Oscilador de Van de Pol

Ramses Pacheco Ortiz

14 de Abril Del 2018

1 Introducción

Esta práctica representa la actividad 8 del curso,la cual trabajamos con el modelo de Van der Pol prosiguiendo un poco con el tema de oscilaciones no conservativas mas adelante explicaremos mas a profundidad este modelo.

Para realizar esta actividad utilizamos la misma herramienta de python(jupyter lab) de las actividades 6 y 7 ,usando algunas bibliotecas de las actividades pasadas como Numpy,Matplotlib,Pylab y en particular odeint de scipy utilizado en actividades anteriores para realizar las integraciones.

2 Modelo de Van der Pol

El modelo de van der pol es un oscilador no conservativo con un amortiguamiento,su comportamiento esta dado por la siguiente expresion matematica:

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

En esta expresion x es la posicion que esta en funcion del tiempo t y u es una constante que indica la no linealidad y la fuerza del amortiguamiento.

2.1 Historia

El oscilador de van der Pol fue descrito por el ingeniero y físico Balthasar van der Pol mientras trabajaba en Philips. Van der Pol encontró oscilaciones estables, que llamó oscilaciones de relajación,conocidas en la actualidad como ciclos límite, en circuitos que usaban válvulas de vacío.

Van der Pol y su colega, van der Mark, informaron en septiembre de 1927 de Nature,que para determinadas frecuencias aparecía un ruido irregular, siempre cerca de las frecuencias de acoplamiento. Fue uno de los primeros descubrimientos experimentales de la Teoría del caos.

La ecuación de van der Pol tiene una larga historia en física y biología. Por ejemplo, en biología, Fitzhugh y Nagumo aplicaron la ecuación a un campo

bidimensional en el modelo de FitzHugh-Nagumo para describir el potencial de acción de las neuronas. También se ha usado en sismología para modelar el comportamiento de dos placas en una falla.

2.2 Forma Bidimensional

El teorema de Liénard prueba que el sistema tiene un ciclo límite. Aplicando la transformación de Liénard:

$$y = x - x^3/3 - \dot{x}/\mu.$$

donde el ’.’ indica derivada, la ecuación se puede escribir en forma bidimensional:

$$\dot{x} = \mu\left(x - \frac{1}{3}x^3 - y\right)$$
$$\dot{y} = \frac{1}{\mu}x.$$

2.3 Resultados del oscilador no forzado

Hay dos regímenes de funcionamiento interesantes para el oscilador no forzado:

- Cuando $= 0$, no hay amortiguamiento, y la ecuación queda:

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

- Cuando > 0, el sistema alcanzará un ciclo límite, en el que se conservará la energía. Cerca del origen $x = dx/dt = 0$ el sistema es inestable, y lejos del origen hay amortiguamiento.

2.4 El oscilador de van der Pol forzado

Utilizando una fuente de excitación sinusoidal Asin(t) la ecuación diferencial queda:

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

en la que A es la amplitud de la ecuación de onda y su velocidad angular.

2.5 Hamiltoniano para el oscilador de van der pol

Tambien se puede escribir el hamiltoniano del oscilador del van der pol independiente del tiempo aumentando el sistema de unos de cuatro dimensiones, para esto primero tenemos que escribir el sistema por medio de ecuaciones diferenciales no lineales como las siguientes:

$$\ddot{x} - \mu(1 - x^2)\dot{x} + x = 0,$$
$$\ddot{y} + \mu(1 - x^2)\dot{y} + y = 0.$$

De las cuales podemos obtener el hamiltoniano de la siguiente manera:

$$H(x, y, p_x, p_y) = p_x p_y + xy - \mu(1 - x^2)yp_y,$$

donde px y pyson los momentos respectivos a x y a y

2.6 Circuitos electricos

En 1920 van der pol constuyo el oscilador con un triodo o tetrodo. despues de que Reona Esaki el tunes del diodo en 1957,haciendo que el oscilador de van der pol con el circuito eléctrico fuera mas sencillo.

La expresión matematica del circuito esta dada por la siguiente ecuación:

$$\dot{V} = \frac{1}{C}\left(-\phi(V) - W\right)$$

$$\dot{W} = \frac{1}{L}V$$

La cual a su vez puede ser escrita como:

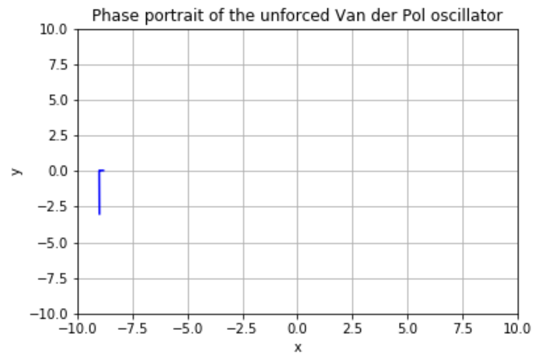
$$\ddot{V} - \frac{1}{C}\left(\alpha - 3\gamma V^2\right)\dot{V} + \frac{1}{LC}V = 0$$

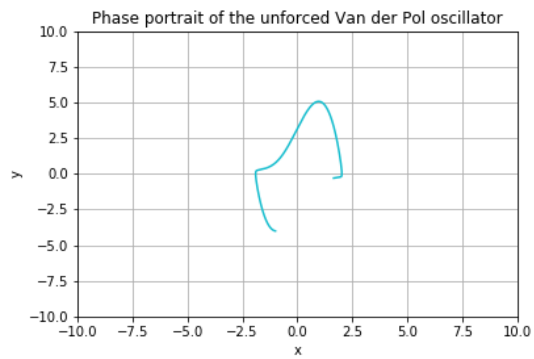
3 Exploracion de las soluciones del modelo en el Espacio Fase

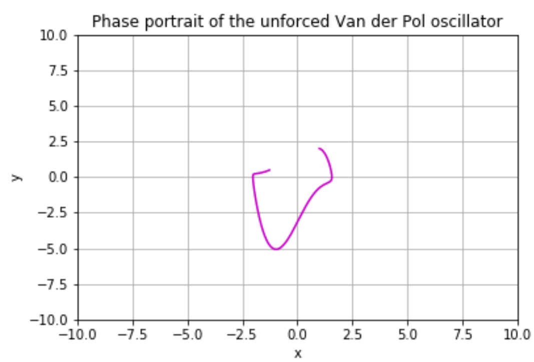
para realizar esta parte,cmabiaremos las condiciones iniciales, y se mantendra constante la mu=3.0,para llevar a cabo las siguientes graficas que mostraremos nos basamos en el codigo de la actividad anterio.

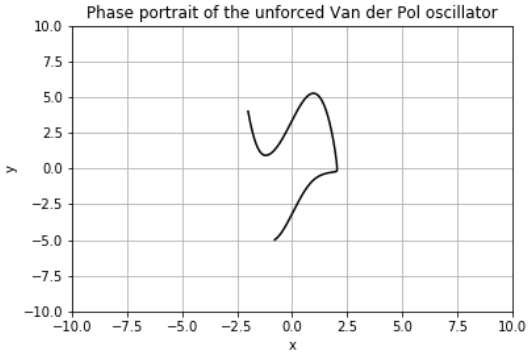
Cambiamos las condiciones iniciales para comprobar que verdaderamente convergia a la misma figura sin importar las condiciones iniciales.

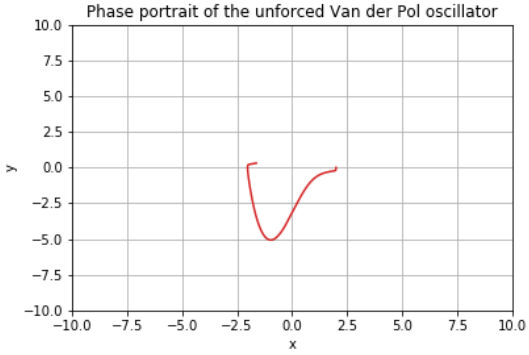
A continuación mostraremos las graficas que se llevaron a cabo:


Phase portrait of the unforced Van der Pol oscillator
y
x
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0


Phase portrait of the unforced Van der Pol oscillator
y
x
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0


Phase portrait of the unforced Van der Pol oscillator
y
x
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0 −7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0





como se puede observar sin inportar las condiciones que le pusimos siempre converge a la misma figura

4 Resultados y Discusión

En esta actividad realizamos 4 graficas basandonos en archivo de texto de wikipedia del oscilador van der pol

A continuacion se mostrara el codigo que se utilizo para realizar las graficas:

- Primera grafica

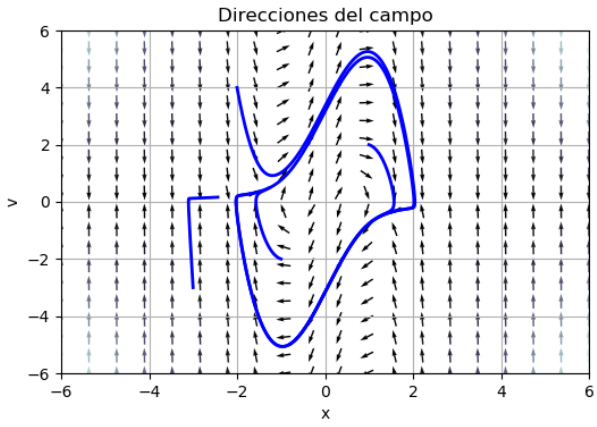
```
from numpy import *
import pylab as p
# Definition of parameters
mu= 3.0
def dX_dt(X, t=0):
    xpri=X[1]
    ypri=mu*(1-X[0]**2)*X[1]-X[0]
    return array([xpri, ypri ])
```

```
from scipy.integrate import odeint
from scipy import array
import matplotlib.pyplot as plt

stoptime = 5.0
numpoints = 5000

#Tiempo
t = [stoptime * float(i) / (numpoints - 1) for i in range(numpoints)]

#Condiciones iniciales
xi = -3.0
vi = -3.0

#Resolver ecuaciones
x, y = odeint(dX_dt,(xi,vi),t).T

with open('parte1.dat', 'w') as archivo:
    for t1, x1,y1 in zip(t, x, y):
        print (t1, x1,y1 ,file=archivo)
```

En la imagen anterio se hixo los mismo pero cambiando las condiciones iniciales(x1,y2) y el nombre del archivo hasta realizar 5 partes, al final graficas estas cinco partes para juntarlas en una sola:

```
from numpy import loadtxt
from numpy import *
import pylab as p
from matplotlib.font_manager import FontProperties
from scipy import integrate
%matplotlib inline


#-----------------------------------------------------------
# define a grid and compute direction at each point
nb_points   = 20

x = linspace(-6, 6, nb_points)
y = linspace(-6, 6, nb_points)

X1 , Y1  = meshgrid(x, y)                       # create a grid
px1, py1 = dX_dt([X1, Y1])                      # compute growth rate on the gridt
M = (hypot(px1, py1))                           # Norm of the growth rate
M[ M == 0] = 1.                                 # Avoid zero division errors
px1 /= M                                        # Normalize each arrows
py1 /= M

#-----------------------------------------------------------
# Drow direction fields, using matplotlib 's quiver function
# I choose to plot normalized arrows and to use colors to give information on
# the growth speed

Q = p.quiver(X1, Y1, px1, py1, M, pivot='mid', cmap=p.cm.bone)
p.grid()

lw=2

t1, x1, y1 = loadtxt('parte1.dat', unpack=True)
t2, x2, y2 = loadtxt('parte2.dat', unpack=True)
t3, x3, y3 = loadtxt('parte3.dat', unpack=True)
t4, x4, y4 = loadtxt('parte4.dat', unpack=True)
t5, x5, y5 = loadtxt('parte5.dat', unpack=True, skiprows=1500)

p.plot(x1, y1, 'b', linewidth=lw)
p.plot(x2, y2, 'b', linewidth=lw)
p.plot(x3, y3, 'b', linewidth=lw)
p.plot(x4, y4, 'b', linewidth=lw)
p.plot(x5, y5, 'b', linewidth=lw)

p.xlim(-6, 6)
p.ylim(-6, 6)

p.xlabel('x')
p.ylabel('v')

p.title('Direcciones del campo')
p.savefig('reto.png', dpi=100)
```

A continuacion se mostrara la grafica resultante:

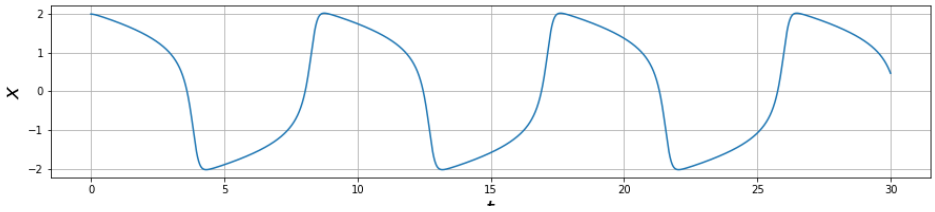


- Segunda grafica

  primeramente se definiola funcion van el por,despues en el codigo que se mostrara a continuacion muestra muchas varaibles debido a que son varias las condiciones iniciales

```
from scipy.integrate import odeint
from scipy import array, arange,pi, sin
import matplotlib.pyplot as plt

def dX_dt(X,t):
    x = X[0]
    y = X[1]
    a = X[0]
    b = X[1]
    c = X[0]
    d = X[1]
    e = X[0]
    f = X[1]
    g = X[0]
    h = X[1]
    j = X[0]
    k = X[1]
    l = X[0]
    n = X[1]
    o = X[0]
    p = X[1]
    q = X[0]
    r = X[1]
    s = X[0]
    v = X[1]
    dx = y
    dy = u*(1 - x*x)*dx - x
    return array([dx, dy])
```

  A continuacion se mostrara la grafica:

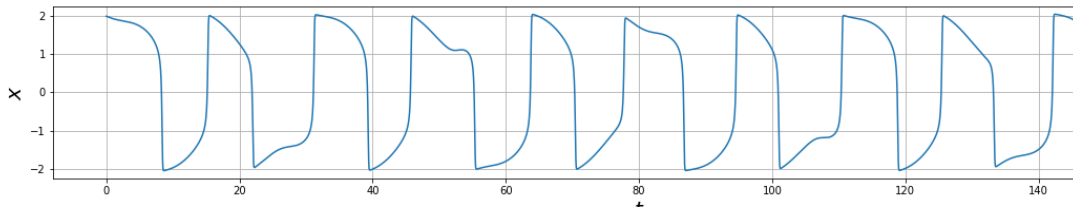


- Tercera grafica

  Para realizar la siguiente grafica se utilizo elsigueinte codigo:

```
import numpy as np

def dX_dt(X,t):
    x = X[0]
    y = X[1]
    dx = y
    dy = u*(1 - x*x)*dx - x + M*np.sin(w*t)
    return array([dx, dy])

t0 = 0
tmax = 160
pastemps = 0.01
t = arange(t0, tmax, pastemps)

#Condiciones iniciales
x0 = 2.0
y0 = 0.0

#Parametros
u = 8.53
M = 1.2
w = np.pi/5.0

#Solución a la ecuación diferencial
x, y = odeint(dX_dt,(x0,y0),t).T

#Trayectoria del ocilador
p1 = plt.figure(figsize=(20,3.0))
plt.grid()
plt.xlabel('$t$', fontsize = 20)
plt.ylabel('$x$', fontsize = 20)
plt.plot(t, x)
plt.show()

p1.savefig('3rafig.png')
```

La grafica resultante es:



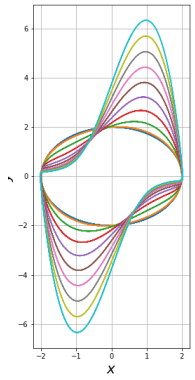
- Cuarta y ultima grafica

  Primero tubimos que juntar cada solucion de la ecuacion diferencial y despues graficamos las trayectorias de fase

```
t0 = 0
tmax = 30
pastemps = 0.01
t = arange(t0, tmax, pastemps)

#Mu
u = 0.01

#Solución a la ecuación diferencial
a, b = odeint(dX_dt,(x0,y0),t).T

u = 0.1

#Solución a la ecuación diferencial
c, d = odeint(dX_dt,(x0,y0),t).T

u = 0.5

#Solución a la ecuación diferencial
e, f = odeint(dX_dt,(x0,y0),t).T

u = 1.0

#Solución a la ecuación diferencial
g, h = odeint(dX_dt,(x0,y0),t).T

u = 1.5

#Solución a la ecuación diferencial
j, k = odeint(dX_dt,(x0,y0),t).T

u = 2.0

#Solución a la ecuación diferencial
l, n = odeint(dX_dt,(x0,y0),t).T

u = 2.5

#Solución a la ecuación diferencial
o, p = odeint(dX_dt,(x0,y0),t).T

u = 3.0

#Solución a la ecuación diferencial
q, r = odeint(dX_dt,(x0,y0),t).T

u = 3.5

#Solución a la ecuación diferencial
s, v = odeint(dX_dt,(x0,y0),t).T

u = 4.0

#Solución a la ecuación diferencial
x, y = odeint(dX_dt,(x0,y0),t).T
```

```
#Trayectoria de fase
p2 = plt.figure(figsize=(4,9))
plt.grid()
plt.xlabel('$x$', fontsize = 20)
plt.ylabel('$y$', fontsize = 20)
plt.plot(a, b)
plt.plot(c, d)
plt.plot(e, f)
plt.plot(g, h)
plt.plot(j, k)
plt.plot(l, n)
plt.plot(o, p)
plt.plot(q, r)
plt.plot(s, v)
plt.plot(x, y)
#plt.xlim(-2.5,2.5)
#plt.ylim(-6.0,6.0)
plt.show()

p2.savefig('4tafig.png')
```

A continuacion se mostrara la figura resultante:



5 Conclusiones

Continuando con el tema de ecuacions diferanciales no lineales y comparando esta actividad con la anterior me paracio mas interesanta el sistema que trabajamos y respeto a las graficas mas padres, mas complicadas para modelarlas a mi paracer.

Me parace bastante interesante o mas bien sorprendio de todo lo que podemos hacer con python ya queesto nos abre un posiblidad muy extensa de desarrollarnos en nuestros estudios y la facilidad quenos brinda al momento de modelar cietos sistemas o resolver ciertas expresiones matematicas ya sea lineales o no lineales.

6 Bibliografia

- Van der Pol oscillator. consultado: 14 de abril del 2018 de wikipedia.sitio web:https://$en.wikipedia.org/wiki/Van_der_Pol_oscillator$
- Van der Pol oscillator. consultado: 14 de abril del 2018 de Scholarpedia.Sitio web: $http://www.scholarpedia.org/article/Van_der_Pol_oscillator$

7 Apéndice

- **Este ejercicio pareciera similar al desarrollado en las actividades 6 y 7. ¿Qué aprendiste nuevo?**

  un poco pero aprendi mucho mas de teoria que no conocia.

- **¿Qué fue lo que más te llamó la atención del oscilador de Van der Pol?**

  sin dudad alguna la teoria se me hizo muy interesante

- **Has escuchado ya hablar de caos. ¿Por qué sería importante estudiar este oscilador?** No,nuca habia escuchado sobre el,seria importante ya que esta relacionado con muchos ambitos de estudio que puede desarrollarse a su vez de diferentes puntos de vista.
- **¿Qué mejorarías en esta actividad?** Nada en absoluto
- **¿Algún comentario adicional antes de dejar de trabajar en Jupyter con Python?**

  fabulosa aplicacion

  Cerramos la parte de trabajo con Python ¿Que te ha parecido?

  Fenomenal, aprendi bastante sobre muchas cosas desde teoria,herramientas matematicas, herramientas computacionales,etc.