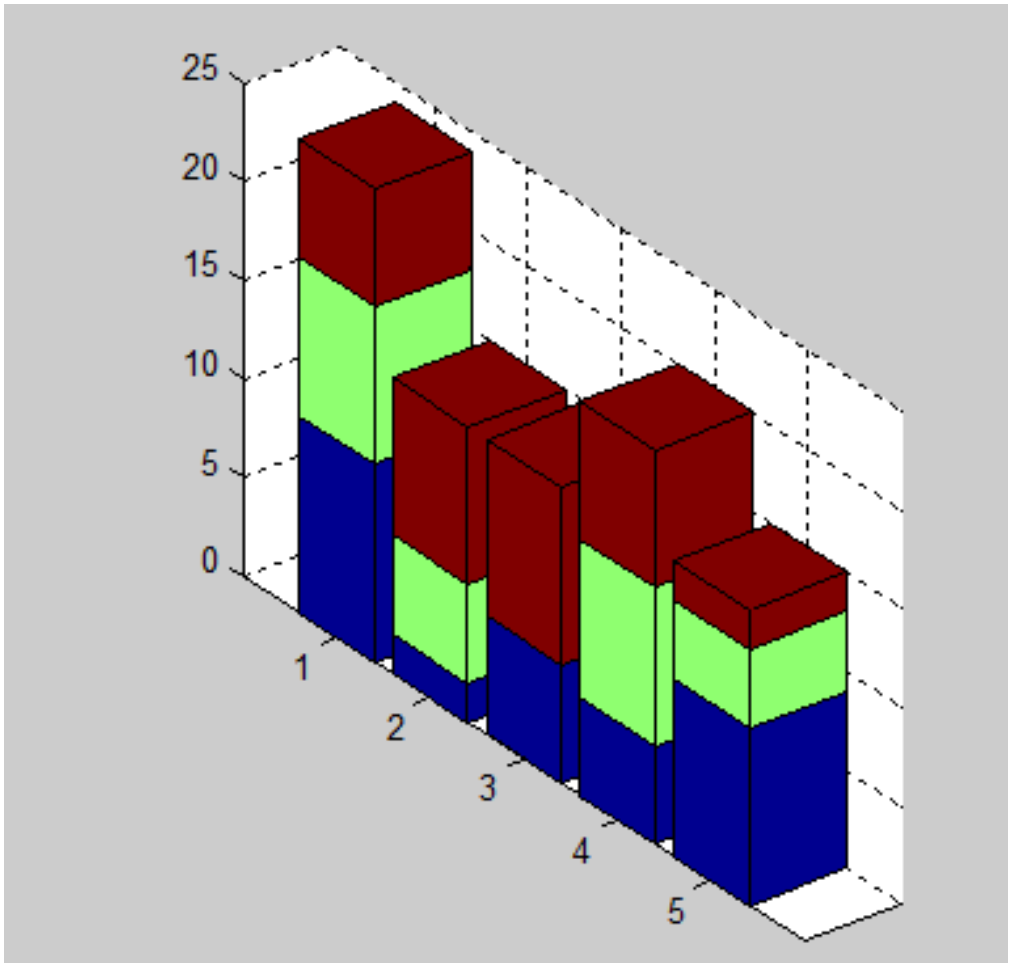
```
y =

    10     8     6
     2     5     8
     6     0     9
     5     8     7
     9     4     2

>> bar(y,'grouped')
```



```
>> bar(y,'stacked')
```



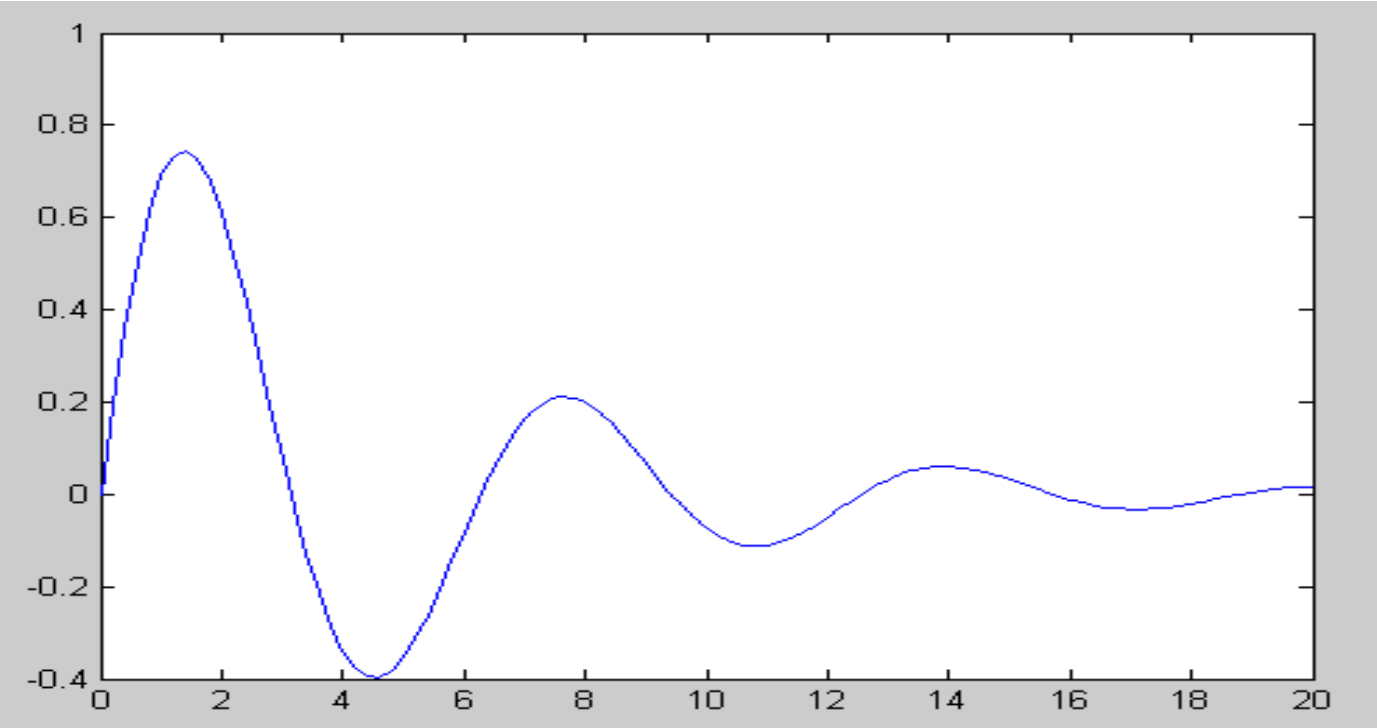
```
>> bar3(y,'stacked')
```



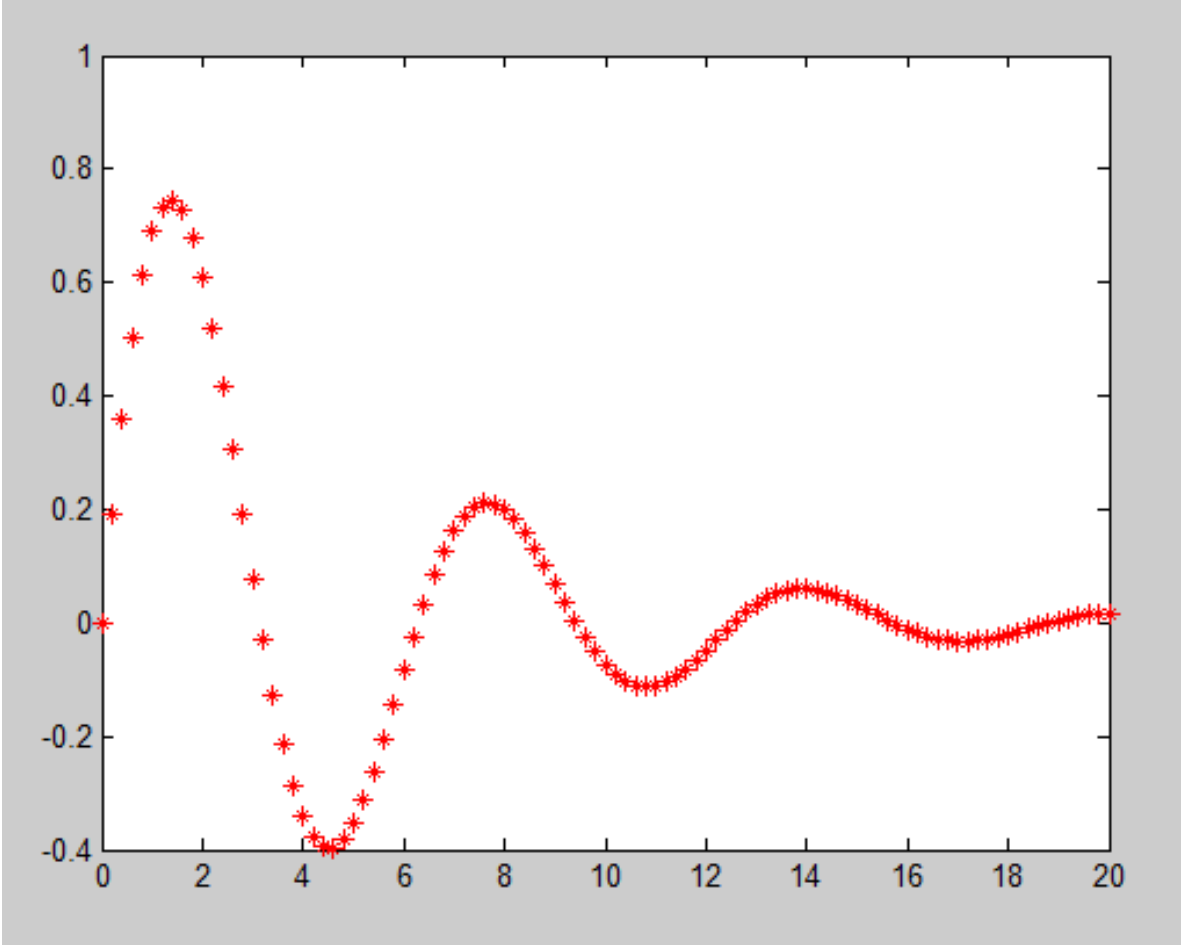
f) Ejecutar >> x = 0:0.2:20;

>> y = sin(x).*exp(-0.2*x);

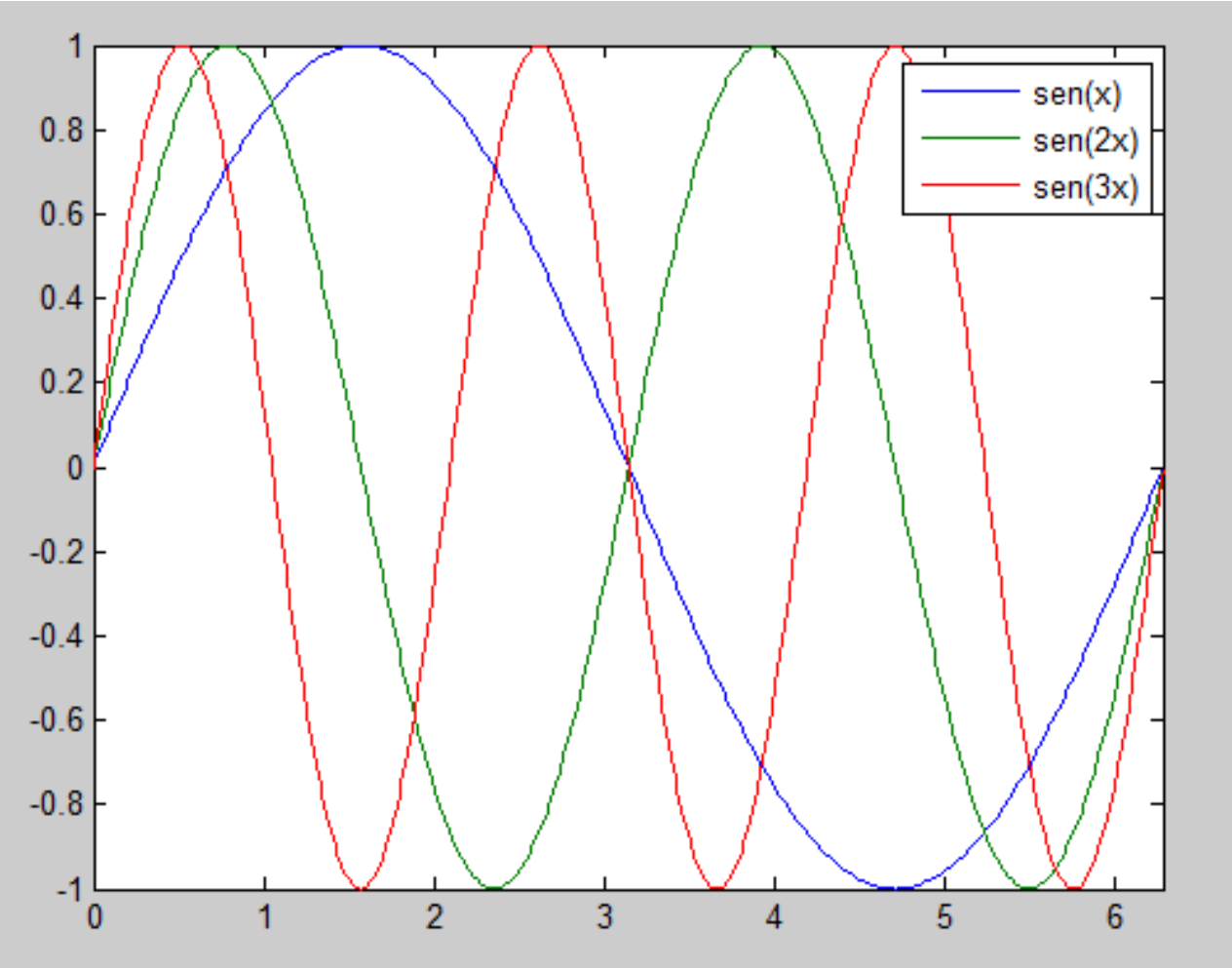
>> plot(x,y)



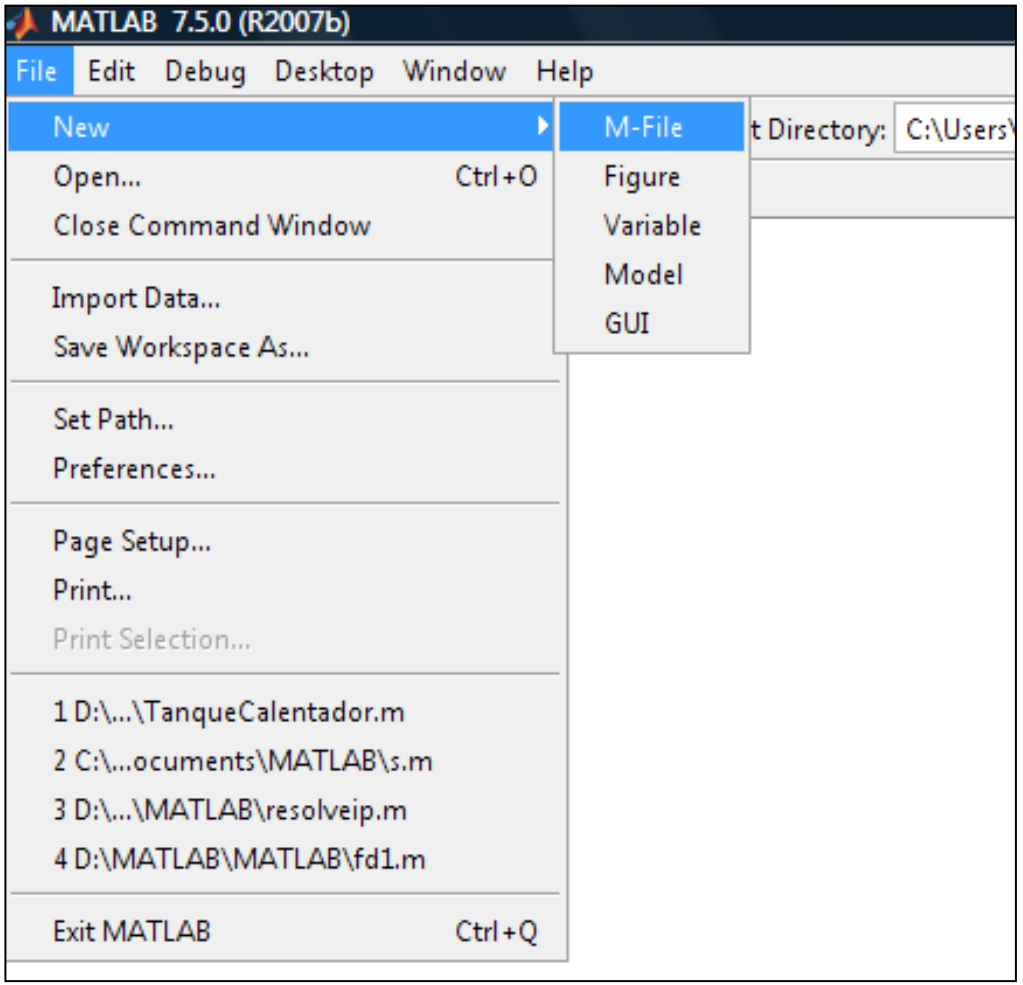
>> plot(x,y,'r*')

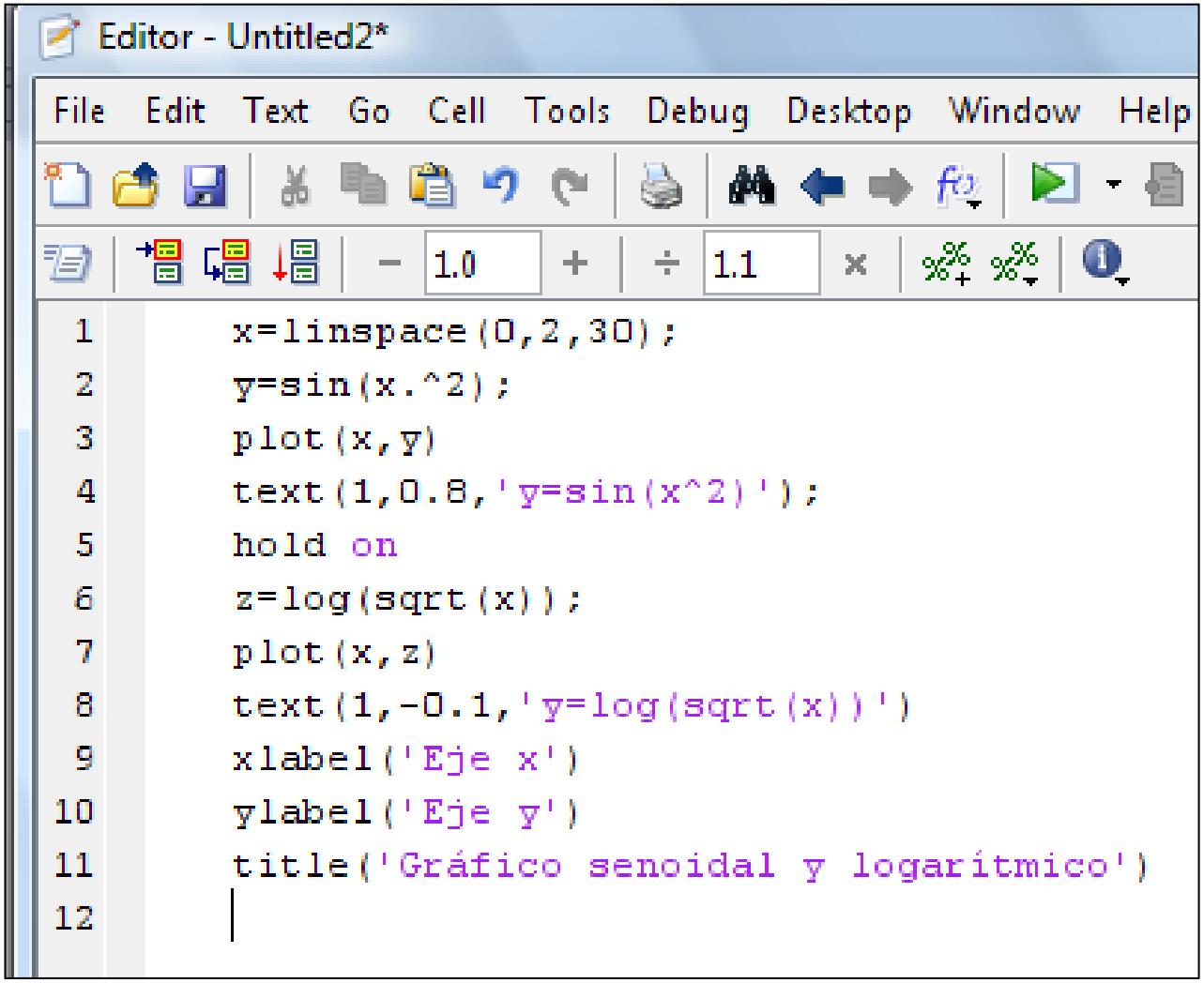


g) Ejecutar >> fplot('[sin(x), sin(2*x), sin(3*x)]',[0,2*pi])
>> legend('sen(x)','sen(2x)','sen(3x)')



h) Ejecutar en la ventana de edición el programa:




Editor - Untitled2*
File Edit Text Go Cell Tools Debug Desktop Window Help
1.0
1.1
x=linspace(0,2,30);
y=sin(x.^2);
plot(x,y)
text(1,0.8,'y=sin(x^2)');
hold on
z=log(sqrt(x));
plot(x,z)
text(1,-0.1,'y=log(sqrt(x))')
xlabel('Eje x')
ylabel('Eje y')
title('Gráfico senoidal y logarítmico')

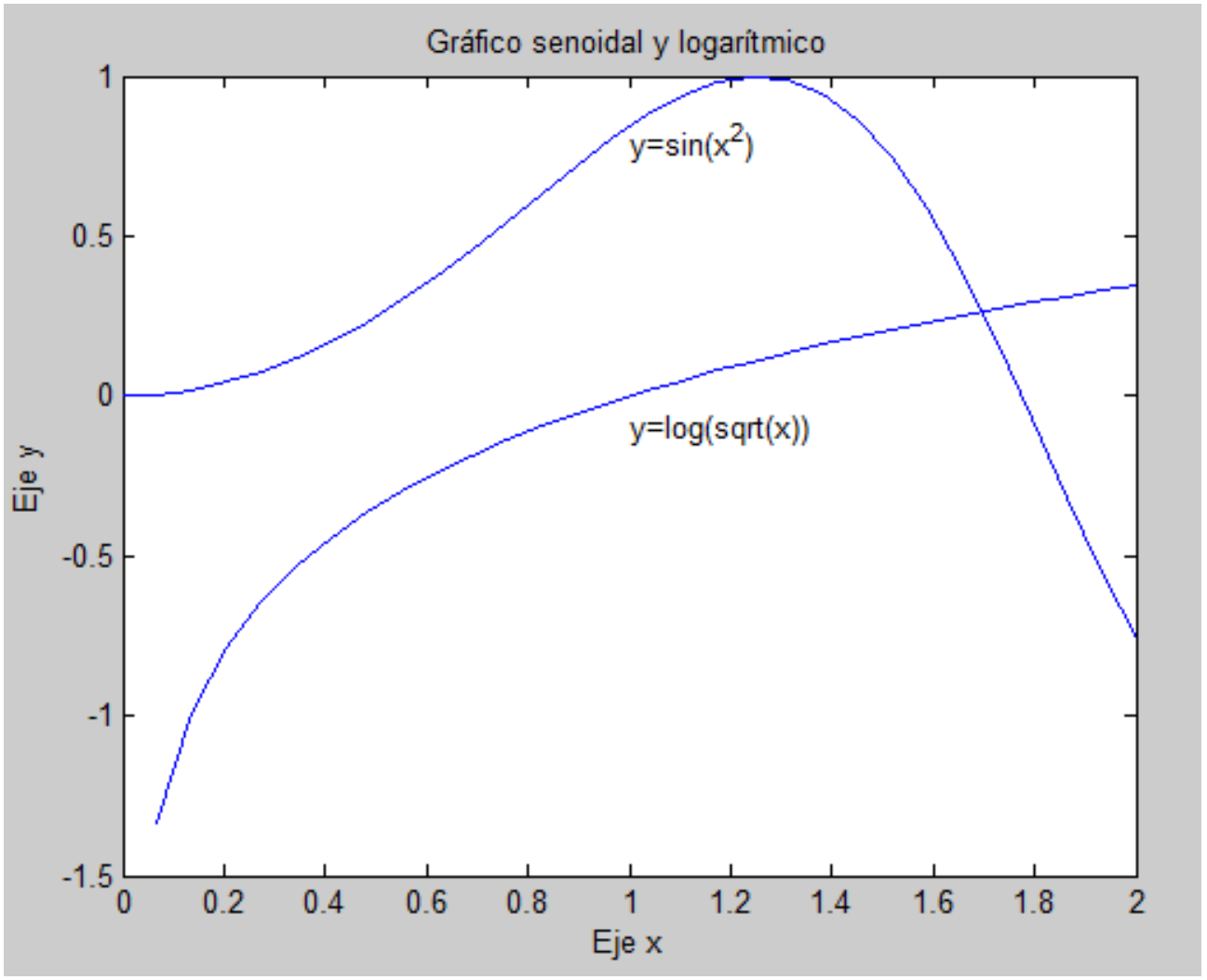
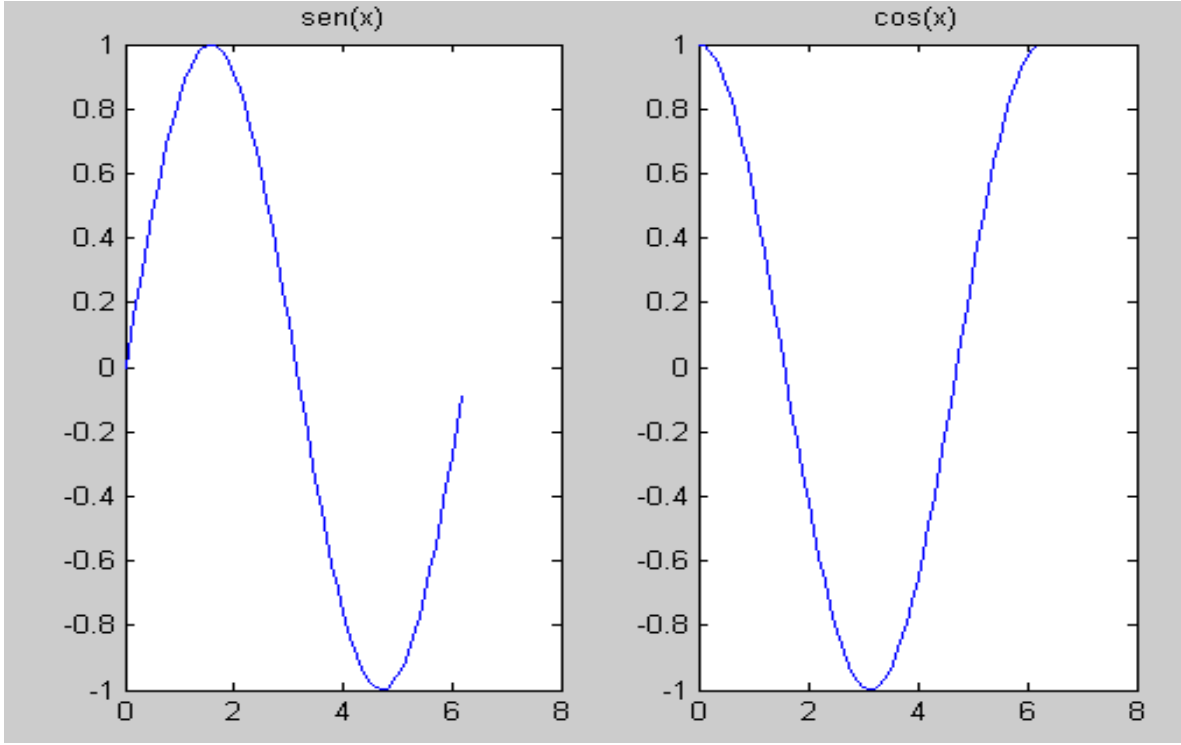


Gráfico senoidal y logarítmico
y=sin(x^2)
y=log(sqrt(x))
Eje y
Eje x
1
0.5
0
-0.5
-1
-1.5
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

Ejemplos:

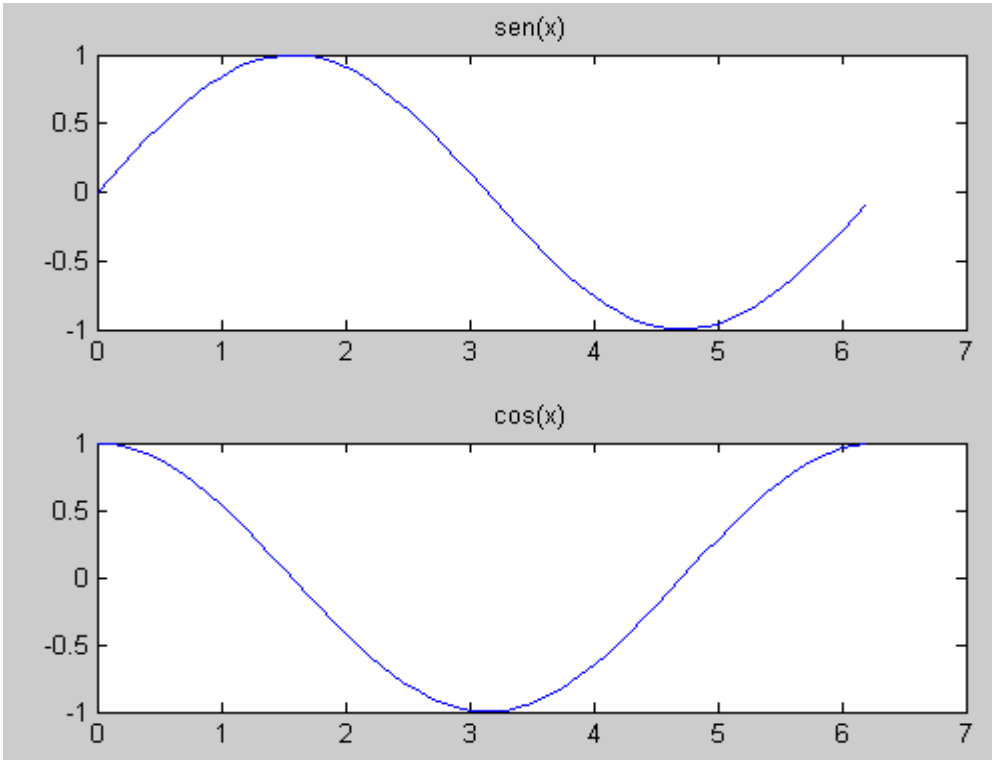
a) Graficar en dos subgráficas
una fila y dos columnas:

```
x = [0:0.1:2*pi];
y = sin(x);
z = cos(x);
subplot(121);
plot(x,y)
title('sen(x)')
subplot(122);
plot(x,z)
title('cos(x)')
```



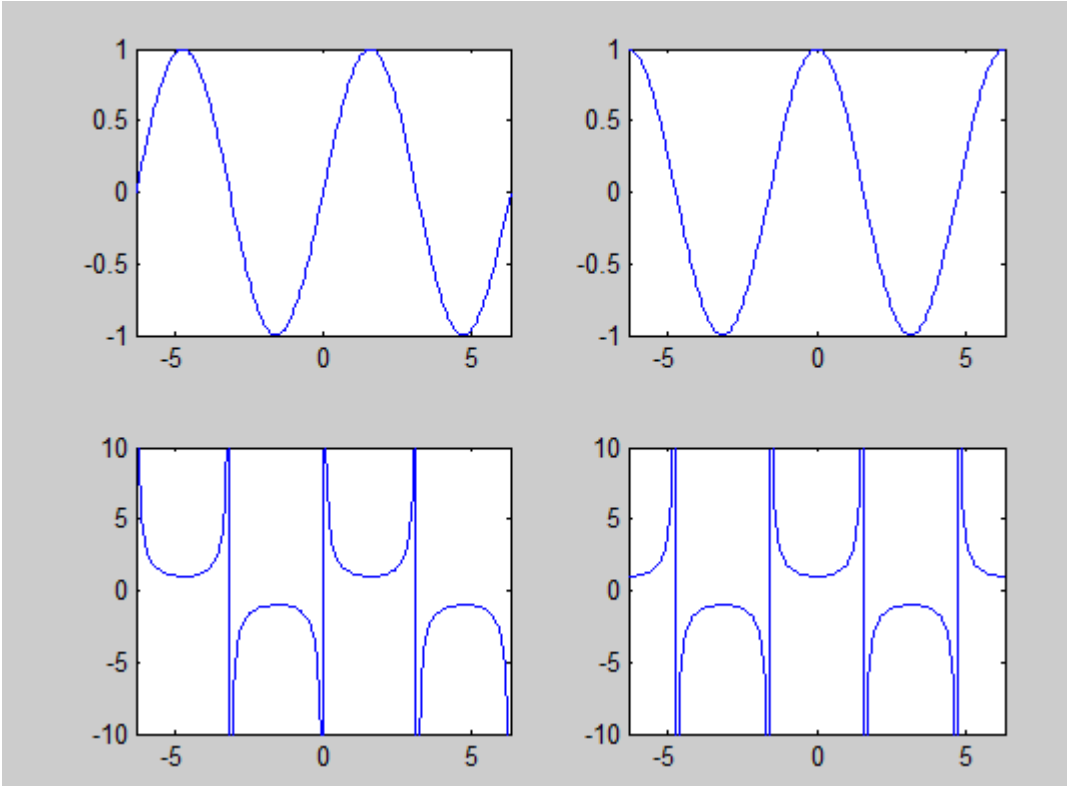
b) Graficar en dos subgráficas dos fila y una columna:

```
x = [0:0.1:2*pi];
y = sin(x);
z = cos(x);
subplot(211);
plot(x,y)
title('sen(x)')
hold on
subplot(212);
plot(x,z)
title('cos(x)')
```



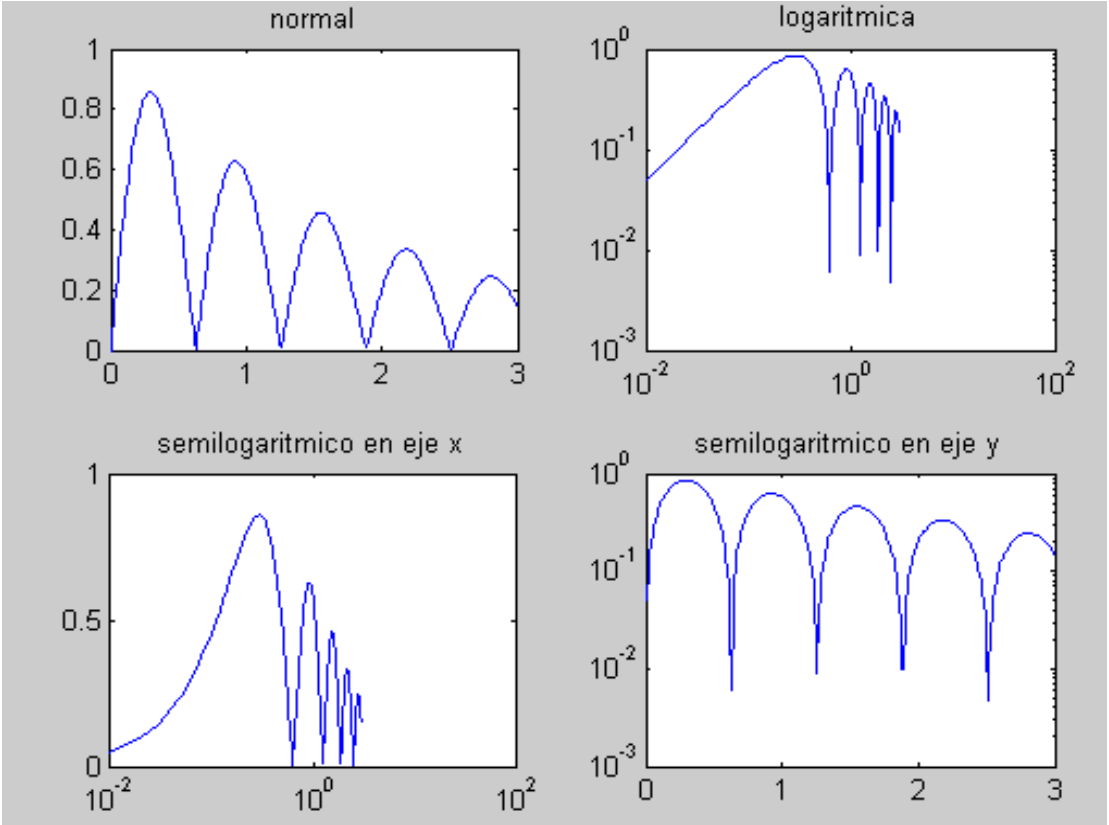
c) Graficar en cuatro subgráficas dos filas y dos columnas:

```
subplot (221);
fplot('sin(x)',[-2*pi 2*pi]);
subplot (222);
fplot('cos(x)',[-2*pi 2*pi]);
subplot (223);
fplot('csc(x)',[-2*pi 2*pi -10 10]);
subplot (224);
fplot('sec(x)',[-2*pi 2*pi -10 10]);
```



d) Graficar en diferentes escalas

```
x = 0:0.01:3;
y = abs(exp(-0.5*x).*sin(5*x));
subplot(221);
plot(x,y)
title('normal')
hold on
subplot(222)
loglog(x,y)
title('logaritmica')
subplot(223)
semilogx(x,y)
title('semilogaritmico en eje x')
subplot(224)
semilogy(x,y)
title('semilogaritmico en eje y')
```



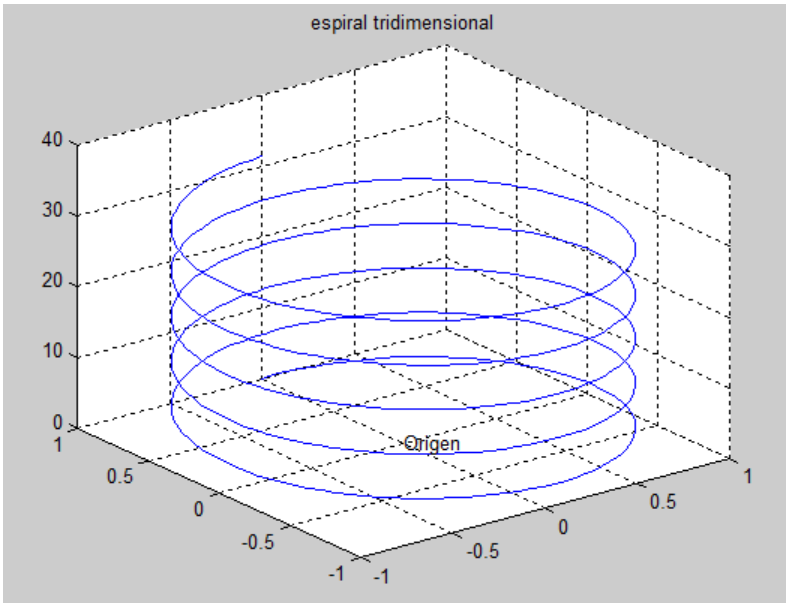
Gráficas en tres dimensiones

plot3(x,y,z)

x, y ,z son las coordenadas de la función

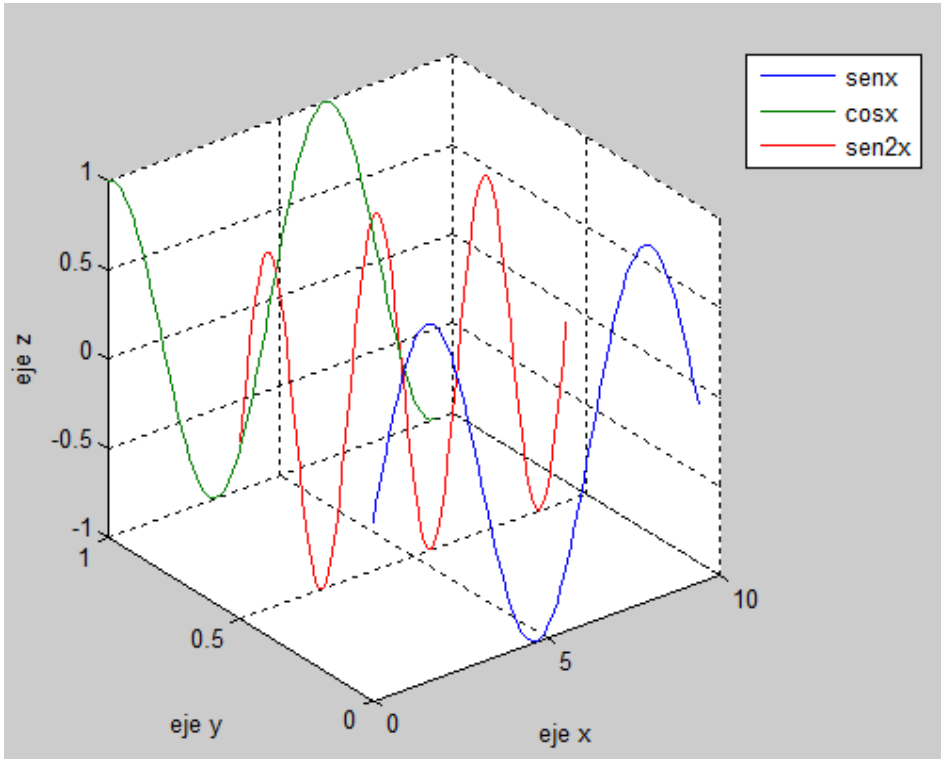
Por ejemplo: graficar x=sen(t), y=cos(t), z=t

```
t=linspace(0,10*pi,500);
plot3(sin(t),cos(t),t)
title('espiral tridimensional')
text(0,0,0,'Origen')
grid
```



Graficar: senx, cosx, cos2x en tres planos diferentes

```
x=linspace(0,3*pi,100);
z1=sin(x); z2=cos(x); z3=sin(2*x);
y1=zeros(size(x));
y2=ones(size(x));
y3=y2/2;
plot3(x,y1,z1,x,y2,z2,x,y3,z3)
xlabel('eje x'); ylabel('eje y'); zlabel('eje z');
grid
legend('senx', 'cosx', 'sen2x')
```

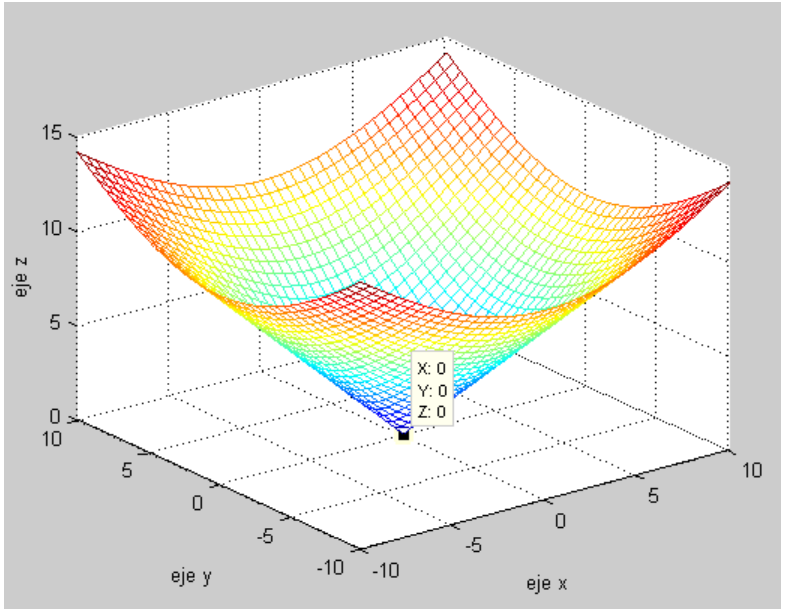


Gráficas de malla: mesh(x,y,z)

Primero hay que definir la rejilla con meshgrid que genera la matrices x,y

Ejemplo:

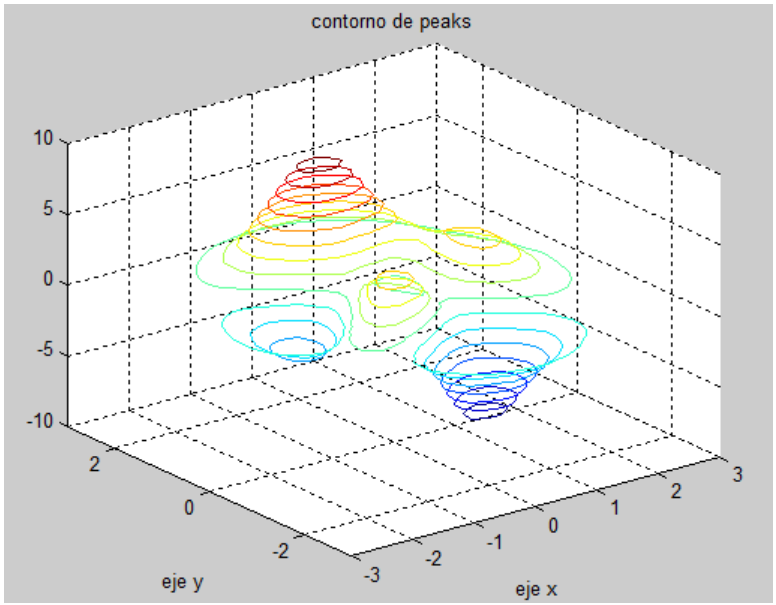
```
[x,y]=meshgrid(-10:0.5:10);
z=sqrt(x.^2+y.^2)
mesh(x,y,z)
xlabel('eje x');
ylabel('eje y');
zlabel('eje z');
```



Gráficas de contorno: contour3(x,y,z,n)

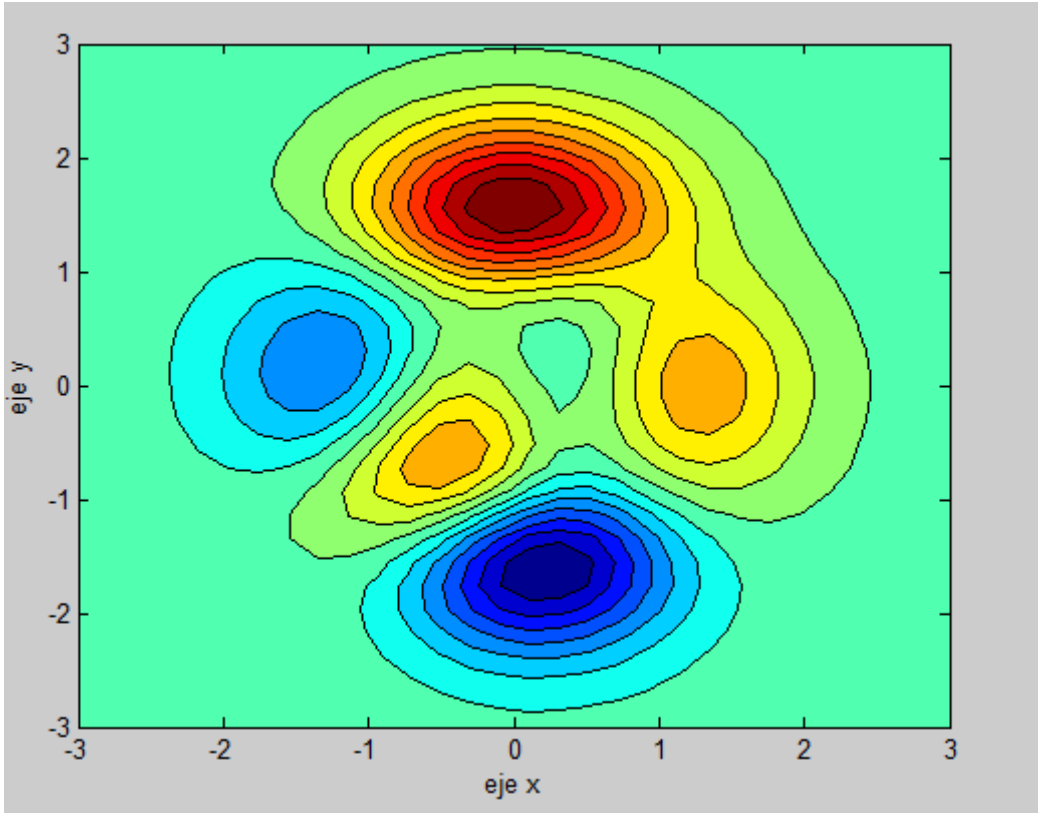
Da una gráfica de dos dimensiones de los contornos de la gráfica.

```
%peaks es una función de dos variables de Matlab
[x,y,z]=peaks(30);
%Poner 16 colores
contour3(x,y,z,16)
xlabel('eje x'); ylabel('eje y');
title('contorno de peaks')
```



Graficar n líneas de contorno en un plano

```
[x,y,z]=peaks(30);
%Poner 16 colores
contourf(x,y,z,16)
xlabel('eje x');
ylabel('eje y');
```



3. CÁLCULO NUMÉRICO

3.1 LÍMITES

OPERACIÓN MATEMÁTICA	COMANDO MATLAB
$\lim_{x \to 0} f(x)$	limit(f,x,0)
$\lim_{x \to a} f(x)$	limit (f, x, a) o limit (f, a)
$\lim_{x \to a^-} f(x)$	limit (f,x,a, 'left')
$\lim_{x \to a^+} f(x)$	limit (f,x,a, 'right')

Ejemplos:

Hallar el límite de las funciones:

a) $\lim_{x \to 2} \dfrac{x - \sqrt{2+x}}{-3 + \sqrt{1+4x}}$,

b) $\lim_{x \to 0} \dfrac{sen[(ax)]^2}{x^2}$

```
>> syms x a
>> limit((x-(2+x)^(1/2))/(-3+(1+4*x)^(1/2)),2)

ans = 9/8

>> limit(sin(a*x)^2/x^2,x,0)

ans = a^2
```

3.2 DERIVADAS

OPERACIÓN MATEMÁTICA	COMANDO MATLAB
$\frac{\partial f}{\partial x}$	diff(x) o diff(f,x)
$\frac{\partial f}{\partial t}$	diff(f,t)
$\frac{\partial^n f}{\partial b^n}$	diff(f,b,n)

Ejemplos:

a) Hallar la derivada con respecto a x de f(x) = sen(5x)

```
>> syms x
>> f = sin(5*x)
>> diff (f)

ans = 5*cos (5*x)
```

b) $g(x) = e^x \cos(x)$

```
>> g = exp(x) * cos(x)
>> diff (g)

ans = exp(x)*cos(x)-exp(x)*sin(x)
```

En estos ejemplos, Matlab simplifica. En otros casos, se debe usar el comando: *simplify*

Para una constante también se debe definir como simbólica:

Ejemplo: diff (5)

```
        ans = [ ]

c = sym('5')

diff(c)
   ans = 0
```

Ejemplos:

a) Hallar la derivada de la función f(t) = sen(st): $\frac{\partial f}{\partial t}$

```
>> syms  s   t
>> f = sin(s*t)
>> diff(f,t)
ans = cos(s*t)*s
```

b) Hallar la derivada con respecto a s: $\frac{\partial f}{\partial s}$

```
>> diff(f,s)
ans = cos(s*t)*t
```

c) Hallar la segunda derivada de f con respecto a t: $\frac{\partial^2 f}{\partial t^2}$

```
>> diff(f,t,2)
ans = -sin(s*t)*s^2
```

e) f(x) = log(sen(2x))

```
>> syms x
>> diff(log(sin(2*x)))
ans = 2*cos(2*x)/sin(2*x)
```

b) $\frac{\partial f}{\partial y}$

```
>> diff(f,y)
ans = cos(x*y)*x-2*sin(x*y^2)*x*y
```

Ejemplos:

f(x,y) = sen(xy)+cos(xy^2)
Calcular:

a) $\frac{\partial f}{\partial x}$

```
>> syms x y
>> f = sin(x*y)+cos(x*y^2)
>> diff(f,x)
ans = cos(x*y)*y-sin(x*y^2)*y^2
```

c) $\frac{\partial^2 f}{\partial x^2}$

```
>> diff(diff(f,x),x)
ans = -sin(x*y)*y^2-cos(x*y^2)*y^4
```

d) $\frac{\partial^2 f}{\partial y^2}$

```
>> diff(diff(f,y),y)
Ans = -sin(x*y)*x^2-4*cos(x*y^2)*x^2*y^2-2*sin(x*y^2)*x
```

e) $\frac{\partial^2 f}{\partial x \partial y}$

```
>> diff(diff(f,x),y)
Ans = -sin(x*y)*x*y+cos(x*y)-2*cos(x*y^2)*x*y^3-2*sin(x*y^2)*y
```

4.3 INTEGRALES

OPERACIÓN MATEMÁTICA	COMANDO MATLAB
$\int f \ dx$	int (f) integral indefinida o int (f,x)
$\int_a^b f(x)dx$	int (f,x,a,b) integral definida o int (f,a,b)
$\iint f(x)dx$	Int(int(f,x)) Integral doble
$\iint f(x,y)dxdy$	Int(int(f(x,y),x),y)
$\int_a^b \int_c^d f(x,y)dxdy$	Int(int(f(x,y),x,a,b),y,c,d))

Ejemplos:

a) Hallar la integral de $\int x^n dx$

```
>> int (x^n)
ans = x^(n+1)/(n+1)
```

$$\frac{x^{n+1}}{n+1}$$

b) $\int \frac{dy}{y}$

```
>> int (y ^(-1))
ans =  log(y)
```

c) $\int \frac{1}{a+u^2} du$

```
>> int (1/(a+u^2))
ans = 1/a^(1/2)*atan(u/a^(1/2))
```

$$\frac{\text{atan}\left(\frac{u}{a^{1/2}}\right)}{a^{1/2}}$$

d) $f = sen(a\theta + b)$

```
>> f = sin(a*teta+b)
   >> int(f)
ans = -1/a *cos(a *teta  + b)
```

e) $\int_0^{\infty} \exp(-x^2)dx \Rightarrow$

```
>> int (exp(-x^2), x, 0, inf)
ans =1/2 * pi^(1/2)
```

Ejemplos:

a) $\int a\ln(bx)dx$

```
>> syms a b x
>> int(a*log(b*x),x)

Ans =   a*x*log (b*x)-a*x
```

f) $\int_{-\infty}^{\infty} e^{-ax2}dx$

```
>> syms  a  positive
>> syms  x
>> f = exp (-a * x^2);
>> int (f,x,-inf, inf)
ans = 1/a^(1/2) *  pi ^ (1/2)
```

b) $\iint a\ln(xy)dxdy$

```
>> int(int(a*log(x*y),x),y)

ans = a*y*x*log(x*y)-2*a*x*y
```

c) $\int_0^1 a\ln(xy)dx$

```
>> int(a*log(x*y),x,0,1)
Ans = a*log(y)-a
```

d) $\int_0^1 \int_2^3 a\ln(xy)dxdy$

```
>> int(int(a*log(x*y),x,2,3),y,0,1)
Ans = -2*a*log(2)+3*a*log(3)-2*a
```

4.4 ECUACIONES DIFERENCIALES: dsolve

$$Ejemplo: \frac{d^2y(t)}{dt^2} + y(t) = 4$$

```
dsolve('D2y+y=4')

ans = C2*cos(t) + C3*sin(t) + 4
```

Si se conocen condiciones iniciales: y(0)=1, y'(0)=0

```
dsolve('D2y+y=4','y(0)=1','Dy(0)=0')

ans = 4 - 3*cos(t)
```

Ejemplo: $\frac{d^2y(x)}{dt^2} + y(x) = 4$, variable independiente es x

```
dsolve('D2y+y=4','x')
```

```
ans =C8*cos(x) + C9*sin(x) + 4
```

Para un sistema de ecuaciones:

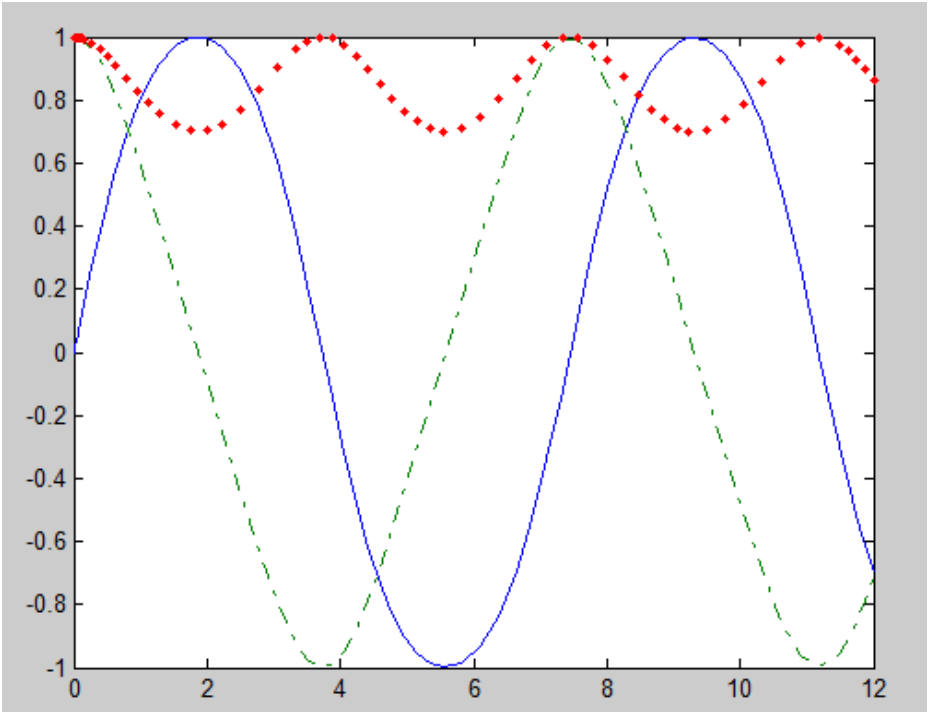
$\frac{dx}{dt} = 3x + 4y$, $\frac{dy}{dt} = -7x + 6y$

```
[x,y]=dsolve('Dx=3*x+4*y', 'Dy= - 7*x+6*y', 'x(0)=2','y(0)=1')
```

USO DE: ode45

$$\frac{dy1}{dt} = y2y3\,, \qquad y1(0) = 0,$$
$$\frac{dy2}{dt} = -y1y3, \qquad y2(0) = 1,$$
$$\frac{dy3}{dt} = -0.5y1y2, \qquad y3(0) = 1$$

```
function dy = rigid(t,y)
dy = zeros(3,1);    % a column vector
dy(1) = y(2) * y(3);
dy(2) = -y(1) * y(3);
dy(3) = -0.51 * y(1) * y(2);
```




```
[T,Y] = ode45(@rigid,[0 12],[0 1 1]);
plot(T,Y(:,1),'-',T,Y(:,2),'-.',T,Y(:,3),'.')
```

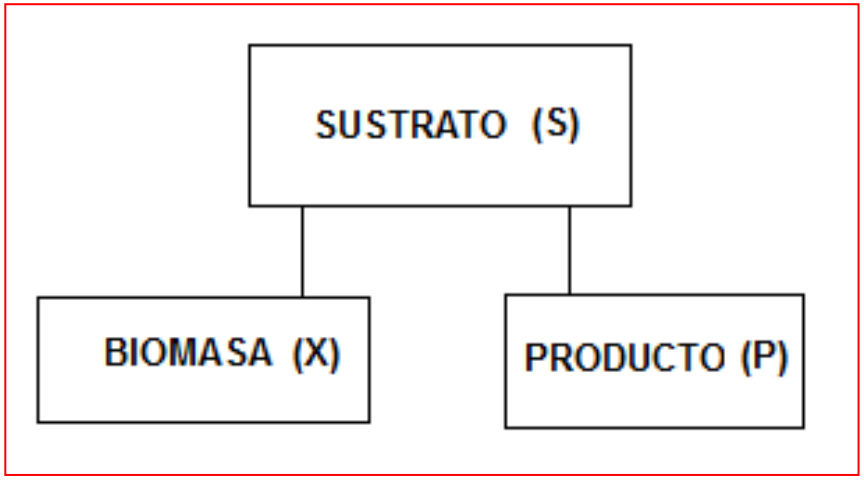
APLICACIÓN: EL BIOREACTOR

El uso de células vivas para la producción de productos químicos crece anualmente con ritmos asombrosos. Tanto microorganismos (bacterias, hongos, algas) como células humanas, vegetales o animales se utilizan para la producción varios productos químicos, como por ejemplo insulina, antibióticos, biosurfactantes. Son responsables también de la producción de alcohol vía fermentación, producción de quesos, vinos, champagne, etc. También los procesos biológicos son muy usados en el tratamiento de residuos y efluentes.

Cuando una pequeña cantidad de células vivas es adicionada en una solución líquida que contiene los nutrientes esenciales, y que se encuentra a una temperatura y un PH adecuado, las células crecerán.

El agua es el componente principal de las células, por lo tanto un suministro de agua adecuado es indispensable para lograr el mantenimiento y crecimiento microbiano y es medio de transporte de los sustratos (o contaminantes) hacia el interior de las células, y también el transporte de los compuestos que se producen dentro de la célula y que son devueltos al medio de cultivo.

La figura muestra la utilización de los sustratos para la obtención de los productos de la reacción biológica.



$$\alpha S \rightarrow \gamma X + \beta P$$

Monod en 1942 desarrolló una ecuación muy simple para representar los procesos biológicos que funciona en general muy bien.

$$\mu = = \mu_{max} \frac{S}{S + K_s}$$

Donde,
μmax=velocidad específica de crecimiento máxima, h-1
Ks=constante de saturación, g/l
S=concentración de sustrato limitante, g/l

Ecuaciones dinámicas.

$$dX/dt = \mu X$$
$$dP/dt = Y_{PX}\mu X$$
$$dS/dt = -Y_{XS}\mu X - \mu X$$

Ejemplo:

Condiciones iniciales: X(0)=0.05 g/l, P(0)=0 g/l, S(0)=14.5 g/l

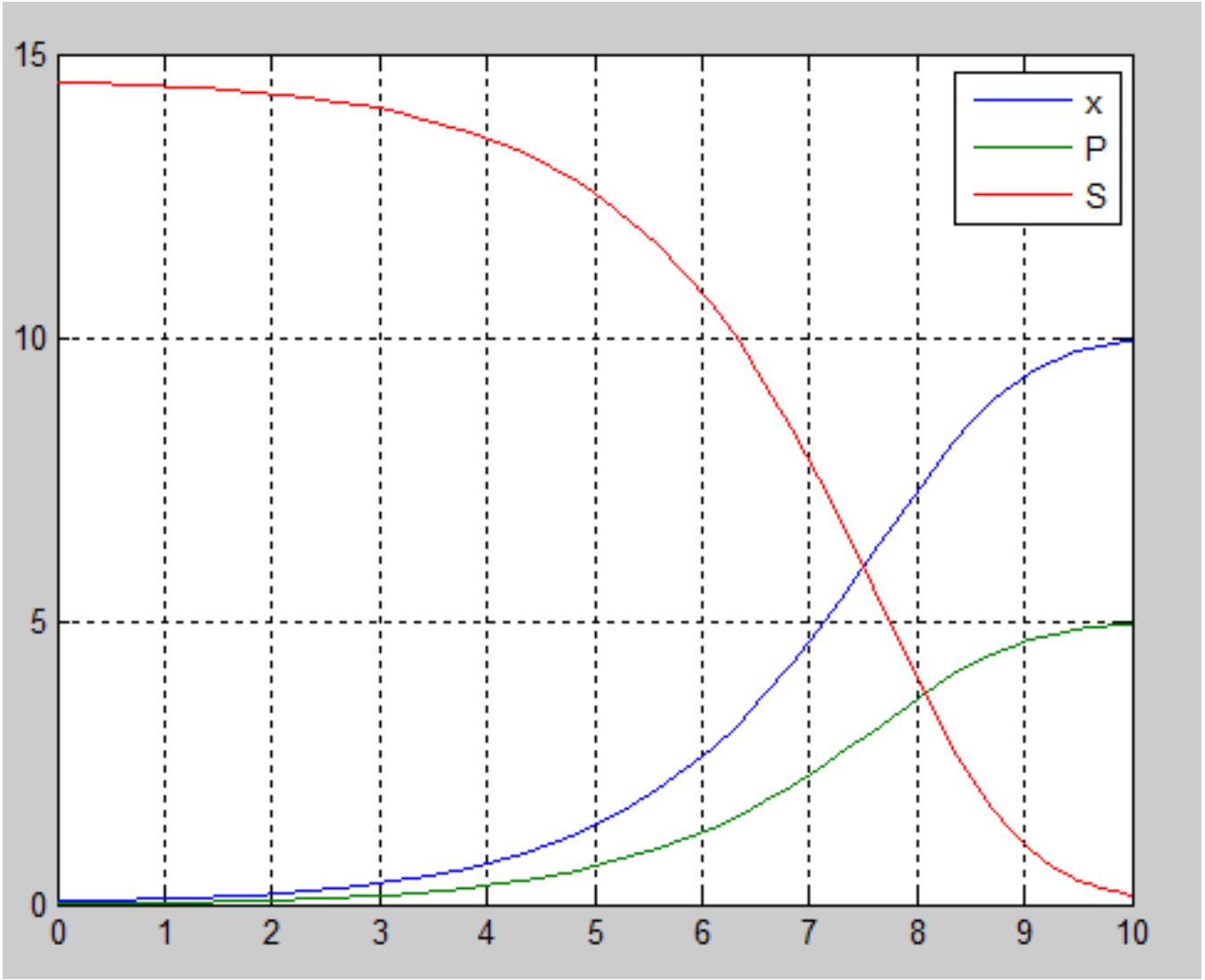
μmax=1 /h, Ks=7, Ypx=0.5, Yxs=0.45

MATLAB (biodig.m)

```
function dx = biodig(t,x)
global mumax Ks Ypx Yxs
dx=zeros(3,1);
mu=mumax*x(3)/(Ks+x(3));
%mu=mumax*S/(Ks+S);
dx(1)=mu*x(1);
%dx=mu*x;
dx(2)=Ypx*mu*x(1);
%dP=Ypx*mu*x;
%dx(3)=(1/Yxs)*mu*x(1)-mu*x(1);
dx(3)=-Yxs*mu*x(1)-mu*x(1);
%dS=(1/Yxs)*mu*x-mu*x;
end
```

MATLAB (Reactor2.m)

```
%Resolución de un bioreactor
%X: Biomasa
%S: sustrato
%P: producto
clear; clc; clf;
global mumax Ks Ypx Yxs
mumax=1; %velocidad máxima de crecimiento celular
Ks=7;   %velocidad específica de consumo de sustrato
%coeficiente de rendimiento de producto basado en el
%crecimiento celular
Ypx=0.5;
%coeficiente de rendimiento de biomasa basado en el
%consumo total de sustrato
Yxs=0.45;
yo=[0.05 0 14.5];   %condiciones iniciales xo,Po,So
intervalo=[0 10];   %intervalo de tiempo
[T,X]=ode45(@biodig,intervalo,yo);
plot(T,X)
legend('x','P','S')
```


15
10
5
0
0
1
2
3
4
5
6
7
8
9
10
x
P
S

4. DINÁMICA DE SISTEMAS

4.1. SISTEMAS

Un sistema es una combinación de componentes que actúan conjuntamente para alcanzar un objetivo específico. Un sistema es dinámico cuando la salida presente depende de las entradas pasadas y es estático cuando la salida presente depende solamente de las entradas presentes.

Los componentes básicos de un sistema son:

a) Elementos que son las partes del sistema
b) Estructura. Se refiere a las interrelaciones entre las partes del sistema.
c) Ambiente. Relaciona el sistema con el todo. Es su entorno
d) Entradas. Son las fuentes de energía, recursos e información
e) Salidas. Son los productos o resultados.

Los sistemas pueden clasificarse de las siguientes maneras:

a) Sistemas de lazo abierto y
b) Sistemas en lazo cerrado, que son los que realimentan parte de su salida a la entrada.

4.2 MODELO MATEMÁTICO

Es la descripción matemática que predice el funcionamiento del sistema. En los sistemas físicos el modelo matemático se describe por ecuaciones diferenciales. Los sistemas lineales se modelan con ecuaciones diferenciales lineales y son aquellos que se les aplica el principio de superposición, esto es, la respuesta de un sistema a varias entradas simultáneas es la suma de las respuestas individuales.

Para elaborar un modelo:

a) Se debe dibujar un diagrama esquemático del sistema y definir las variables.
b) Escribir las ecuaciones utilizando las leyes físicas de cada componente, combinándolos de acuerdo al diagrama y obtener el modelo matemático
c) Verificar la validez del modelo comparando la predicción de las ecuaciones del modelo con los resultados experimentales. El modelo se debe ajustar hasta que haya una buena concordancia entre lo teórico y lo práctico.

En general, la construcción de modelos se basa en la observación del sistema. Existen algunos caminos básicos para obtener un modelo:

Modelamiento de Sistemas: Esta estrategia consiste en descomponer (abstractamente) el sistema en subsistemas más simples, cuyos modelos sean factibles de obtener gracias a la experiencia previa. Una vez obtenidos estos submodelos, se buscan las relaciones que existen entre ellos, para interconectarlos y obtener el modelo del sistema original.

Esta estrategia busca una descripción desde adentro del sistema, generalmente basada en el conocimiento de las leyes que rigen los sistemas simples.

Identificación de Sistemas: Esta estrategia consiste en acumular un número suficiente de observaciones sobre las señales de entrada y salida del sistema, con el propósito de emplearlas para construir un modelo del mismo. No se centra en lo que existe al interior del sistema, sino en su comportamiento respecto al entorno.

Estrategia híbrida: Existe una tercera estrategia, que realmente es una combinación de las anteriores: Al igual que en la estrategia de modelamiento, se emplea el conocimiento que esté a la mano acerca de la estructura interna del sistema y las leyes que rigen su comportamiento, y se emplean observaciones para determinar la información que haga falta.

Para un sistema continuo de una única entrada y una única salida, el modelo empleado corresponde a una ecuación diferencial ordinaria de coeficientes constantes:

$$a_n \frac{d^n x}{dt^n} + \ldots + a_1 \frac{dx}{dt} + a_0 x(t) = b_m \frac{d^m u}{dt^m} + \ldots + b_1 \frac{du}{dt} + b_0 u(t)$$

Otro tipo de ecuaciones diferenciales que se emplearán relacionan vectores de variables mediante matrices.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}$$

4.3 LAPLACE CON MATLAB

LAPLACE DIRECTA

Para calcular la Transformada de Laplace de una función se usa el comando

>> laplace(f(t))

Ejemplos:

Calcular la transformada de Laplace de:

a) f(t) = cos(wt)+sen(wt)

```
>> syms w t
>> laplace(cos(w*t)+sin(w*t))

ans =

s/(s^2+w^2)+w/(s^2+w^2)
```

b) $f(t) = 3t + 2t^2$

```
>> laplace((3*t)+2*t^2)

ans =

3/s^2+4/s^3
```

c) $f(t) = 3e^{-2t} - 2e^{5t}$

```
>> laplace(3*exp(-2*t)- 2*exp(5*t))

ans =

3/(s+2)-2/(s-5)
```

LAPLACE I NVERSA

Dada la Transformada de Laplace F(s), la Transformada Inversa de Laplace es f(t). El comando o función matlab para obtener la función en el tiempo es:

>> ilaplace(fs)

Ejemplo:

$$F(s) = \frac{s+2}{s^2+2s+2}$$

```
>> syms s
>> fs=(s+2)/(s^2+2*s+2);
>> ilaplace(fs)

ans =
exp(-t)*(cos(t)+sin(t))
```

$$f(t) = e^{-t}[\cos(t) + \sin(t)]$$

4.4 ECUACIONES DIFERENCIALES CON MATLAB

Para resolver o solucionar una ecuación diferencial:

a) Se aplica transformada de Laplace a la ecuación
b) Se despeja F(s)
c) Se halla la transformada inversa

Ejemplo:

Resolver la ecuación diferencial:

$$\frac{d^2y}{dt^2} + 3\frac{dy}{dt} + 2y(t) = 5, \quad \text{condiciones iniciales } y(0) = -1, y'(o) = 2$$

a) Aplicamos transformada de Laplace:

$$s^2Y(s) - sy(0) - y'(0) + 3sY(s) - 3y(o) + 2Y(s) = 5/s,$$

Reemplazando los valores de y(0) y $y'(0)$, se tiene:

b) Despejando Y(s),

$$Y(s) = \frac{-s^2 - s + 5}{s(s^2 + 3s + 2)} = \frac{-s^2 - s + 5}{s(s+1)(s+2)}$$

c) Aplicando Transformada inversa de Laplace:,

$$Y(s) = \frac{-s^2 - s + 5}{s(s^2 + 3s + 2)}$$

```
>> syms s
>> fs=(-s^2-s+5)/(s*(s^2+3*s+2));
>> ilaplace(fs)

ans =

5/2+3/2*exp(-2*t)-5*exp(-t)
```

La transformada inversa es:

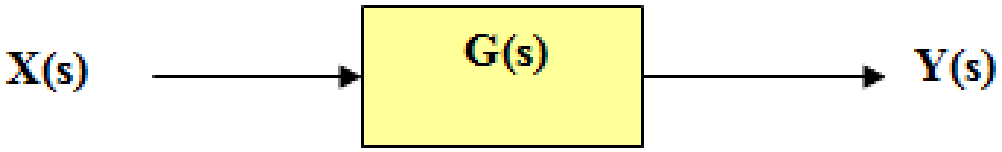
$$y(t) = 1.5e^{-2t} - 5e^{-t} + 2.5$$

4.5 FUNCIÓN DE TRANSFERENCIA

La función de transferencia de un sistema descrito mediante una ecuación diferencial, se define como el cociente de la Transformada de Laplace de la salida y la Transformada de Laplace de la entrada.

El método para encontrar la función de transferencia de un sistema es el siguiente:

1. Escribir la ecuación diferencial del sistema
2. Aplicar la Transformada de Laplace de la ecuación diferencial con condiciones iniciales cero
3. Sacar el cociente entre la variable de salida y la entrada



sistema

$$G(s) = \frac{Y(s)}{X(s)} = \frac{b_0 + b_1 s + \ldots + b_m s^m}{a_0 + a_1 s + \ldots a_n s^n}$$

En Matlab:

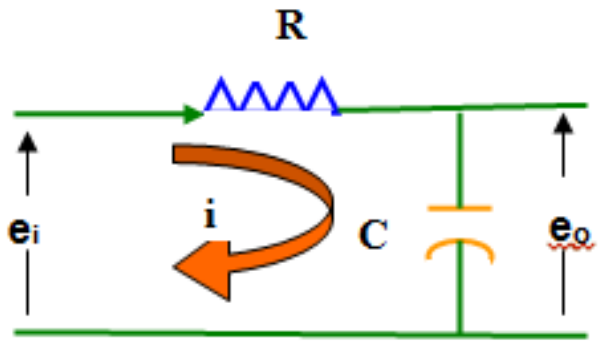
```
>> num = [bo  b1  b2 ....  bm];
>>  den = [ao  a1  a2 ....  an];
>> Gs = tf (num, den)
```

4.6 DIAGRAMA EN BLOQUES

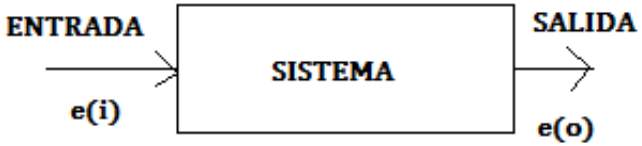
La estructura de un sistema se representa por un diagrama en bloques según el siguiente procedimiento:

a) Definir la entrada y la salida
b) Escribir las ecuaciones que describa el comportamiento de cada elemento del sistema
c) Aplicar transformada de Laplace a cada elemento
d) Integrar los elementos en un diagrama completo (estructura)

Ejemplo: Circuito serie RC



a) Entrada: e_i, salida: e_o



b) Ecuaciones de cada elemento

Para la resistencia

$$e_R = Ri, \quad i = \frac{e_i - e_o}{R}$$

para el condensador

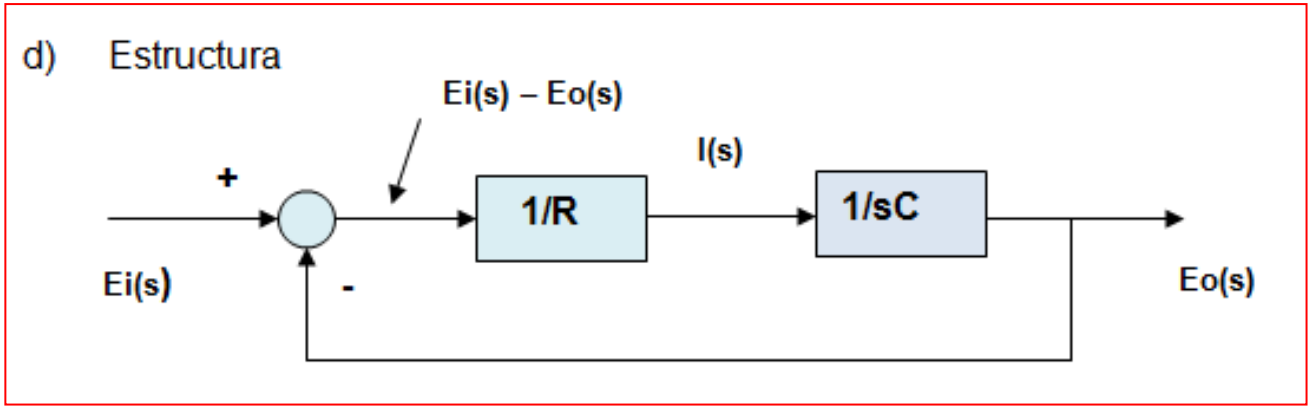
$$e_o = \frac{1}{C}\int idt$$

c)Transformada de Laplace

$$I(s) = \frac{E_i(s) - E_o(s)}{R},$$

$$E_o(s) = \frac{1}{sC} I(s)$$

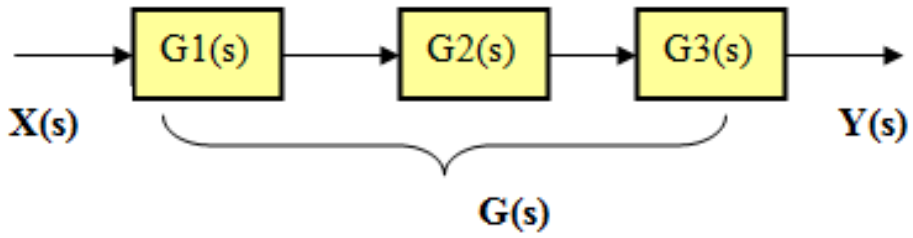
d) Estructura



TIPOS DE ESTRUCTURAS

Los diagramas de bloques mediante los cuales se estructura un sistema son complejos y son generalmente combinaciones de los siguientes tipos de estructuras:

ESTRUCTURA SERIE



$$G(s) = G1(s)*G2(s)*G3(s)$$

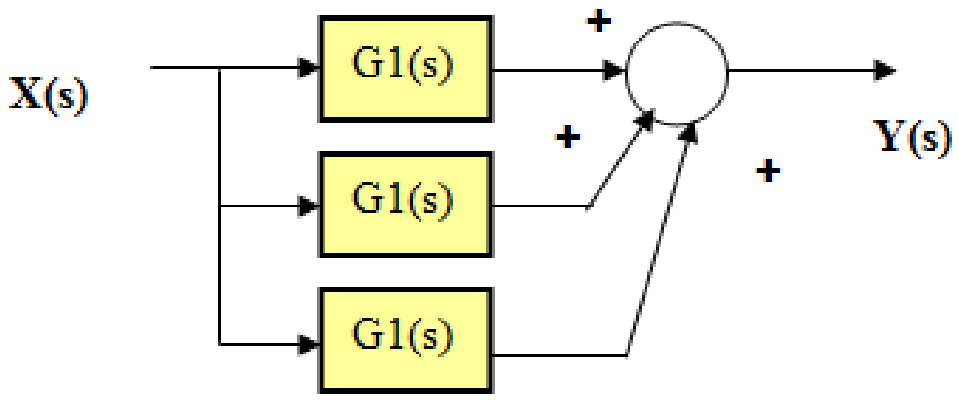
Matlab

```
G1 = tf (num1,den1);
G2 = tf (num2,den2);
G3 = tf (num3,den3);
Gs = G1*G2*G3
```

también puede ser:

```
Gs = series(G1,G2,G3)
```

ESTRUCTURA PARALELA



$$G(s) = G1(s)+G2(s)+G3(s)$$

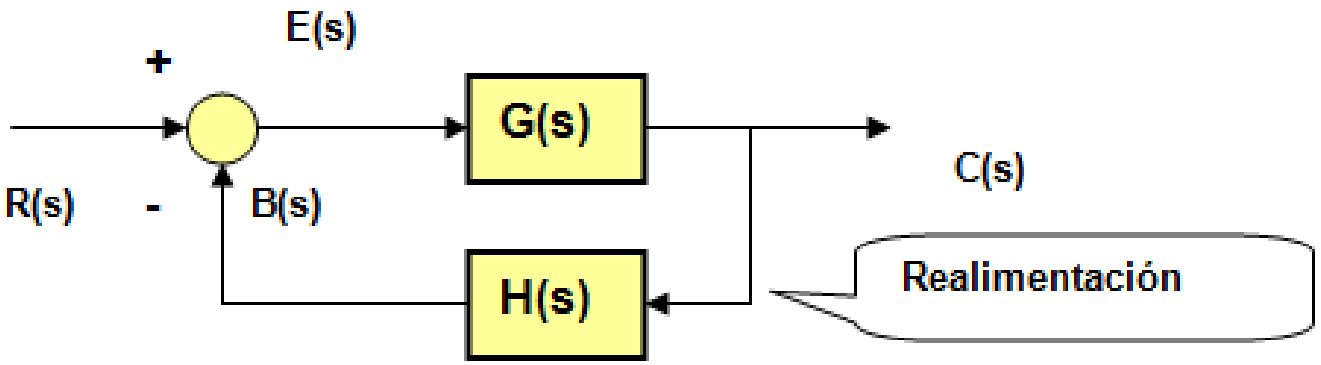
Matlab

```
G1 = tf (num1, den1);
G2 = tf (num2, den2);
G3 = tf (num3, den3);
Gs = G1+G2+G3
```

también puede ser :

```
Gs = parallel(G1, G2, G3)
```

ESTRUCTURA REALIMENTADA



a) Función de transferencia en lazo abierto: Gla

$$Gla = \frac{B(s)}{E(s)} = G(s) * H(s)$$

b) Función de transferencia en lazo cerrado: Glc

$$Glc = \frac{C(s)}{R(s)} = \frac{G(s)}{1 + G(s)H(s)},$$

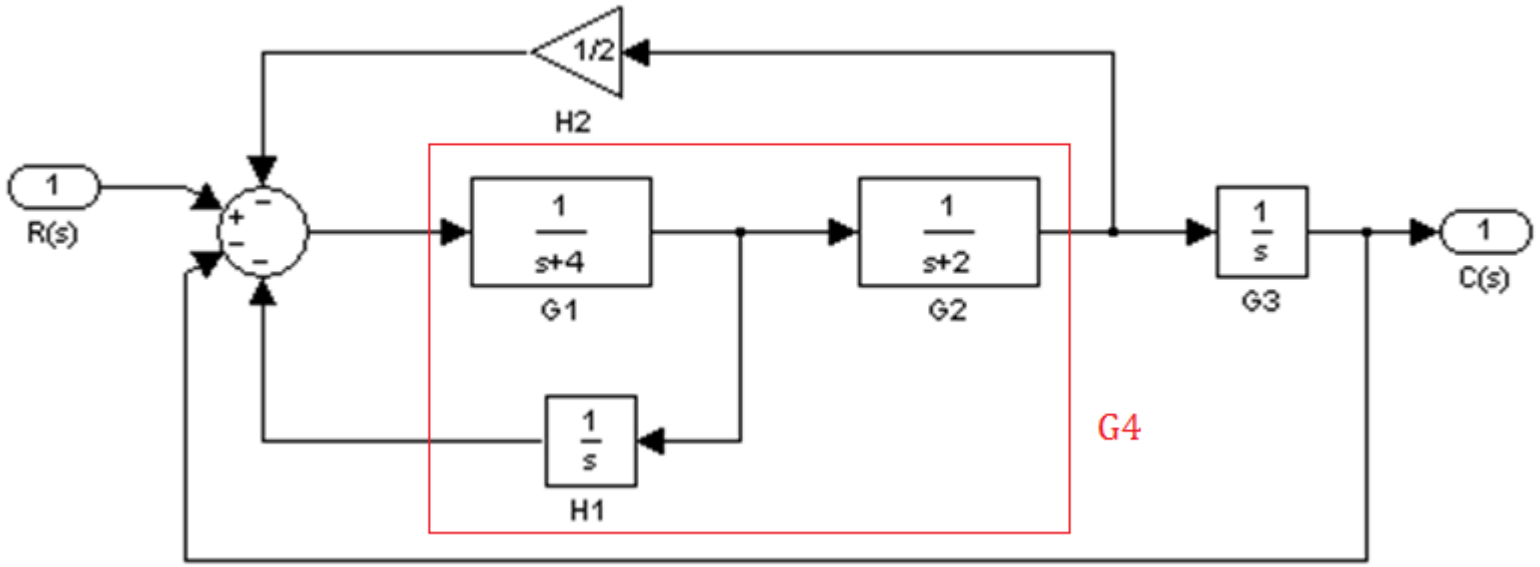
Matlab:

*>> Gla = Gs*Hs*

>> Glc = feedback(Gs,Hs)

Ejemplo:

Obtener la función de transferencia del sistema por Matlab: $G(s) = \frac{C(s)}{R(s)}$



```
% (1) Primera realimentación

G1=tf(1,[1 4])
H1=tf(1,[1 0])

% Realimentación de G1 y H1
Glc1=feedback (G1,H1)

% (2) Estructura serie
G2=tf(1,[1 2])
G4=Glc1*G2

% (3) Segunda realimentación
H2=1/2
Glc2=feedback(G4,H2)

% (4) Segunda estructura serie
G3=tf(1,[1 0])
G5=Glc2*G3

% (5) Tercera realimentación
Glc=feedback(G5,1)
```

Respuesta: $$Glc = \frac{s}{s^4 + 6s^3 + 9.5s^2 + 3s}$$

RESPUESTA DE UN SISTEMA

Generalmente se conoce como respuesta de un sistema la salida en el dominio del tiempo que tiene el sistema cuando a su entrada se le aplica una función escalón unitaria. También se conoce como respuesta al paso unitario.

Para el ejemplo anterior la respuesta al paso unitario se obtiene adicionando la instrucción:

```
>> step(Glc)
```

Esta respuesta tiene como característica importante la amplitud de pico, el sobreimpulso (overshoot) y el tiempo de establecimiento (setting time).

Ejemplo:

La función de transferencia en lazo cerrado de un sistema es:

$$Glc(s) = \frac{s^2 + 3s + 2}{s^3 + 6s^2 + 5s + 6}$$

Matlab:

```
>> num=[1 3 2];
>> den=[1 6 5 6];
>> Glc=tf(num,den)
>>step(Glc)
```