LICENCIATURA EN FÍSICA

DEPARTAMENTO DE FÍSICA

FÍSICA COMPUTACIONAL 1

Reporte 6

Alumna:

Brambilla Zamorano Fátima Fernanda

Fecha:
16/03/18

1 Coupled spring equations

1.1 Introducción

Los métodos clasicos para empezar ecuaciones diferenciales están cambiando rápidamente por técnicas de solución para una variedad de ecuaciones diferenciales, para enfatizar sistemas y aspectos más cualitativos de la teoría de ecuaciones diferenciales ordinarias. En particular, hay un enfasis en las ecuaciones no lineales, debido en gran medida a la amplia disponibilidad de algoritmos númericos altamente poderosos, y a capacidades gráficas que vienen con sistemás algebráicos computacionales, como lo son *Mathematica* y *Maple*. En el siguiente articulo se presentará un viejo problema que ahora se utiliza en los libros de texto. Se trata del problema de dos resortes y dos masas atados en serie, y colgando del techo. Bajo la suposición de que que las fuerzas restauradoras se comportan de acuerdo a la Ley de Hooke, este problema de dos grados de libertad esta modelado por dos parejas de ecuaciones diferenciales de segundo orden.

1.2 The coupled spring model

El modelo consiste en dos resortes y dos masas. Un muelle, que tiene una constante elástica k1, esta atado al techo y a un peso con masa m1 en su parte inferior. A este peso, se le ata un segundo resorte de constante elástica k2, y así mismo este resorte tiene en su parte inferior atado un peso de masa m2. Como se puede ver en la siguiente figura.

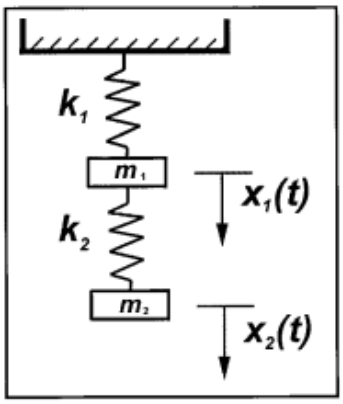


Figure 1. The coupled springs.

Permitiendo al sistema volver a su posición de equilibrio, medimos el desplazamiento del centro de masa de cada peso desde el equilibrio en función del tiempo, y denotamos estas mediciones como x1(t), y x2(t) respectivamente.

1.2.1 Asumiendo la Ley de Hooke

Bajo la suposición de pequeñas oscilaciones, las fuerzas de restauración son de la forma: -k1L1, y -k2L2, donde L1 y L2 son las elongaciones (o compresiones) de los dos resortes. Dado que la masa superior esta atada a ambos resortes, hay dos fuerzas restauradoras actuando sobre ella: una hacía arriba de la forma -k1x1, ejercida por la elongación (o comprensión) x1 del primer resorte; y otra hacía arriba de la forma -k2(x2 - x1) de la resistencia del segundo resorte a ser comprimido (o elóngado) por la cantidad (x2 - x1). La segunda masa solo "siente" la fuerza restauradora de la elongación (o compresión) del segundo resorte. Asumiendo que no hay fricción en el sistema, entonces la ley de Newton implica que las dos ecuaciones que representan el movimiento de las dos masas son las siguientes:

$$m1x'1 = -k1x1 - k2 * (x2 - x1) \tag{1}$$

$$m2x'2 = -k2 * (x2 - x1) \tag{2}$$

Sin embargo, estas ecuaciones no nos son de mucha ayuda para resolver el sistema, por lo que después de un largo procedimiento para obtener un par de ecuaciones que nos sean útiles, llegamos al siguiente sistema:

$$x'1 = u \tag{3}$$

$$u' = (-k1x1 - k2(x1 - x2))/m1 \tag{4}$$

$$x'2 = v \tag{5}$$

$$v' = (-k2(x2 - x1))/m2 \tag{6}$$

2 Procedimiento

Para resolver el sistema de manera númerica, nos apoyamos en *Jupyter Lab*, que es algo diferente al notebook, donde habíamos estado trabajando anteriormente, no obstante, al estar ya familiarizados con su empleo, no hubo ni un problema al empezar a trabajar en la plataforma mencionada. En primer lugar, definimos un vector con las ecuaciones previas, para poder trabajar los ejemplos que más adelante se nos pedía resolver y comparar con los resultados mostrados en el archivo de texto.

Figure 1: Celda 1

```
In [1]: def vectorfield(w, t, p):
            """
            Defines the differential equations for the coupled spring-mass system.

            Arguments:
                w :  vector of the state variables:
                          w = [x1,y1,x2,y2]
                t :  time
                p :  vector of the parameters:
                          p = [m1,m2,k1,k2,L1,L2,b1,b2]
            """
            x1, y1, x2, y2 = w
            m1, m2, k1, k2, L1, L2, b1, b2 = p

            # Create 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
```

Después de definir el vector, el cual podía reutilizarse para cada uno de los ejemplos que se resolverían más adelante, en una segunda celda se pusieron las condiciones iniciales y los datos indicados en el ejemplo 3.1 del archivo de texto. En esta celda también se escribió el siguiente código, para que el programa pudiera resolver el problema:

Figure 2: Celda 2

```
# ODE solver parameters
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 50.0
numpoints = 1250

# Create the time samples for the output of the ODE solver.
# I use a large number of points, only because I want to make
# a plot of the solution that looks nice.
t = [stoptime * float(i) / (numpoints - 1) for i in range(numpoints)]

# Pack up the parameters and initial conditions:
p = [m1, m2, k1, k2, L1, L2, b1, b2]
w0 = [x1, y1, x2, y2]

# Call the ODE solver.
wsol = odeint(vectorfield, w0, t, args=(p,),
              atol=abserr, rtol=relerr)

with open('two_springs.dat', 'w') as f:
    # Print & save the solution.
    for t1, w1 in zip(t, wsol):
        print  ( t1, w1[0], w1[1], w1[2], w1[3], file=f )
```

En la tercera celda, se importaron diferentes bibliotecas para la creación de gráficas, y el guardado de archivos de datos o imágenes, y se escribo el código para crear la primera gráfica, como se muestra en la siguiente figura.

Figure 3: Celda 3

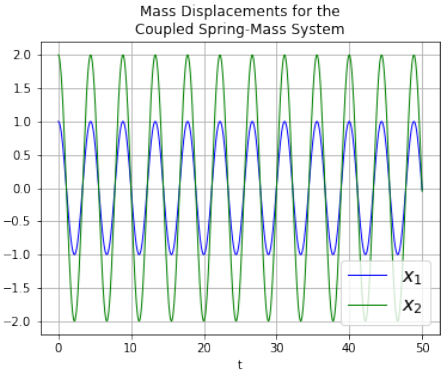
```
# Plot the solution that was generated
#Ejemplo 1 del archivo
from numpy import loadtxt
from pylab import figure, plot, xlabel, grid, hold, legend, title, savefig
from matplotlib.font_manager import FontProperties
%matplotlib inline
t, x1, xy, x2, y2 = loadtxt('two_springs.dat', unpack=True)

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

xlabel('t')
grid(True)
# hold(True)
lw = 1

plot(t, x1, 'b', linewidth=lw)
plot(t, x2, 'g', linewidth=lw)

legend((r'$x_1$', r'$x_2$'), prop=FontProperties(size=16))
title('Mass Displacements for the\nCoupled Spring-Mass System')
savefig('two_springs.png', dpi=100)
```

Las tres figuras anteriores muestran el código base que se uso para cada ejemplo del texto, de modo que a continuación se mostraran los resultados obtenidos, así como los fragmentos de código que se hayan usado a parte para cada ejemplo.

3 Resultados

En primer lugar, se mostrará la gráfica obtenida a partir del código mostrado en la figura 3. En esta gráfica se puede apreciar el comportamiento del sistema de resorte-masa.

Figure 4: Gráfica 1 ejemplo 2.1



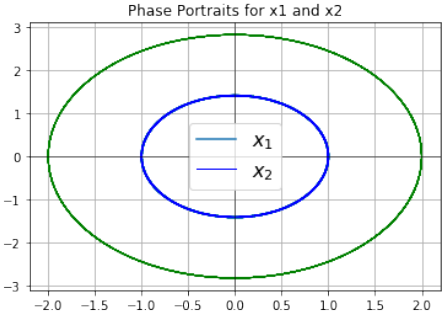
En seguida, se muestra otro fragmento de código, el cual fue utilizado para obtener una gráfica que mostrará los retratos de fase de x1 y x2.

Figure 5: Celda 4

```
# Plot the solution that was generated

from numpy import loadtxt
from pylab import figure, plot, xlabel, grid, hold, legend, title, savefig
from matplotlib.font_manager import FontProperties
import matplotlib.pyplot as plt
%matplotlib inline
import matplotlib
plot(x1,xy)
grid(True)
matplotlib.pyplot.axis('on')
t, x1, xy, x2, y2 = loadtxt('two_springs.dat', unpack=True)

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

grid(True)

lw = 1

plot(x1, xy, 'b', linewidth=lw)
plot(x2, y2, 'g', 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('Phase Portraits for x1 and x2')
savefig('G2.1b.png', dpi=100)
```

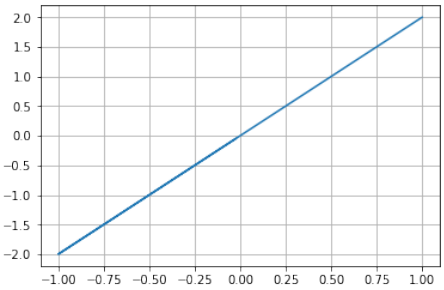
Esta sección de código, genero la siguiente gráfica:

Figure 6: Gráfica 2 ejemplo 2.1



La siguiente gráfica muestra el comportamiento de x1 contra x2:

Figure 7: Gráfica 3 ejemplo 2.1



Ahora, comparando las gráficas obtenidas mediante Jupyter Lab, con las gráficas mostradas en el archivo de texto, las cuales se muestra en la figura 8. Se puede apreciar que en este caso las gráficas obtenidas mediante la solución númerica son las mismas que las obtenidas mediante la solución teoríca del archivo de texto.

Figure 8: Gráficas del ejemplo 2.1

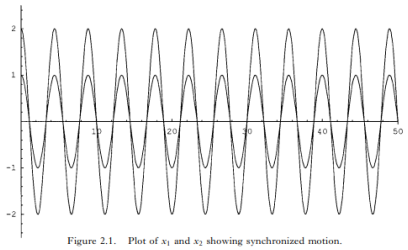


Figure 2.1. Plot of x_1 and x_2 showing synchronized motion.

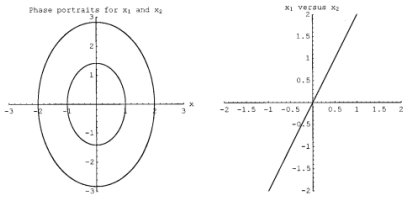
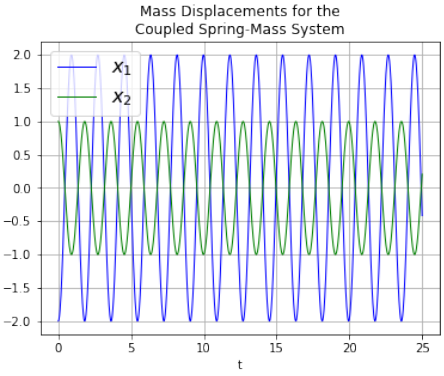


Figure 2.2. Plots for Example 2.1.

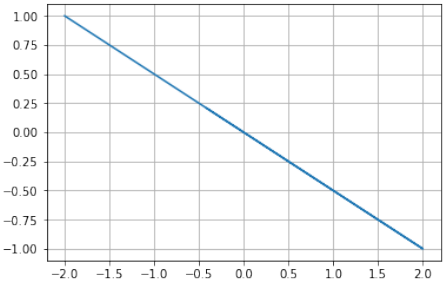
Siguiendo con los resultados obtenidos para el ejemplo 2.2, se mostrarán sus gráficas, ya que para obtenerlas se utilizaron los mismos códigos mostrados con anterioridad, cambiando únicamente los datos númericos.

Figure 9: Gráfica 1 del ejemplo 2.2



Como en el ejemplo anterior, esta primera gráfica representa el comportamiento del sistema resorte-masa, mostrando en distintos colores el comportamiento de la elongación del resorte uno y del resorte dos. A continuación se muestra la gráfica del comportamiento de x1 contra x2 para este ejemplo:

Figure 10: Gráfica 2 del ejemplo 2.2



Así como en el caso anterior, se puede ver que hay un comportamiento lineal entre ambas, sin embargo, en este ejemplo la pendiente mostrada es de carácter negativo, mientras que en el ejemplo 1 era de carácter positivo. Así como para el ejemplo anterior, también compararemos las gráficas obtenidas con la solución númerica con las esperadas del archivo de texto.

Figure 11: Gráficas del ejemplo 2.2

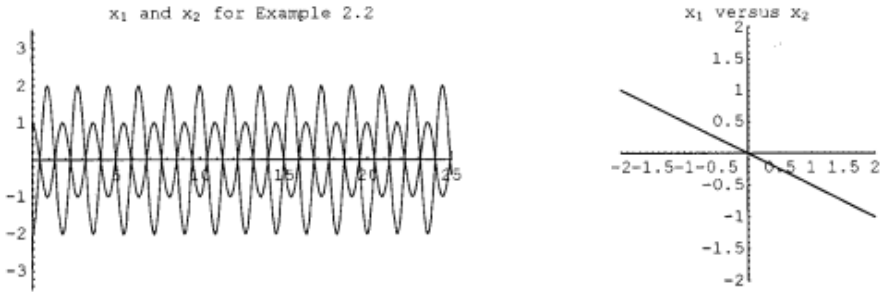
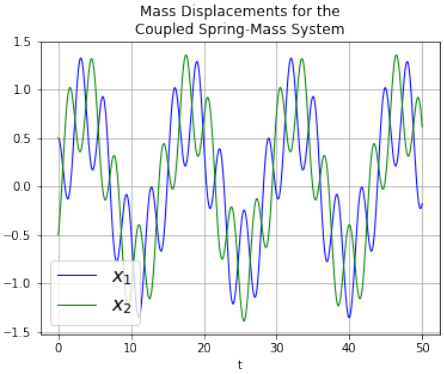


Figure 2.3. Plots for Example 2.2.

Comparando las gráficas obtenidas con las esperadas, se puede ver que son básicamente las mismas, lo que me lleva a pensar que la solución númerica obtenida esta bastante cercana a la solución teórica.

Ahora, pasamos a las gráficas obtenidas para los datos del ejemplo 2.3, como en los casos anteriores se presentará primero la gráfica del comportamiento del sistema resorte-masa

Figure 12: Gráfica 1 del ejemplo 2.3



Para este caso, el comportamiento del sistema es ligeramente diferente a los casos anteriores, esto puede deberse a que sus condiciones son bastante diferentes a como lo fueron en los casos anteriores, no obstante, aún puede verse el comportamiento oscilatorio del sistema. La figura 13 nos mostrará una gráfica del comportamiento de x1, con respecto a su primera derivada. Mientras que la figura 14 nos mostrará el comportamiento de x2 con respecto a su derivada también.

Figure 13: Gráfica 2 del ejemplo 2.3

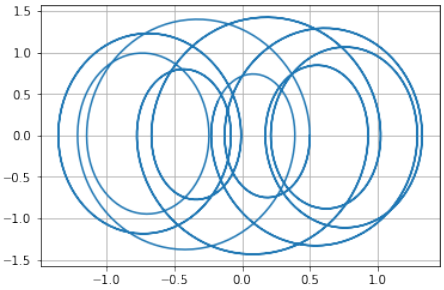
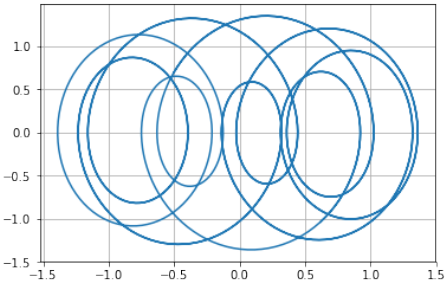
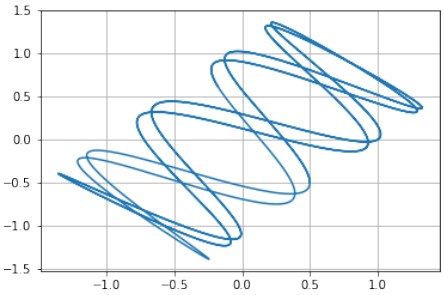


Figure 14: Gráfica 3 del ejemplo 2.3



Por último, se presenta la gráfica del comportamiento de x1 contra x2.

Figure 15: Gráfica 4 del ejemplo 2.3



Comparando las gráficas obtenidas con las esperadas, notamos una ligera diferencia en las gráficas de los comportamientos de x1 y x2 con respecto a sus derivadas, mientras que las gráficas de x1 contra x2, y la del comportamiento oscilatorio del sistema son las mismas

Figure 16: Gráficas del ejemplo 2.3

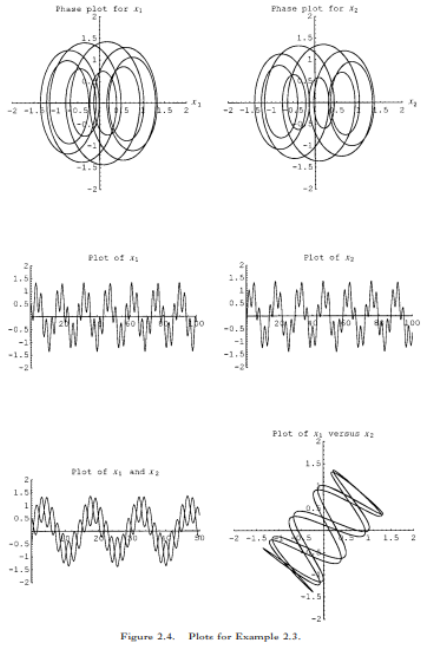
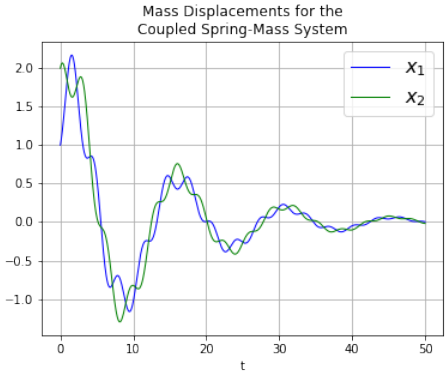


Figure 2.4. Plots for Example 2.3.

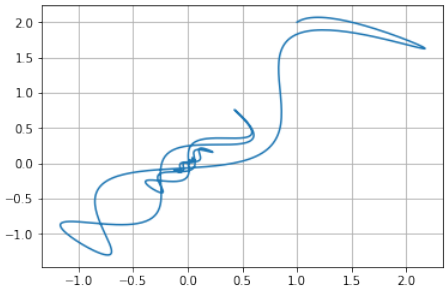
Por último las gráficas del ejemplo 2.4, como en los casos anteriores se mostrará primero la gráfica del comportamiento oscilatorio del sistema:

Figure 17: Gráfica 1 del ejemplo 2.4



La siguiente figura muestra el comportamiento de x1 contra x2, siendo en esta ocasión un movimiento un tanto más difícil de explicar, ya que, a diferencia de los casos anteriores, se tiene un movimiento que se puede considerar asimétrico y "no cerrado" como en el ejemplo 2.3

Figure 18: Gráfica 2 del ejemplo 2.4



Las siguientes figuras muestran los comportamientos de x1 y x2 con sus respectivas derivadas, en el mismo orden que se mencionan.

Figure 19: Gráfica 2 del ejemplo 2.4

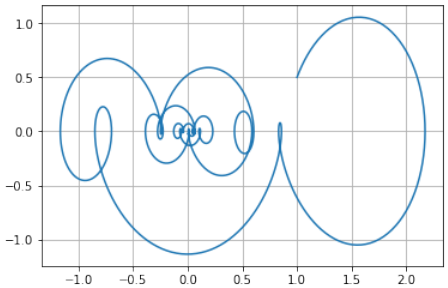
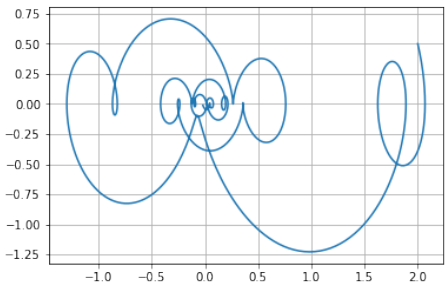
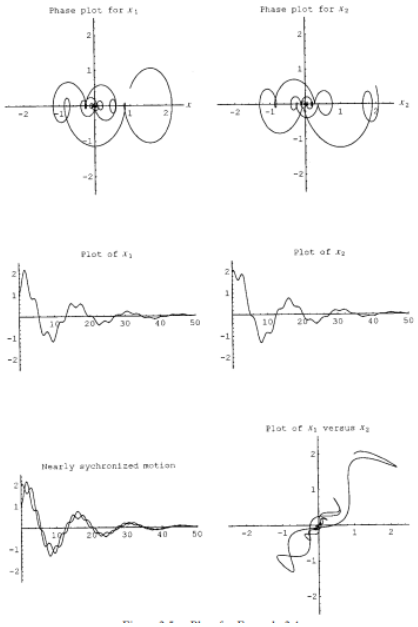


Figure 20: Gráfica 3 del ejemplo 2.4



En el caso de los comportamientos de x1 y x2 con sus respectivas derivadas, podemos decir lo mismo que se dijo anteriormente sobre la gráfica del comportamiento de x1 contra x2, que no hay simetría en sus movimientos. Para finalizar con esta parte, compararemos las gráficas obtenidas con las gráficas esperadas:

Figure 21: Gráficas del ejemplo 2.4



Comparando estas las gráficas obtenidas por medio de la solución númerica, y las esperadas del archivo de texto, podemos ver que se tratan de las mismas, es decir, como en los casos de los ejemplos anteriores, se obtuvieron los mismos resultados de la solución esperada.

4 Conclusiones

Los métodos númericos para resolver ecuaciones diferenciales, entre otras ecuaciones matemáticas, son una herramienta bastante útil y poderosa, la cual los físicos debemos empezar a dominar para poder emplearlas cómodamente en nuestro trabajo, ya que facilitan la obtención de resultados, que por otros métodos quizás no seríamos capaces de obtener.

5 Bibliografía

1. Temple H Fay, S. D. (2002). coupled springs equations. En T. H. Fay, Coupled Springs Equations (pág. 16).

2. Weckesser, W. (17 de 02 de 2018). SciPhy Cookbook. Obtenido de http://scipy-cookbook.readthedocs.io/items/CoupledSpringMassSystem.html

6 Ápendice

- ¿En general te pareció interesante esta actividad de modelación matemática?, ¿qué te gusto más?, ¿qué no te gusto?
  Al principio fue un poco difícil definir que era lo que teníamos que hacer, pero fue interesante trabajar en ello

- La cantidad de material te pareció ¿bien?, ¿suficiente?, ¿demasiado?
  Creo que la cantidad de material era justo el necesario

- ¿Cuál es tu primera impresión de *Jupyter Lab* ?
  que no es muy diferente de Jupyter Nootebook, no más allá de la apariencia

- Respecto al uso de funciones de SciPy, ¿ya habías visto integración número-ica en tus cursos anteriores?, ¿Cuál es tu experiencia?
  No lo había visto antes, y aunque mi experiencia no fue muy grata (más que nada porque no soy muy amiga de los lenguajes de programación), creo que es una herramienta bastante útil para la resolución de problemas númericos

- El tema de masas acopladas con resortes, ¿ya lo habías resuelto en tu curso de Mecánica 2?
  Si, es un problema que resolvimos teoricamente en el curso de Mecánica II

- ¿Qué le agregarías o quitarías a esta actividad para hacerla más interesante o divertida?
  No le cambiaría nada