Práctica 9. Ecuaciones Diferenciales Ordinarias.

Métodos Numéricos para la Computación

Grado en Ingeniería Informática. Mención Computación
Escuela de Ingeniería Informática
Universidad de Las Palmas de Gran Canaria

Contenidos

- Resolución de ecuaciones diferenciales ordinarias de primer orden mediante las funciones ode() de MATLAB
- Resolución de ecuaciones de orden superior. Ejemplo del resorte.
- Resolución de sistemas de ecuaciones. Atractor de Lorenz.

Tarea 1.Modelo de un amortiguador

Un sistema físico como un amortiguador de un automóvil está basado en un resorte. Puede representarse mediante un modelo matemático basado en una ecuación de segundo orden, lineal y de coeficientes constantes.

Donde *m* es la masa, *b* un factor de fricción o amortiguación, *k* la constante del resorte y *f(t)* una fuerza externa aplicada al amortiguador. Si suponemos que inicialmente está en reposo y en la situación inicial, se tiene:

$$m\frac{d^2x}{dt^2}+b\frac{dx}{dt}+kx=f(t)$$

$$x(0)=0;$$

$$x'(0)=0$$

El modelo de segundo orden se transforma en un sistema de primer orden:

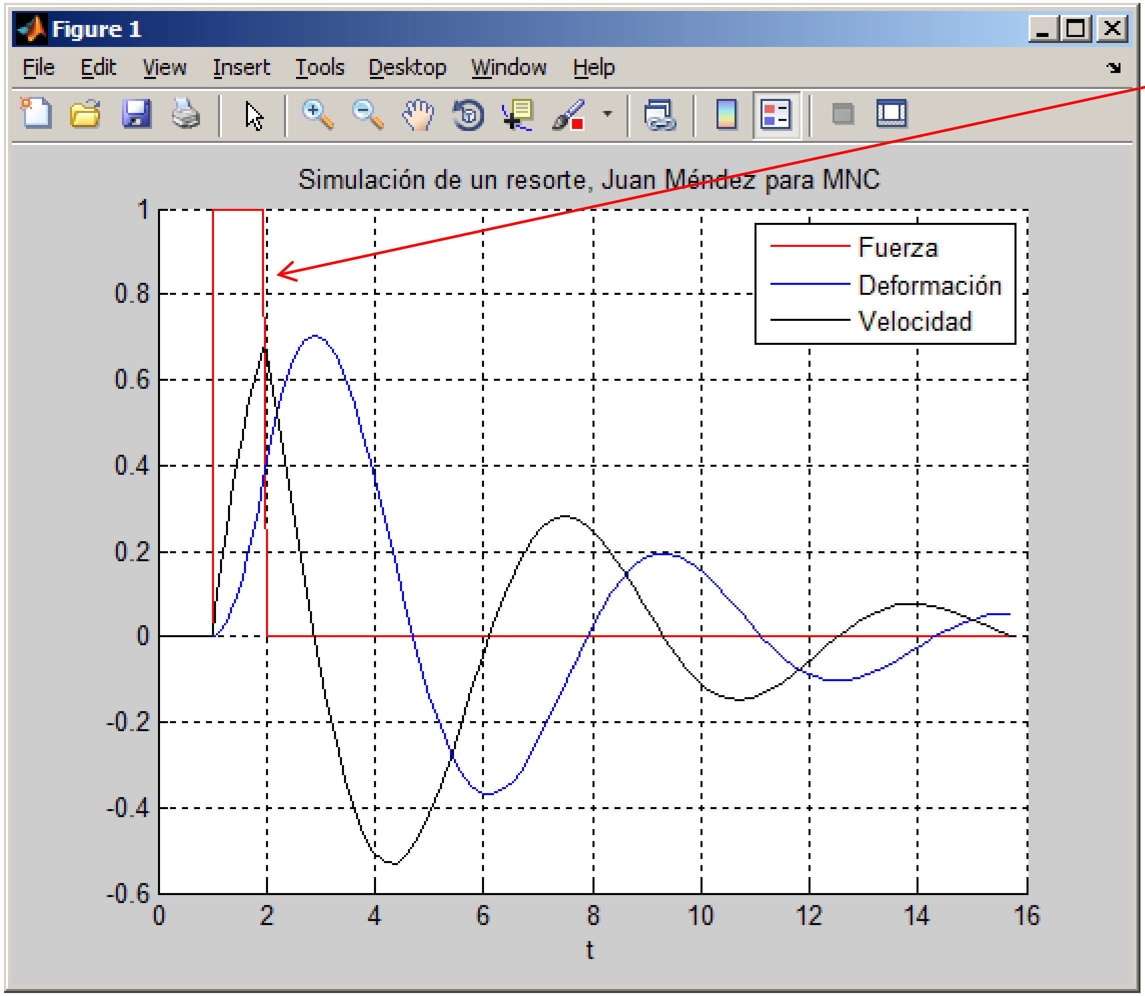
$$\frac{dx_1}{dt}=x_2$$

$$\frac{dx_2}{dt}=f(t)-(b/m)x_2-(k/m)x_1$$

$$x_1(0)=0;$$

$$x_2(0)=0$$

Resultado de la Simulación

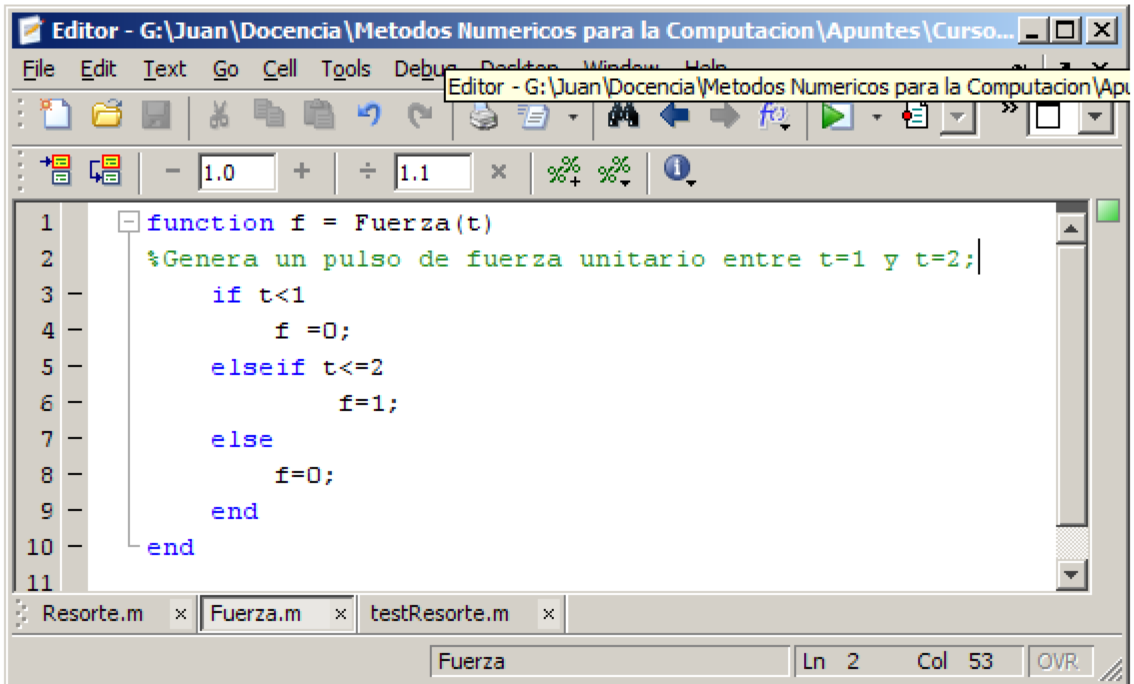


La fuerza aplicada es un pulso de valor 1 entre t=1 y t=2.

Si no fuese por ese pulso de fuerza, el resorte seguiría en la posición x=0 y en reposo.

El factor de amortiguamiento produce que el sistema regrese progresivamente a la situación de reposo.

El modelo de la Fuerza



El código completo de resolución:

```
function [t,x,v] = Resorte(m,r,k,x0,dx0,rango,f)
% simula el modelo de un resorte. la fuerz exterior e una función

[t,s]= ode45(@(u,v)sfunc(u,v,m,r,k,f),[rango(1),rango(2)],[x0;dx0]);
% la primera columna es x1 la variable, la segunda la x2 s(:,2)
x=s(:,1);
v=s(:,2);
end

% sfunc es la función del sistema
function dx = sfunc(t,x,m,r,k,f)

    % debe ser una columna de valores
    dx(1,1) = x(2);
    dx(2,1) = f(t)-(r/m)*x(2)-k*x(1);
end
```

El programa que realiza la preparación de datos, invocación de la resolución y presentación d e los resultado

```matlab
clear all;
clc;

m = 1;
b = 0.4;
k = 1;
tmax = 5*pi;

[t,x,v]= Resorte(m,b,k,0,0,[0,tmax],@Fuerza);

f = zeros(size(t));
for a=1:length(t);
    f(a) = Fuerza(t(a));
end
figure;
hold on;
plot(t,f,'r-');
plot(t,x,'b-');
plot(t,v,'k-');
xlabel('t');
grid on;
legend('Fuerza','Deformación','Velocidad');
title('Simulación de un resorte, Juan Méndez para MNC');
```

Tarea 2: Atractor de Lorenz

Es un ejemplo clásico de sistema de EDO que presenta un interés conceptual importante en la ciencia actual, pues siendo un sistema completamente determinista, presenta un comportamiento caótico. Su estudio fue fundamental para asentar la Teoría del Caos.

$$\frac{dx}{dy} = a(y - x)$$

$$\frac{dy}{dy} = x(b - z) - y$$

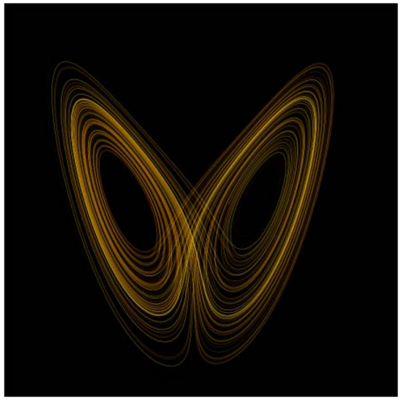
$$\frac{dz}{dy} = xy - cz$$

Para los valores a=10, c=8/3 y b= 28, y valores iniciales [1 1 1] el sistema tienen un comportamiento caótico, que resolveremos en la Práctica correspondiente.

https://en.wikipedia.org/wiki/Lorenz_system

Un poco de cultura científica

https://es.wikipedia.org/wiki/Teor%C3%ADa_del_caos



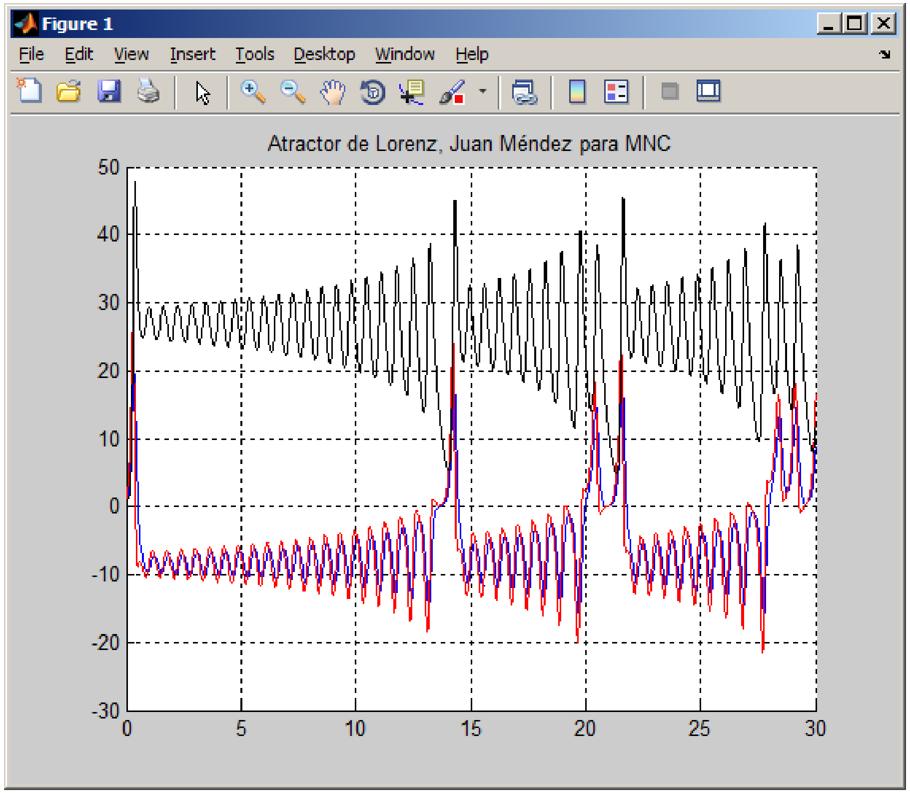
La **teoría del caos** es la denominación popular de la rama de las matemáticas, física, biología, meteorología, economía, que trata ciertos tipos de sistemas complejos y sistemas dinámicos muy sensibles a las variaciones en las condiciones iniciales.

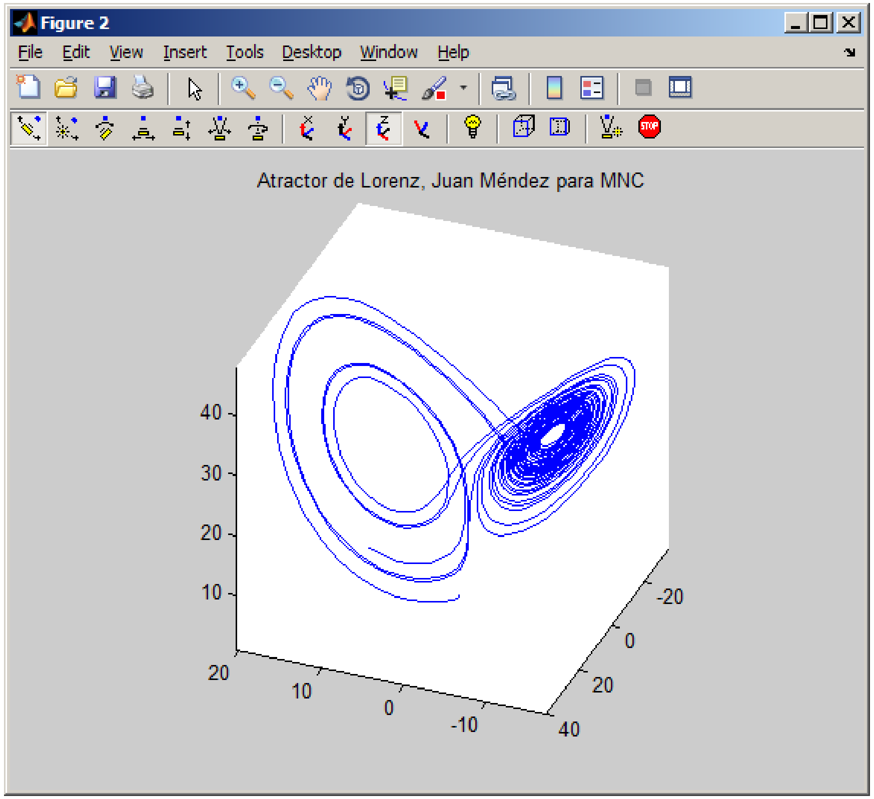
Pequeñas variaciones en dichas condiciones iniciales pueden implicar grandes diferencias en el comportamiento futuro, imposibilitando la predicción a largo plazo, a pesar que son sistemas deterministas.

Este obstáculo de la predicción se conoce con el nombre *efecto mariposa* por una charla de Lorenz con el título "*¿Puede el batir de las alas de una mariposa en Brasil dar lugar a un tornado en Texas?*".

Si los diagramas de fases no muestran una trayectoria bien definida, sino que ésta es errabunda alrededor de algún movimiento bien definido. Cuando esto sucede se dice que el sistema es atraído hacia un tipo de movimiento, es decir, que hay un **Atractor**

Gráficos del Atractor de Lorenz





```matlab
function  AtractorLorenz
% Lanza una simulación del Atractor de Lorenz

    % entre t=0 y t=30
    [t,X]= ode45(@LorenzFunction,[0,30],[1 1 1]);

    figure;
    hold on;
    plot(t,X(:,1),'b-');
    plot(t,X(:,2),'r-');
    plot(t,X(:,3),'k-');
    title('Atractor de Lorenz, Juan Méndez para MNC');
    grid on;

    figure;
    plot3(X(:,1),X(:,2),X(:,3),'b-');
    title('Atractor de Lorenz, Juan Méndez para MNC');
    cameratoolbar;

end

function dX = LorenzFunction(t,X)

    a = 10;
    b = 28;
    c = 8/3;
    %vector columna para los resultados.
    dX(1,1)  = a*(X(2)-X(1));
    dX(2,1)  = X(1).*(b-X(3))-X(2);
    dX(3,1)  = X(1).*X(2) -c*X(3);
end
```

Programa que computa y dibuja el Atractor de Lorenz usando ode45.

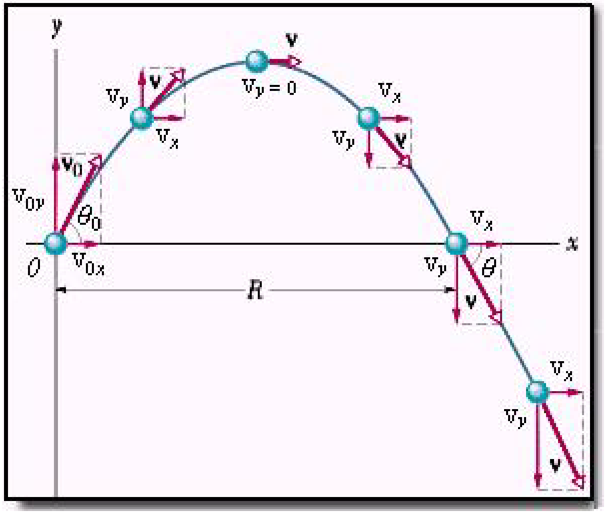
Ejecutar AtractorLorenz en la línea de comandos.

Tarea 3. Tiro parabólico

El tiro parabólico es un problema bien definido y estudiado. Se conoce perfectamente su solución, pero en esta tarea el alumno aprenderá a resolverlo sin utilizar la solución analítica, sino resolviendo numéricamente el problema de un sistema de EDO de segundo orden.

$$\frac{d^2y}{dt^2} = -g \quad y(0) = 0 \quad y'(0) = v\sin\theta$$

$$\frac{d^2x}{dt^2} = 0 \quad x(0) = 0 \quad x'(0) = v\cos\theta$$



Transformación a sistema de EDO

$$x_1 = x \quad x_1(0) = 0$$

$$x_2 = x' \quad x_2(0) = v\cos\theta$$

$$x_3 = y \quad x_3(0) = 0$$

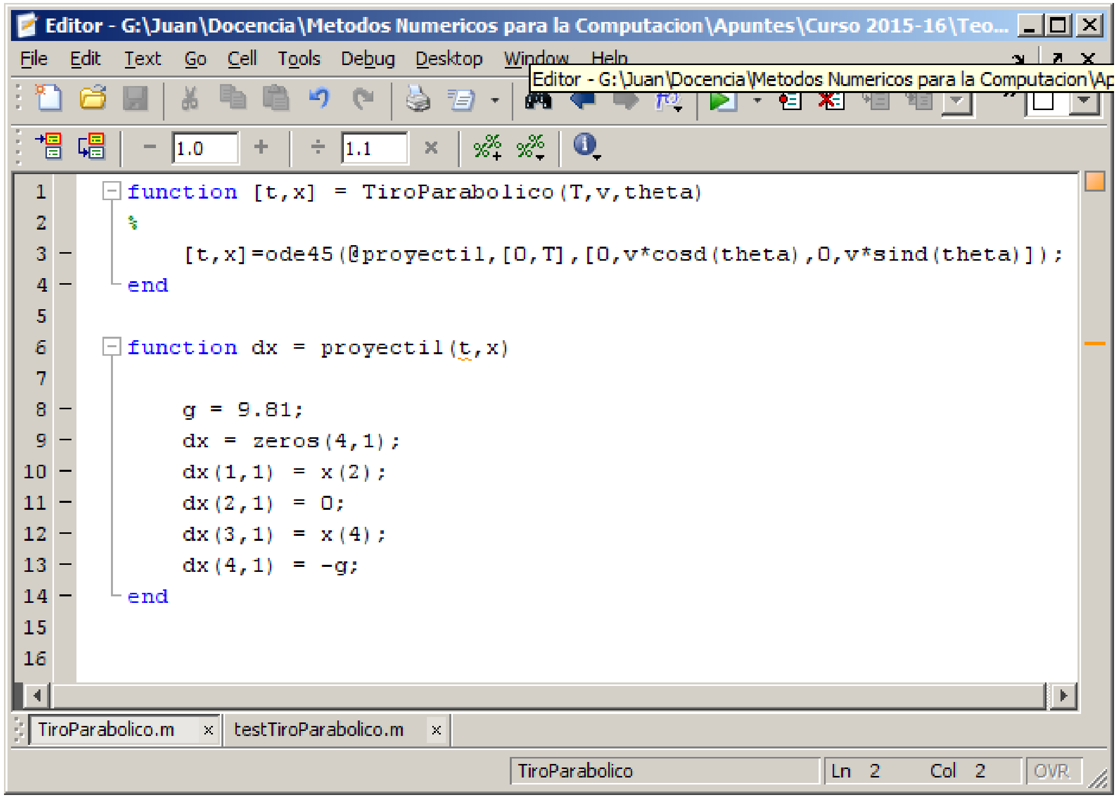
$$x_4 = y' \quad x_4(0) = v\sin\theta$$

$$\frac{dx_1}{dt} = x_2$$

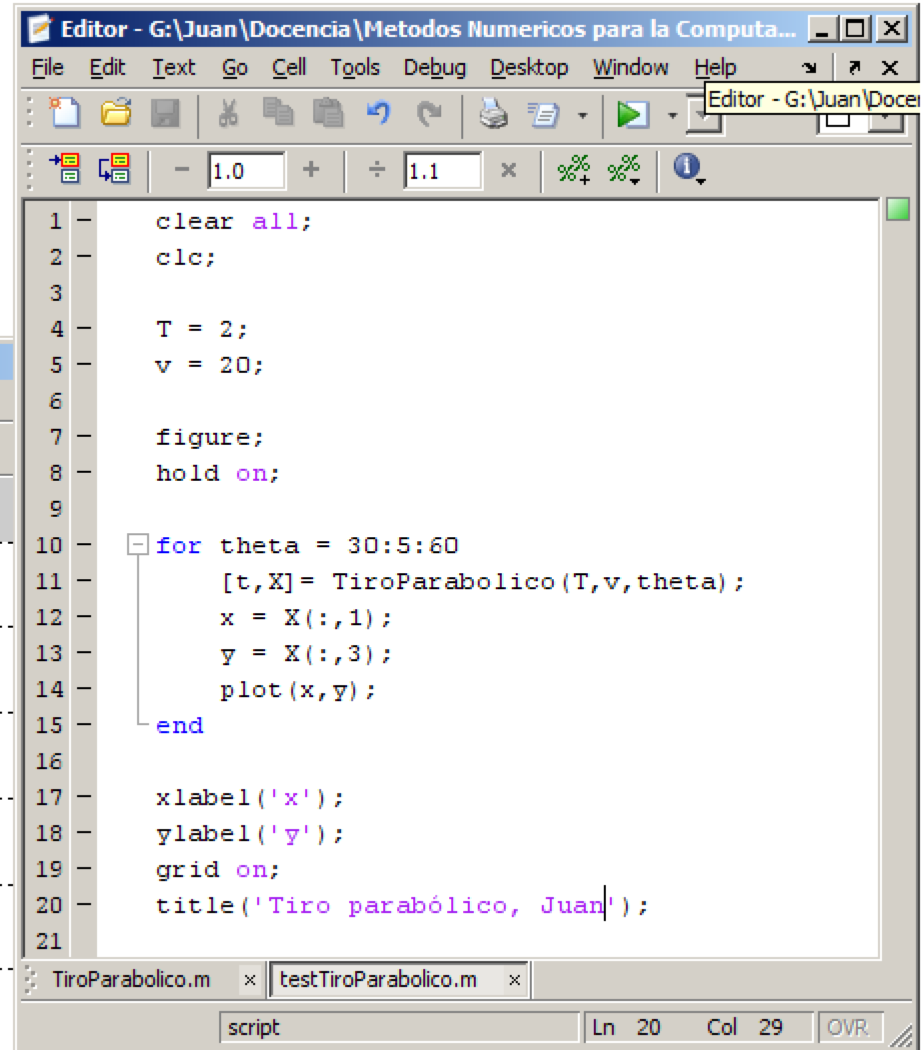
$$\frac{dx_2}{dt} = 0$$

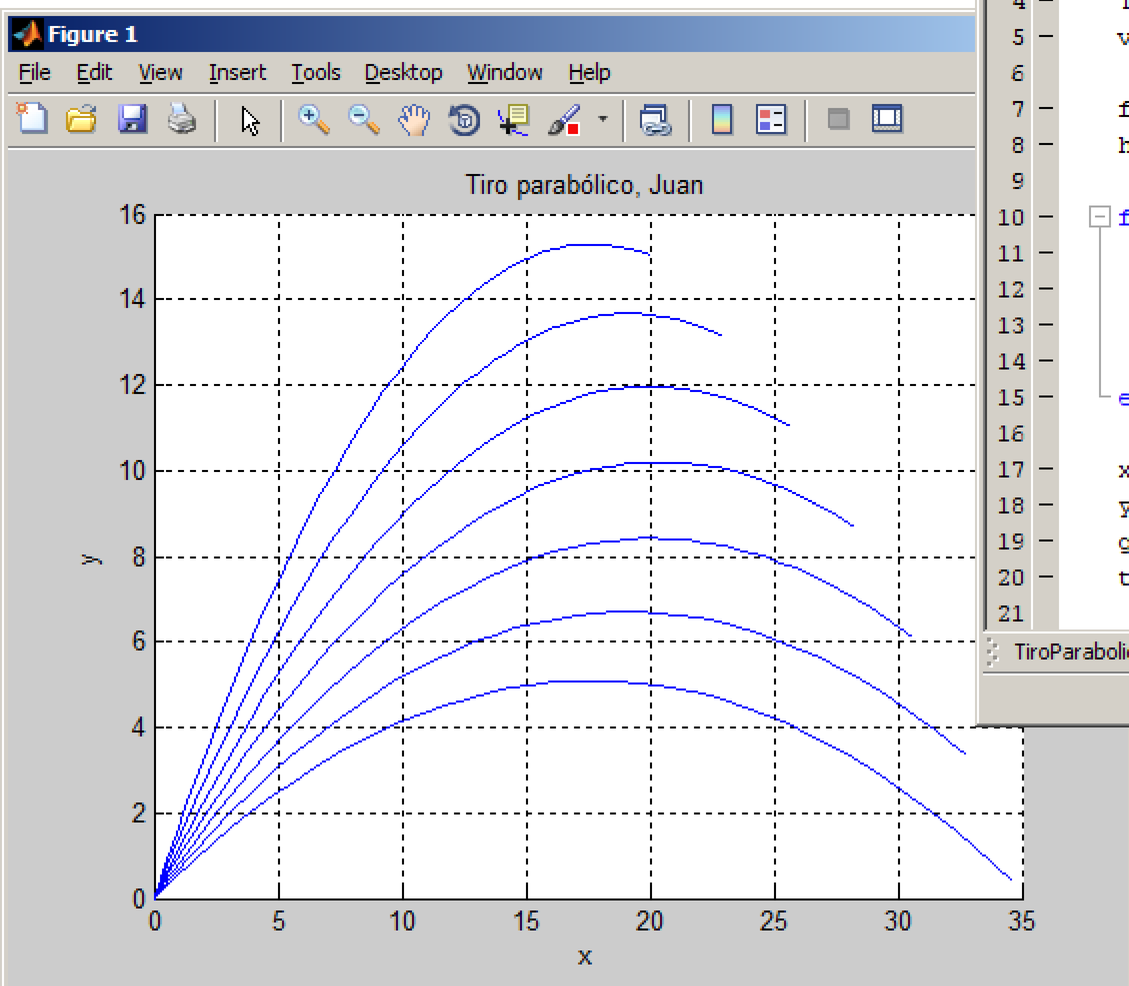
$$\frac{dx_3}{dt} = x_4$$

$$\frac{dx_4}{dt} = -g$$



Programa lanzador y gráficos





La velocidad de salida del proyectiles es siempre la misma, 20 m/s, y el tiempo de vuelo computado es de 2 sg, pero el ángulo de elevación diferente.

¿Qué hemos aprendido?

- Utilizar MATLAB para resolver ecuaciones diferenciales ordinarias de primer orden.
- Resolver sistemas de ecuaciones de primer orden.
- Transformar ecuaciones de orden superior en sistemas de primer orden y resolverlos.

Qué debe entregar el alumno

- Cada alumno entregará en el Campus Virtual una memoria en PDF o Word en la que estará contenida una descripción del trabajo realizado, incluyendo descripción, el listado MATLAB o C de la actividad realizada y la captura de pantalla de las gráficas o imágenes generadas. Para autentificar las imágenes cuando sea posible el alumno incluirá su nombre en cada imagen mediante la función title().
- En principio la tarea quedará abierta para su entrega hasta cierta fecha que se indicará.
- Se puede trabajar en grupo en el Laboratorio, pero la memoria elaborada y entregada será individual.

Bibliografía

1. Finite Difference Methods for Ordinary and Partial Differential Equations, R. J. LeVeque, SIAM, 2007
2. Introduction to Numerical Methods in Differential Equations, M.H. Holmes, Springer, 2000.
3. Tutorial de ODE: http://www.math.tamu.edu/reu/comp/matode.pdf

4. Ejemplos MATLAB

http://www.sc.ehu.es/sbweb/energias-renovables/MATLAB/numerico/diferencial/diferencial.html

http://www.sc.ehu.es/sbweb/energias-renovables/MATLAB/numerico/diferencial/diferencial_1.html