Universidad de Sonora

Licenciatura en Física

Física Computacional I

Actividad 6 - Sistema de resortes acoplados

Alumno:
José Gabriel Navarro I.

Profesor:
Carlos Lizarraga Celaya

15 de Marzo de 2018

1 Introducción

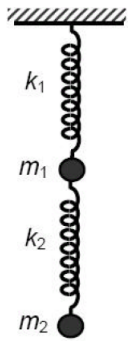
En el presente reporte se habla acerca de la sexta actividad realizada para la clase de Física Computacional I, el cual abarca el análisis de las ecuaciones de un sistema de resortes acoplados.

Primeramente se muestra una síntesis acerca de la teoría detrás de este sistema, esto basado en el artículo "Coupled Spring Equations" de Fay y Graham. En esta misma síntesis se incluye el código utilizado en Python con la nueva herramienta de Jupyter: Jupyter Lab, para resolver estos problemas de una manera numérica. Después de esto, se presenta el error relativo que existe entre el valor numérico que se encontró utilizando la librería scipy.integrate.odeint y la solución analítica dada por el artículo. Al final se presenta una conclusión de la actividad, así como la bibliografía utilizada para la investigación de los fundamentos y el apéndice.

2 Síntesis - Sistema de resortes acoplados

2.1 Introducción

En este artículo, se investiga uno de los problemas más interesantes de Mecánica, y que ahora normalmente se utiliza para la introducción al estudio de ecuaciones diferenciales. Este problema es el de dos resortes con dos masas puestas en serie, colgando del techo. Si suponemos que las fuerzas restauradoras de los resortes se comportan según la Ley de Hooke, estos dos grados de libertad nos dan un modelo de ecuaciones diferenciales lineales de segundo grado. Al sustituir una ecuación en la otra, el movimiento de las masas puede ser descrito por una ecuación diferencial lineal de cuarto grado.



Con estas ecuaciones, podemos investigar los movimientos de las dos masas, para saber si estas están sincronizadas (es decir en fase), o si son opuestas (en desfase). Además, también se puede observar gráficamente la periodicidad, la amplitud, la fase, y otros conceptos al modificar los parámetros de este modelo.

2.2 El modelo de los resortes acoplados

Como se menciono anteriormente, el modelo consiste de dos resortes y dos masas. Un resorte, con una constante k_1, esta colgado del techo con una masa m_1 colgando de ella. De aquí, cuelga otro resorte con una constante de k_2 y debajo de ella cuelga una masa m_2. Al dejarlo en reposo, los resortes se estiran una distancia, a la que llamaremos: x_1 y x_2.

2.2.1 Asumiendo la Ley de Hooke

Si asumimos que el sistema se mueve con oscilaciones pequeñas, podemos asumir que los resortes tendrán una fuerza restauradora dada por la Ley de Hooke, dada de la forma: $-k_1 l_1$ y $-k_2 l_2$ en donde l_1 y l_2 son las elongaciones o comprensiones de los dos resortes. Como m_1 esta atada a los dos resortes, en esta actuan las dos fuerzas restauradoras, mientras que m_2 solamente "siente" la fuerza restauradora del segundo resorte. Sin fricción, la segunda ley de Newton para estas dos masas son de la siguiente forma:

$$m_1\ddot{x}_1 = -k_1 x_1 - k_2(x_1 - x_2)$$
$$m_2\ddot{x}_2 = -k_2(x_2 - x_1)$$

Para encontrar una ecuación para x_1 que no involucre a x_2, resolvermos la ecuación de x_2, y sustituyendo esta ecuación en las ecuaciones anteriores llegamos a la siguiente ecuación diferencial de cuarto grado:

$$m_1 m_2 x_1^{(4)} + (m_2 k_1 + k_2(m_1 + m_2))\ddot{x}_1 + k_1 k_2 x_1 = 0$$

Y si se hace ese mismo proceso pero para x_2, se llega a la misma ecuación anterior. Solamente las posiciones y velocidades iniciales se necesitan para poder determinar la solución.

2.2.2 Algunos ejemplos con masas idénticas

2.1 Describe el movimiento para un sistema de resortes con $k_1 = 6$ *y* $k_2 = 4$ *con condiciones iniciales de* $(x_1(0), \dot{x}_1(0), x_2(0), \dot{x}_2(0))$=$(1, 0, 2, 0)$.

Resolviendo el problema de una forma analítica, podemos llegar a que las ecuaciones para las posiciones de las masas son:

$$x_1(t) = cos\sqrt{2}t$$
$$x_2(t) = 2cos\sqrt{2}t$$

El movimiento esta sincronizado, y por lo tanto las masas se mueven en fase una con la otra. Solamente tienen amplitud diferentes. El retrato de fase de estas son elipses, y como solamente varían en amplitud, al graficar sus posiciones una contra otra, se obtiene una linea recta. A continuación se presenta el código utilizado para encontrar numéricamente estos resultados y las graficas correspondientes:

Para graficar los datos, solamente se utilizo la linea de código de lectura de datos: *t, x1, xy, x2, y2, er1, er2 = loadtxt('dosresortes2_ 1.dat', unpack=True)*, y se graficaron los datos como ya se ha echo anteriormente en otras practicas.

```
def vectorfield(w, t, p):
    """
    Definimos las ecuaciones diferenciales para el sistema de doble masa-resorte.
    Arguments:
        w : Vector del estado de las variables
                  w = [x1,y1,x2,y2]
        t :  Tiempo
        p :  Vector de los parametros:
                  p = [m1,m2,k1,k2,L1,L2,b1,b2]
    """
    x1, y1, x2, y2 = w
    m1, m2, k1, k2, L1, L2, b1, b2 = p

    #Creamos f = (x1',y1',x2',y2')
    f = [y1,
         (-b1 * y1 - k1 * (x1 - L1) + k2 * (x2 - x1 - L2)) / m1,
         y2,
         (-b2 * y2 - k2 * (x2 - x1 - L2)) / m2]
    return f
```

```
#Usamos la funcion ODEINT para resolver las ecuaciones diferenciales definidas por el vector
from scipy.integrate import odeint
import numpy as np

#Valor de los parametros
# Masas:
m1 = 1.0
m2 = 1.0
# Constante del resorte
k1 = 6.0
k2 = 4.0
# Longitudes naturales
L1 = 0
L2 = 0
# Coeficientes de fricción
b1 = 0.0
b2 = 0.0

# Condiciones iniciales
# x1 and x2 son las pocisiones iniciales(contando la longitud de L), y y1 y y2 son las velocidades
x1 = 1.0
y1 = 0.0
x2 = 2.0
y2 = 0.0

# Parametros de la ED
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 50.0
numpoints = 250
```

```
#Creamos los valores del tiempo, entre mas puntos damos, mejores se veran las graficas.
t = [stoptime * float(i) / (numpoints - 1) for i in range(numpoints)]

# Ponemos a las variables en un vector.
p = [m1, m2, k1, k2, L1, L2, b1, b2]
w0 = [x1, y1, x2, y2]

# Llamamos a la funcion para resolver la ED
wsol = odeint(vectorfield, w0, t, args=(p,),
              atol=abserr, rtol=relerr)

with open('dosresortes2_1.dat', 'w') as f:
    # Imprimimos en el documento la solución
    for t1, w1 in zip(t, wsol):
        print (t1, w1[0], w1[1], w1[2], w1[3],np.abs((w1[0]-(np.cos(np.sqrt(2.0)*t1)))/(np.cos(np.sqrt(2.0
```

```
#Graficamos la solución

import numpy
from numpy import loadtxt
from pylab import figure, plot, xlabel, grid, hold, legend, title, savefig, ylabel
from matplotlib.font_manager import FontProperties
import matplotlib.pyplot as plt
%matplotlib inline

t, x1, xy, x2, y2, er1, er2 = loadtxt('dosresortes2_1.dat', unpack=True)

figure(1, figsize=(6, 4.5))

xlabel('t')
ylabel('x')
grid(True)
lw = 1.5

plot(t, x1, 'navy', linewidth=lw)
plot(t, x2, 'indianred', linewidth=lw)
plt.axhline(0, color='black',linewidth=0.5)
plt.axvline(0, color='black',linewidth=0.5)

legend((r'$x_1$', r'$x_2$'), prop=FontProperties(size=16))
title('Posición de las Masas para\nel Sistema Doble Resorte-Masa')
savefig('Ej2_11.png', dpi=100)
```

Las gráficas obtenidas son:

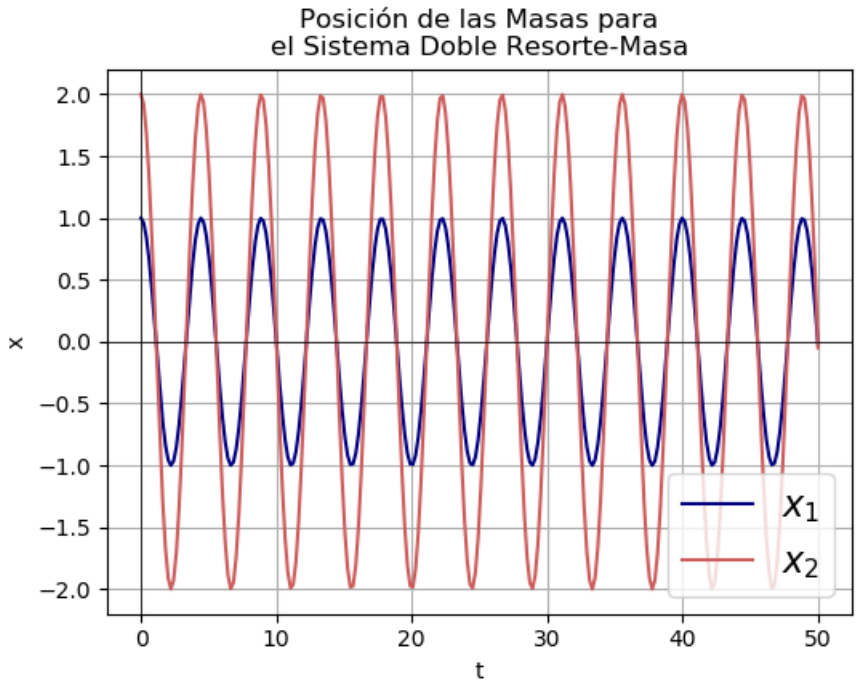
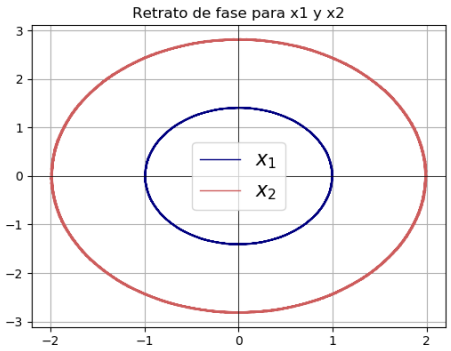


Figure 1: Graficas de x_1 y x_2

(a) x_1 vs x_2



(b) Retrato de fase de x_1 y x_2

2.2 Describe el movimiento para un sistema de resortes con $k_1 = 6$ *y* $k_2 = 4$ *con condiciones iniciales de* $(x_1(0), \dot{x}_1(0), x_2(0), \dot{x}_2(0))$=$(-2, 0, 1, 0)$.

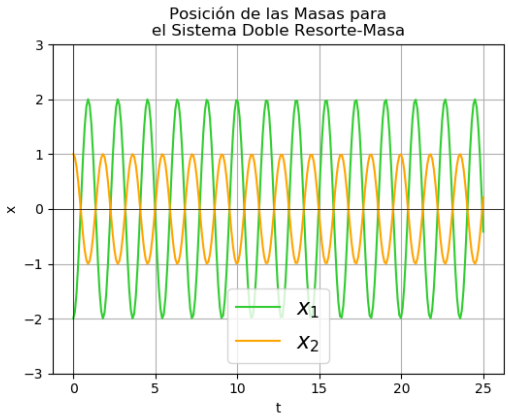
Para este caso, la primera masa se mueve hacia abajo mientras que la otra se mueve hacia arriba, tienen el mismo periodo, pero estan fuera de fase. La solución analítica es:

$$x_1(t) = -2cos2\sqrt{3}t$$
$$x_2(t) = cos2\sqrt{3}t$$

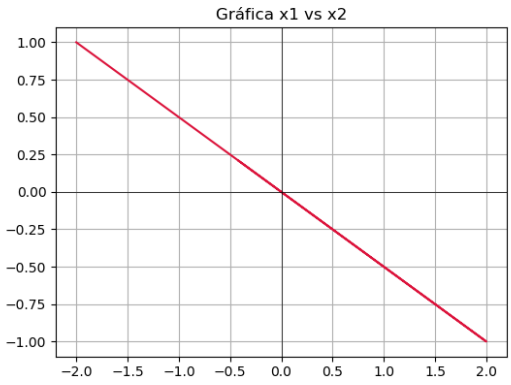
El código utilizado para este caso, es el mismo que se uso en el ejemplo anterior, y así fue con todos los ejemplos. Lo único que cambiaba eran los datos a utilizar y las graficas resultantes:

```
from scipy.integrate import odeint
import numpy as np

# Masas:
m1 = 1.0
m2 = 1.0
# Constante del resorte
k1 = 6.0
k2 = 4.0
# Longitudes naturales
L1 = 0
L2 = 0
# Coeficientes de fricción
b1 = 0.0
b2 = 0.0

# Condiciones iniciales
x1 = -2.0
y1 = 0.0
x2 = 1.0
y2 = 0.0

# Parametros de la ED
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 25.0
numpoints = 250
```

Las gráficas son:



(a) Gráfica de x_1 y x_2



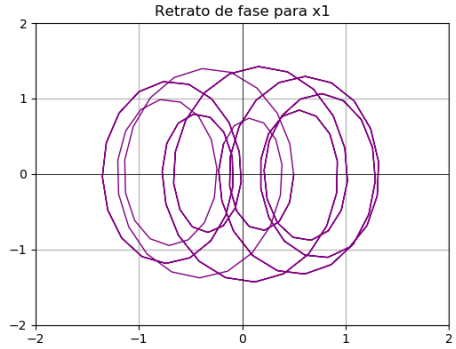
(b) Gráfica de x_1 vs x_2

2.3 Describe el movimiento para un sistema de resortes con $k_1 = 0.4$ *y* $k_2 = 1.808$ *con condiciones iniciales de* $(x_1(0), \dot{x}_1(0), x_2(0), \dot{x}_2(0))$=$(1/2, 0, -1/2, 7/10)$.

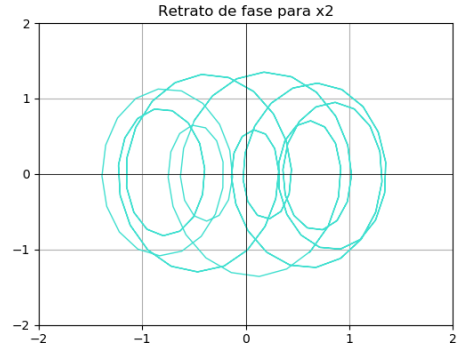
Con este ejemplo podemos observar como las condiciones iniciales solo afectan a la amplitud y fase de la soluciones, mientras que las constantes de los resortes determinan el periodo y frecuencia del fenómeno. Los retratos de fase son casi un movimiento periódico entre ellos, y si graficamos x_1 vs x_2 parece una curva de tipo Lissajous. En el código solamente se edito los datos como las constantes y las condiciones iniciales:

```
m1 = 1.0
m2 = 1.0
# Constante del resorte
k1 = 0.4
k2 = 1.808
# Longitudes naturales
L1 = 0
L2 = 0
# Coeficientes de fricción
b1 = 0.0
b2 = 0.0

# Condiciones iniciales
x1 = 0.5
y1 = 0.0
x2 = -0.5
y2 = 0.7

# Parametros de la ED
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 50.0
numpoints = 250
```

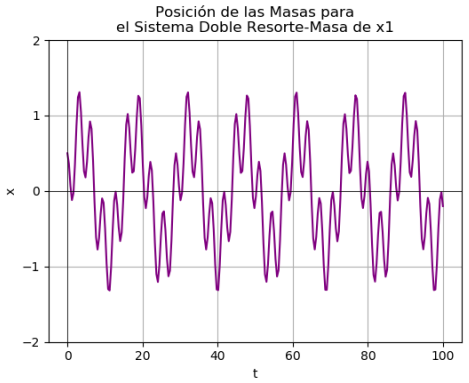
Las graficas resultantes fueron:



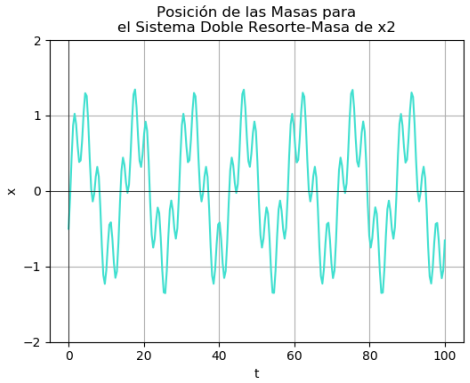
(a) Retrato de fase de x_1



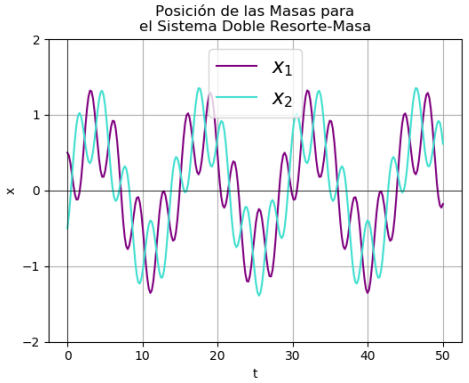
(b) Retrato de fase de x_2



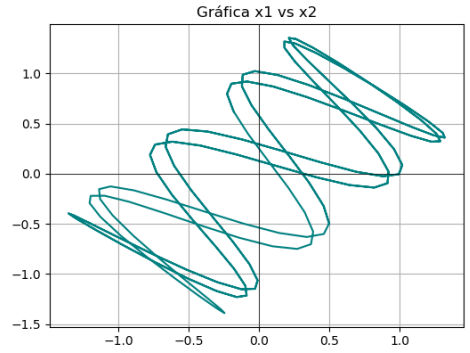
(c) Posición x_1 contra tiempo



(d) Posición x_2 contra tiempo



(e) Gráfica de x_1 y x_2



(f) Gráfica x_1 vs x_2

2.3 Amortiguamiento

Los problemas mas generales acerca de amortiguamiento es el de viscosidad, en donde la fuerza de amortiguamiento es proporcional a la velocidad. El amortiguamiento de la primera masa depende solamente de su velocidad y no en la de la segunda y viceversa. Si añadimos esta fuerza al modelo anteriormente descrito podemos obtener lo siguiente:

$$m_1\ddot{x}_1 = -\delta\dot{x}_1 - k_1x_1 - k_2(x_1 - x_2)$$
$$m_2\ddot{x}_2 = -\delta\dot{x}_2 - k_2(x_2 - x_1)$$

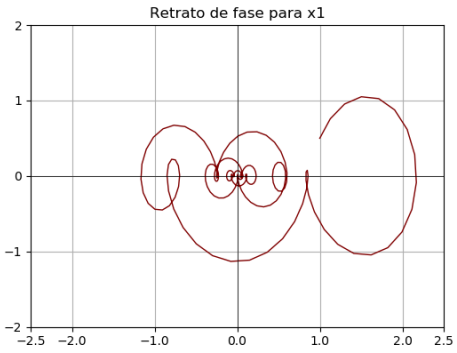
Se realiza el mismo proceso de obtener la ecuación de movimiento para una de las variables de posición, ya sea x_1 o x_2. Sustituimos esta ecuación en la anterior que le corresponde, y obtenemos de nuevo una ecuación diferencial lineal de cuarto grado para ambas posiciones de x.

2.4 Asume $m_1 = m_2 = 1$. *Describe el movimiento para un sistema de resortes con* $k_1 = 0.4$ *y* $k_2 = 1.808$, *con coeficientes de amortiguamiento* $\delta_1 = 0.1$ *y* $\delta_2 = 0.2$ *con condiciones iniciales de* $(x_1(0), \dot{x}_1(0), x_2(0), \dot{x}_2(0))$=$(1, 1/2, 2, 1/2)$.

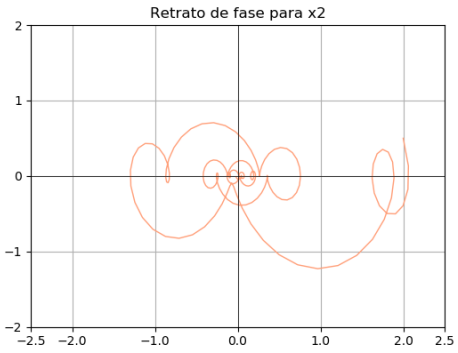
Como existe ahora un amortiguamiento, las amplitudes de ambos movimientos disminuyen conforme pasa el tiempo. Al graficar x_1 y x_2, podemos observar como se mueven casi en sincronía a pesar de que tienen distintas condiciones iniciales. El código para este caso es:

```
from scipy.integrate import odeint

# Masas:
m1 = 1.0
m2 = 1.0
# Constante del resorte
k1 = 0.4
k2 = 1.808
# Longitudes naturales
L1 = 0
L2 = 0
# Coeficientes de fricción
b1 = 0.1
b2 = 0.2

# Condiciones iniciales
x1 = 1.0
y1 = 0.5
x2 = 2.0
y2 = 0.5

# Parametros de la ED
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 50.0
numpoints = 250
```

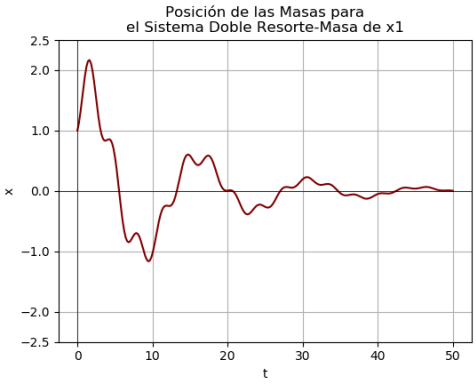
Las graficas resultantes fueron:



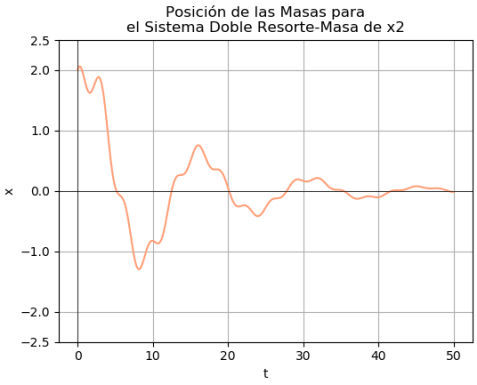
(a) Retrato de fase de x_1



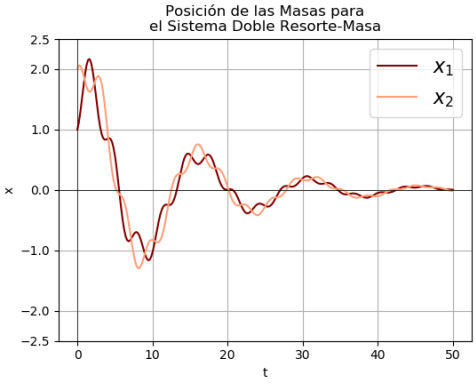
(b) Retrato de fase de x_2



(c) Posición x_1 contra tiempo



(d) Posición x_2 contra tiempo



(e) Gráfica de x_1 y x_2

(f) Gráfica x_1 vs x_2

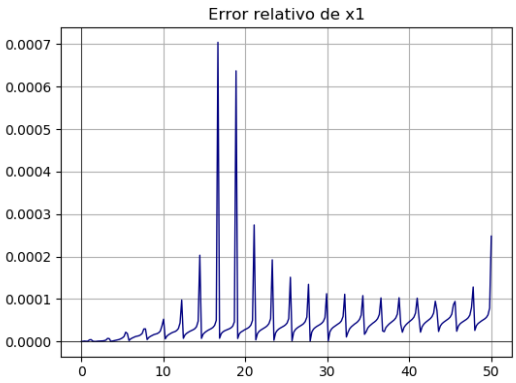
3 Resultado analítico y resultado numérico

Como en Python se realizo una aproximación numérica con los datos disponibles, y en el artículo se proporciona la solución analítica, es posible calcular el error relativo que hay entre nuestra aproximación y el valor real. Para hacer esto, en los primeros dos ejercicios, se imprimió en el archivo la definición del error relativo, restando el valor real al aproximado dividiendo esto entre el valor real.

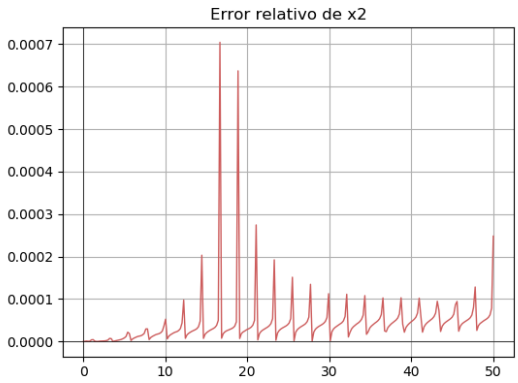
Para el primer caso se obtuvo el error relativo calculando:

```
np.abs((w1[0]-(np.cos(np.sqrt(2.0)*t1)))/(np.cos(np.sqrt(2.0)*t1))) \\
np.abs((w1[2]-(2.0*np.cos(np.sqrt(2.0)*t1)))/(2.0*np.cos(np.sqrt(2)*t1)))
```

En donde el primer comando es para x_1 y el segundo para x_2. Las graficas obtenidas del error relativo contra el tiempo, para el ejemplo 2.1 fueron:

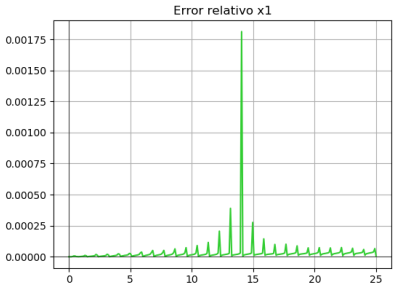


(a) Gráfica del error x_1 contra tiempo

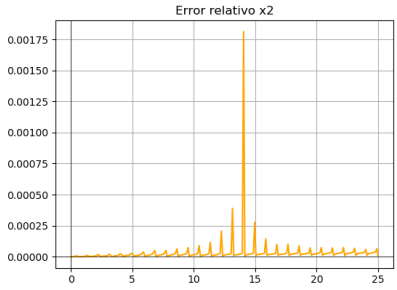


(b) Gráfica del error x_2 contra tiempo

Como se puede notar, los errores son muy pequeños, del orden de $7 * 10^{-3}$, y los errores entre x_1 y x_2 son idénticos. Este es el caso también para el ejemplo 2.2, donde el error es muy pequeño y casi igual para x_1 y x_2:



(a) Gráfica del error x_1 contra tiempo



(b) Gráfica del error x_2 contra tiempo

4 Conclusiones

Este nuevo lugar de trabajo, Jupyter Lab, es mucho mas limpio y eficiente que el notebook. Este permite realizar cálculos y operaciones de una manera mas rápida y eficiente, esto gracias al uso de la librería scipy.integrate.odeint, que permite hasta resolver ecuaciones diferenciales.

Es muy interesante el poder que tiene el lenguaje Python, sin mencionar su habilidad para poder leer archivos y tomarlos como datos numéricos sin especificarlo. Cabe mencionar que el tema de esta actividad fue muy interesante, ya que el semestre anterior se cursó la clase de Mecánica II, en donde se vio este fenómeno de resortes acoplados, pero con este repaso tanto teórico como analítico queda mucho más claro como funciona este fenómeno.

5 Bibliografía

- Temple H. Fay, Sarah Duncan Graham (2003) Coupled Spring Equations. Int. J. Educ. Math. Sci. Tech.. Vol. 34, No. 1, pp. 65-79. Recuperado de: http://math.oregonstate.edu/ gibsonn/Teaching/MTH323-010S15 /Supplements/coupled_spring.pdf

- Oscilaciones amortiguadas. (2010) Recuperado de:http://www.sc.ehu.es/sbweb/fisica/oscilaciones /amortiguadas/amortiguadas.htm

Imagenes utilizadas:

- http://srv2.fis.puc.cl/mediawiki/index.php/Osciladores_Acoplados_(Fiz0312)

6 Apéndice

1. ¿En general te pareció interesante esta actividad de modelación matemática? ¿Qué te gustó mas? ¿Qué no te gustó?

Me gusto mucho esta actividad, este tema es uno de los que mas me gusto en Mecánica II, y poder repasarlo ahora con programación es muy interesante y divertido. Me gusto mucho las gráficas generadas por Python, sobre todo los retratos de fase, pero me hubiera gustado otras referencias acerca del tema.

2. La cantidad de material te pareció ¿bien?, ¿suficiente?, ¿demasiado?

La cantidad de material al principio me parecio que era mucha, sobre todo con el cambio de Jupyter Notebook a Lab, pero una vez empezando la práctica me di cuenta de que no era mucho problema. La cantidad es suficiente.

3. ¿Cuál es tu primera impresión de Jupyter Lab?

Me gusto mucho más que Jupyter Notebook, tiene un diseño que me parece mejor y mas fácil de entender. Sin mencionar que las librerías y funciones que se pueden utilizar ahí son muy buenas e interesantes.

4. Respecto al uso de funciones de SciPy, ¿ya habías visto integración numérica en tus cursos anteriores? ¿Cuál es tu experiencia?.

En Análisis Numérico ya habíamos visto dos tipos de integración numérica, pero no de esta manera. Se utilizaron los métodos del trapecio y de Simpson, con un programa en Fortran, por lo cual este modo de integración es algo nuevo para mi.

5. El tema de sistema de masas acopladas con resortes, ¿ya lo habías resuelto en tu curso de Mecánica 2?

Si se había resuelto anteriormente, pero no con tantas especificaciones. Se vio de una manera rápida y sencilla, ya que en ese momento apenas estaba llevando ecuaciones diferenciales.

6. ¿Qué le quitarías o agregarías a esta actividad para hacerla más interesante y divertida?

Siento que esta práctica ya es buena como es, solamente me gustaría que tuviera mas referencias referente al tema para tener un mejor contexto y sea mas fácil de comprender.