ECUACIONES DIFERENCIALES EN MATLAB

Daniel Parcero Sánchez

Rocío Salgueiro Fernández

Ecuaciones Diferenciales en Matlab®

Matlab ofrece varios algoritmos numéricos para resolver una extensa variedad de ecuaciones diferenciales. Esta demostración enseña la formulación y solución para dos tipos distintos de ecuaciones diferenciales usando Matlab.

- **El problema del valor inicial.**
- **Problemas de valor limite.**

a)El Problema del Valor Inicial

Para esto utilizaremos la ecuación de Van Der Pol, en Matlab VANDERPOLDEMO es una función que define la ecuación de Van Der Pol.

La ecuación de Van Der Pol tiene una larga historia de utilización tanto en física como en biología. Por ejemplo, en biología, Fitzhugh y Nagumo ampliaron la ecuación como un modelo para la acción neuronal. En cuanto a la física, la ecuación ha sido utilizada en sismología para modelizar las dos placas en una falla geológica.

En la dinámica, el oscilador de Van Der Pol es un oscilador no conservativo con amortiguamiento no lineal Se desarrolla en el tiempo de acuerdo con la siguiente ecuación diferencial de 2º orden:

$$\frac{d^2x}{dt^2} - \mu(1 - x^2)\frac{dx}{dt} + x = 0$$

Donde x es la coordenada de posición - que es función del tiempo t - y μ es un parámetro escalar que indica la fuerza de la amortiguación. Se puede demostrar a través del Teorema de Lienard que existe un ciclo límite para el oscilador de Van Der Pol no forzado, por lo que es un ejemplo de un sistema de Lienard.

Resultados para el oscilador no forzado

Dos regímenes de interés para las características del oscilador no forzado son:

- Cuando $\mu = 0$, es decir, no hay ninguna función de amortiguación, la ecuación es:

  $$\frac{d^2x}{dt^2} + x = 0$$

  Esta es una forma del oscilador armonico simple, y por tanto se conserva la energia mecanica.

- Cuando $\mu > 0$, el sistema entrará en un ciclo límite, donde la energía sigue siendo conservada, pero parte de la energía mecanica se perdera en forma de calor. Cerca del origen $x = dx / dt = 0$ el sistema es inestable, y lejos del origen el sistema se amortigua.

El oscilador forzado Van Der Pol

El oscilador de Van Der Pol forzado o dirigido, toma la función original, y añade una función de corrección *A*.sen *(ωt)* para dar una ecuación diferencial de la forma:

$$\frac{d^2x}{dt^2} - \mu(1 - x^2)\frac{dx}{dt} + x - A\sin(\omega t) = 0,$$

Donde *A* es la amplitud, t el tiempo y ω es su velocidad angular.

type vanderpoldemo
dydt = [y(2); Mu(1-y(1)^2)*y(2)-y(1)];*
`La ecuación se escribe como un sistema de dos funciones ODE de primer orden. Estas son evaluadas para distintos valores del parámetro Mu. Para una integración más rápida, elegimos un método de solución basado en el valor del parámetro Mu.`

Para Mu = 1, cualquiera de los métodos de solución ODE Matlab puede resolver la ecuación de Van Der Pol eficientemente. el método de solución ODE45 usado a continuación es un ejemplo. La ecuación es resuelta en el dominio [0, 20].

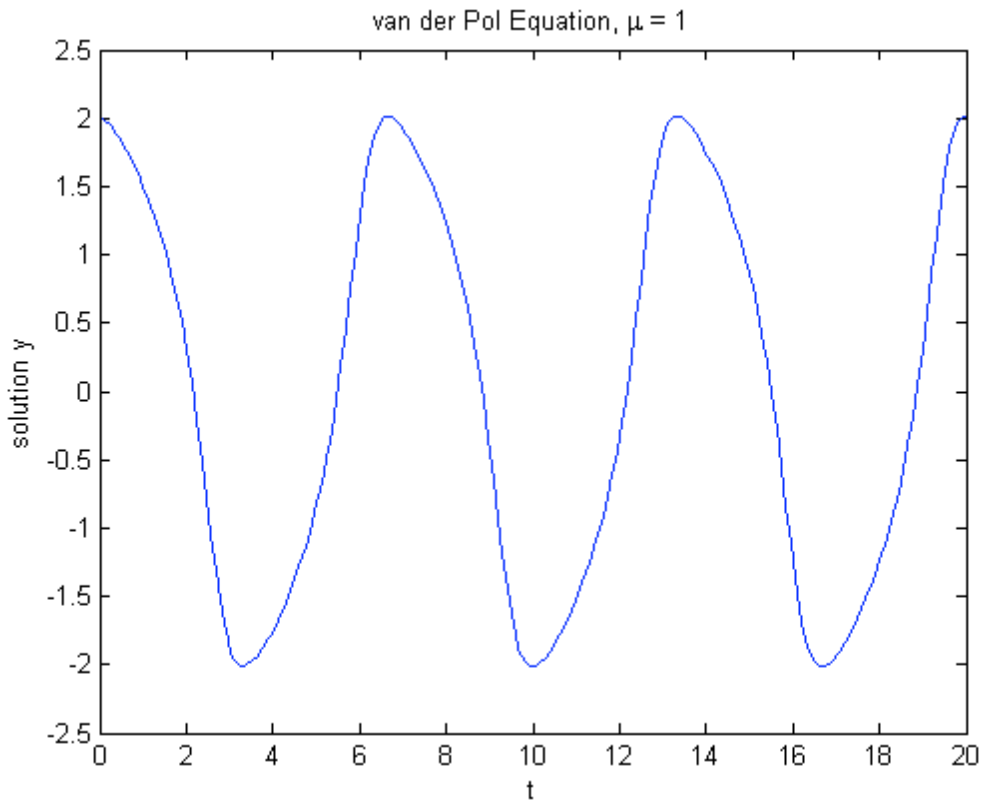
- tspan: es el vector que especifica el intervalo de integración.
– y0: es el vector de condiciones iniciales.
- Mu: es un parámetro, al que le damos el valor que queramos, 1 en este caso..
- ODE: es un método de solución que emplea Matlab para resolver ecuaciones diferenciales ordinarias.
– [t,y]: Nos dice el método empleado por Matlab para la resolución del problema, y las variables de las que depende (ode, tspan y y0), ya explicadas.

```
tspan = [0, 20];
y0 = [2; 0];
Mu = 1;
ode = @(t,y) vanderpoldemo(t,y,Mu);
[t,y] = ode45(ode, tspan, y0);
```

`%` **Gráfico de la solución**

- plot: Comando que ordena a Matlab representar graficamente la función que le damos.
- xlabel: Nos permite ponerle el titulo que queramos al eje x.
- ylabel: Nos permite ponerle el titulo que queramos al eje y.
- title: Nos permite ponerle a la gráfica el titulo que queramos.

```
plot(t,y(:,1))
xlabel('t')
ylabel('Solución y')
title('Ecuación Van Der Pol, \mu = 1')
```



Para magnitudes más grandes de Mu, el problema se vuelve más rígido, es decir, mas dificil de resolver numericamente. Para una integración rapida son necesarios métodos numéricos especiales como ODE15S, ODE23S, ODE23T, y ODE23TB, que pueden resolver problemas rígidos eficientemente.

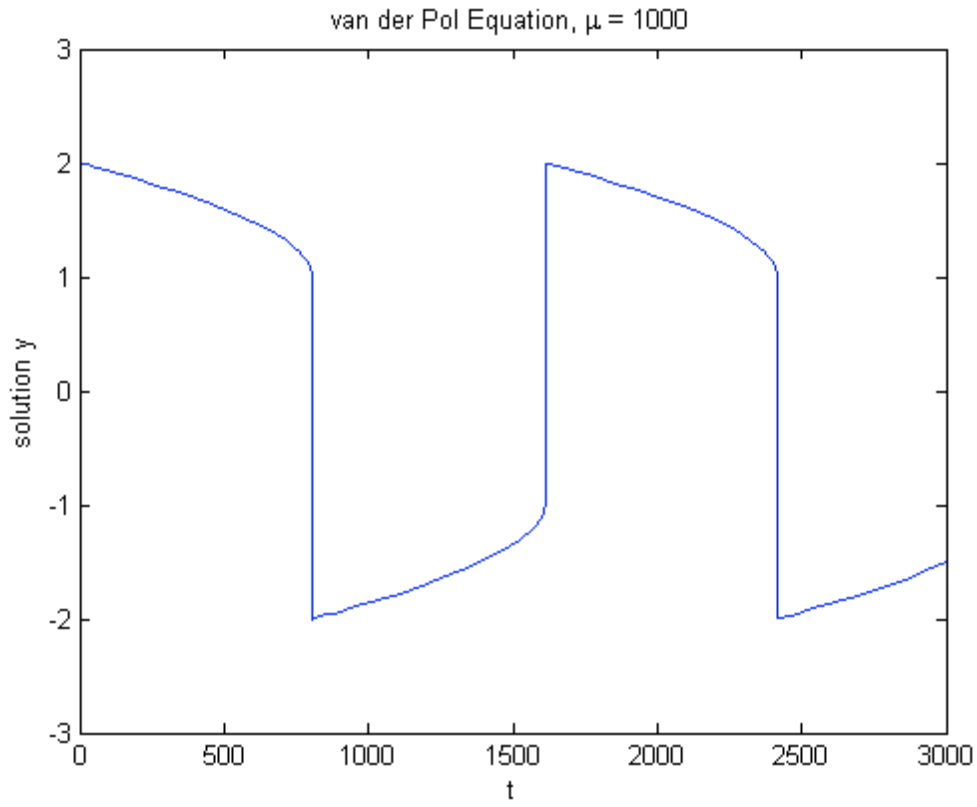
Aquí hay una solución a la ecuación de Van Der Pol para Mu = 1000 usando ODE15S.

- tspan: es el vector que especifica el intervalo de integración.
– y0: es el vector de condiciones iniciales.
- Mu: es un parámetro, al que le damos el valor que queramos, 1000 en este caso.
- ODE: es un método de solución que emplea Matlab para resolver ecuaciones diferenciales ordinarias.
– [t,y]: Nos dice el método empleado por Matlab para la resolución del problema, y las variables de las que depende (ode, tspan y y0), ya explicadas.

```
tspan = [0, 3000];
y0 = [2; 0];
Mu = 1000;
ode = @(t,y) vanderpoldemo(t,y,Mu);
[t,y] = ode15s(ode, tspan, y0);
```

- plot: Comando que ordena a Matlab representar graficamente la función que le damos.
- title: Nos permite ponerle a la gráfica el titulo que queramos.
- axis: Establece los valores minimos y maximos de x y de y.
- xlabel: Nos permite ponerle el titulo que queramos al eje x.
- ylabel: Nos permite ponerle el titulo que queramos al eje y.

```
plot(t,y(:,1))
title('Ecuación Van Der Pol, \mu = 1000')
axis([0 3000 -3 3])
xlabel('t')
ylabel('Solución y')
```



b)Problemas de Valor Límite:

BVP4C resuelve problemas de valor límite para ecuaciones diferenciales ordinarias.

La función de ejemplo TWOODE tiene una ecuación diferencial escrita como un sistema de dos EDO de primer orden.

```
type twoode
dydx = [ y(2); -abs(y(1)) ];
```

TWOBC tiene las condiciones de límite para TWOODE.

```
type twobc
res = [ ya(1); yb(1) + 2 ];
```

Antes de utilizar BVP4C, tenemos que proporcionar una suposición para la solución que queremos representar en la gráfica. El método de solución entonces adapta la grafica dado que refina la solución.

BVPINIT ensambla la suposición inicial en la forma que el método de solución BVP4C tendrá. Para una malla inicial de [0 1 2 3 4] y una suposición constante de y(x) = 1, y'(x) = 0, invocamos BVPINIT de este modo:

```
solinit = bvpinit ([0 1 2 3 4],[1; 0]);
```

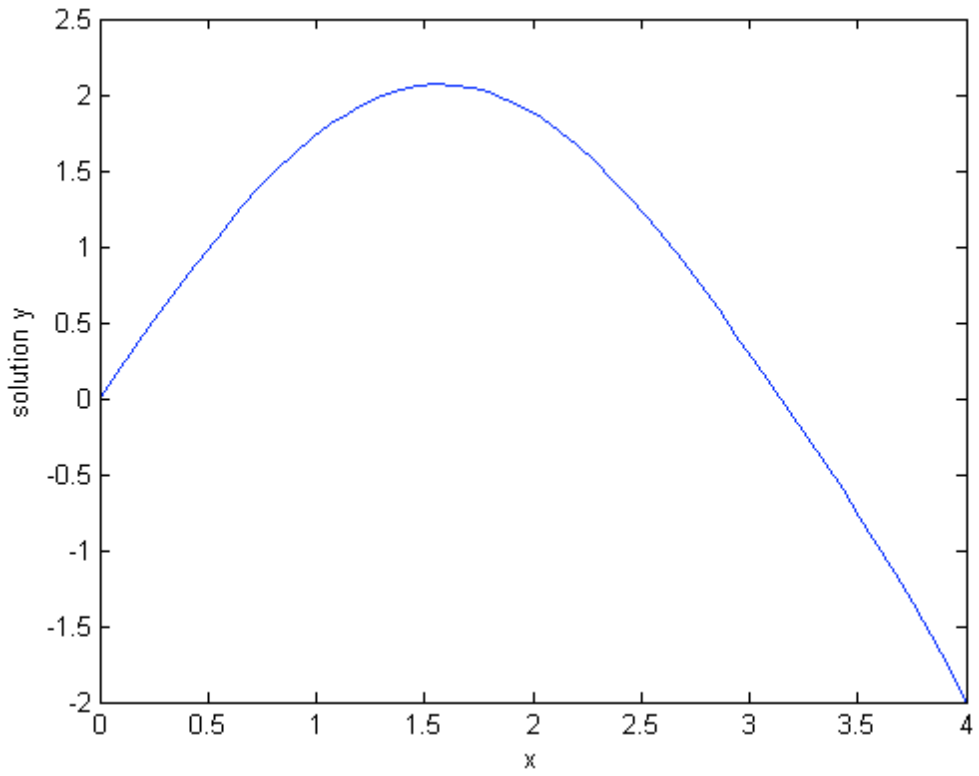
Con esta suposición inicial, podemos resolver el problema con BVP4C.

La solución (abajo) se evalua en los puntos xint usando DEVAL y representando graficamente.

- Linspace: (0,4,50); Proporciona 50 puntos igualmente espaciados entre 0 y 4.
– sol: Nos dice el método empleado por Matlab para la resolución del problema, y las variables de las que depende (twoode, twobc y solinit).
- plot: Comando que ordena a Matlab representar graficamente la función que le damos.
- xlabel: Nos permite ponerle el titulo que queramos al eje x.
- ylabel: Nos permite ponerle el titulo que queramos al eje y.
– hold on: mantiene en la ventana gráfica los dibujos anteriores.
- bvpinit: declara las condiciones iniciales
- deval: evalúa los resultados
- solinit: estimado de valores en la frontera inicial
- plot(xint,yint(1,:),'r'): representa graficamente la función en color rojo

```
sol = bvp4c(@twoode, @twobc, solinit);
xint = linspace(0, 4, 50);
```

```
yint = deval(sol, xint);
plot(xint, yint(1,:),'b');
xlabel('x')
ylabel('Solución y')
hold on
```

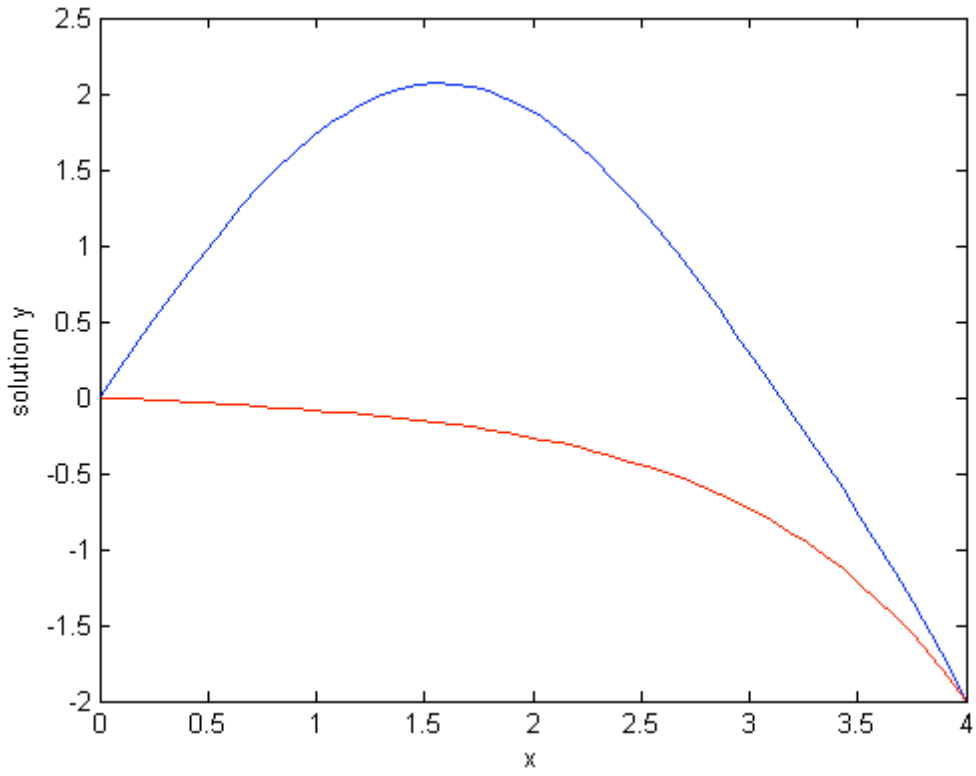


Este problema particular del valor límite tiene exactamente dos soluciones. La otra solución es obtenida para una suposición inicial de:

```
y(x) = -1, y'(x) = 0
```

y representando graficamente como antes.

```
solinit = bvpinit([0 1 2 3 4],[-1; 0]);
sol = bvp4c(@twoode,@twobc,solinit);
xint = linspace(0,4,50);
yint = deval(sol,xint);
plot(xint,yint(1,:),'r');
hold off
```



Además podemos añadir algunos comandos más para modificar las gráficas:

- grid on: comando que pone una cuadrícula en la gráfica.
- grid off: retira la cuadrícula puesta anterirmente e nun gráfico.
- gtext('gráfica'): sitúa un texto en cualquier lugar del gráfico.
- withebg([0.85 1 1]: cambia el color del fondo de la gráfica
- axis equal: pone la misma escala en los ejes.
- plot(xint,yint(1,:),linewidth,2): modifica el ancho del trazado de la gráfica
- plot(xint,yint(1,:),'markeredgecolor','k'): modifica el color de los ejes.