Reporte: Actividad 6

Luisa Julieta Casas Cervantes

14 de Marzo 2018

1 Introducción

Con la intención de brindar un enfoque a fenómenos físicos, comenzaremos por estudiar oscilaciones basándonos en un texto provisto por el profesor en clase. El texto nos presenta distintos ejercicios a realizar, los cuales llevaremos a cabo en la plataforma de Jupyter lab basándonos en un artículo que maneja sistemas de masas acopladas de una manera similar, y lo adaptaremos a cada ejercicio.

2 Código Python

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
```

Para el ejemplo 2.1:

In [2]:
```
############EJEMPLO 2.1####################
# Use ODEINT to solve the differential equations defined by the vector field
from scipy.integrate import odeint
#Parametros para el ejemplo 1
# Parameter values
# Masses:
m1 = 1.0
m2 = 1.0
# Spring constants
k1 = 6.0
k2 = 4.0
# Natural lengths
L1 = 0.0
L2 = 0.0
# Friction coefficients
b1 = 0.0
b2 = 0.0

# Initial conditions
# x1 and x2 are the initial displacements; y1 and y2 are the initial velocities
x1 = 1.0
y1 = 0.0
x2 = 2.0
y2 = 0.0


# 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 )
```

Movimiento sincronizado x1 y x2:

In [4]:
```
########2.1 GRÁFICA#########
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)
```

Gráfica para x1 y x2:

In [5]:
```
############ 2.1 SEGUNDA GRÁFICA###############

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)
```

In [6]:
```
import matplotlib
plot(x1,x2)
grid(True)
matplotlib.pyplot.axis('on')
```

Ejemplo 2.2:

In [7]:
```
##########EJEMPLO 2.2################
from scipy.integrate import odeint
# Parameter values
# Masses:
m1 = 1.0
m2 = 1.0
# Spring constants
k1 = 6.0
k2 = 4.0
# Natural lengths
L1 = 0.0
L2 = 0.0
# Friction coefficients
b1 = 0.0
b2 = 0.0

# Initial conditions
# x1 and x2 are the initial displacements; y1 and y2 are the initial velocities
x1 = -2.0
y1 = 0.0
x2 = 1
y2 = 0.0


# ODE solver parameters
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 25.0
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 )
```

Movimiento sincronizado x1 y x2:

In [8]:
```
############## GRÁFICA EJEMPLO 2.2 ###########
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)
```

Gráfica para x1 y x2:

In [9]:
```
############### SEGUNDA GRÁFICA 2.2 ###########
import matplotlib
plot(x1,xy)
grid(True)
matplotlib.pyplot.axis('on')
```

In [10]:
```
import matplotlib
plot(x1,x2)
grid(True)
matplotlib.pyplot.axis('on')
```

Ejemplo 2.3:

In [11]:
```
################# EJEMPLO 2.3 ###########
from scipy.integrate import odeint

# Parameter values
# Masses:
m1 = 1.0
m2 = 1.0
# Spring constants
k1 = 0.4
k2 = 1.808
# Natural lengths
L1 = 0.0
L2 = 0.0
# Friction coefficients
b1 = 0.0
b2 = 0.0

# Initial conditions
# x1 and x2 are the initial displacements; y1 and y2 are the initial velocities
x1 = 0.5
y1 = 0.0
x2 = -0.5
y2 = 0.7


# 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 )
```

Movimiento sincronizado x1 y x2:

In [12]:
```
#########GRÁFICA EJEMPLO 2.3 #########
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)
```

Gráfica x1:

In [15]:
```
############ SEGUNDA GRÁFICA 2.3 ###############
import matplotlib
plot(x1,xy)
grid(True)
matplotlib.pyplot.axis('on')
```

Gráfica x2:

In [16]:
```
import matplotlib
plot(x2,y2)
grid(True)
matplotlib.pyplot.axis('on')
```

Gráfica x1 vs x2:

In [17]:
```
import matplotlib
plot(x1,x2)
grid(True)
matplotlib.pyplot.axis('on')
```

Ejemplo: 2.4

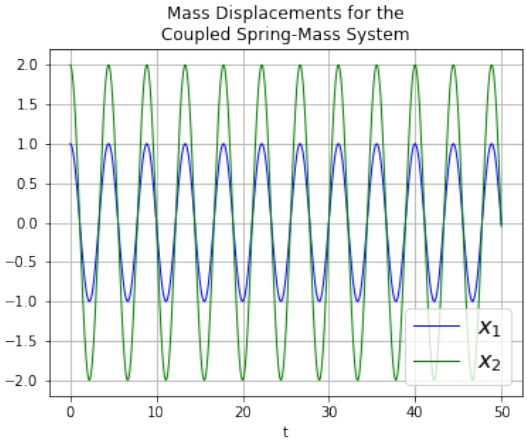
In [18]:
```
###############33 EJEMPLO 2.4 #########
from scipy.integrate import odeint
# Parameter values
# Masses:
m1 = 1.0
m2 = 1.0
# Spring constants
k1 = 0.4
k2 = 1.808
# Natural lengths
L1 = 0.0
L2 = 0.0
# Friction coefficients
b1 = 0.1
b2 = 0.2

# Initial conditions
# x1 and x2 are the initial displacements; y1 and y2 are the initial velocities
x1 = 1.0
y1 = 0.5
x2 = 2.0
y2 = 0.5


# 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 )
```

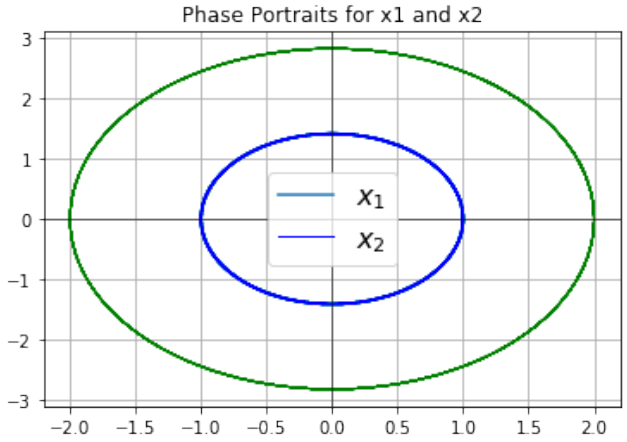
Movimiento sincronizado x1 y x2:

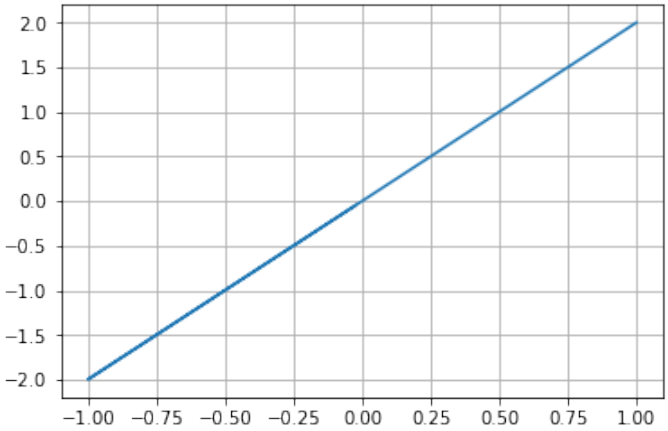
In [20]:
```
##########GRÁFICA 2.4#################
from numpy import loadtxt
from pylab import figure, plot, xlabel, grid, hold, legend, title, savefig
from matplotlib.font_manager import FontProperties
%matplotlib inline
t, x1, xy, x2, y2 = loadtxt('two_springs.dat', unpack=True)

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)
```

3 Gráficas Resultantes

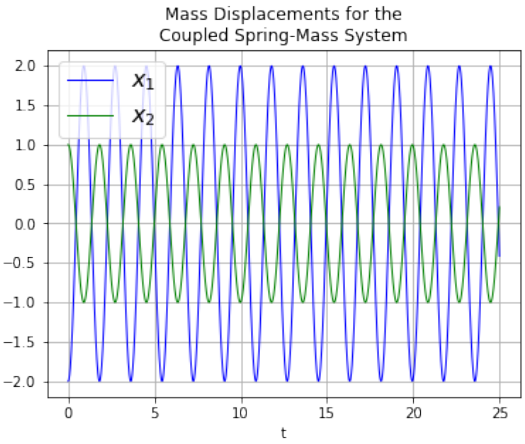
Ejemplo 2.1:

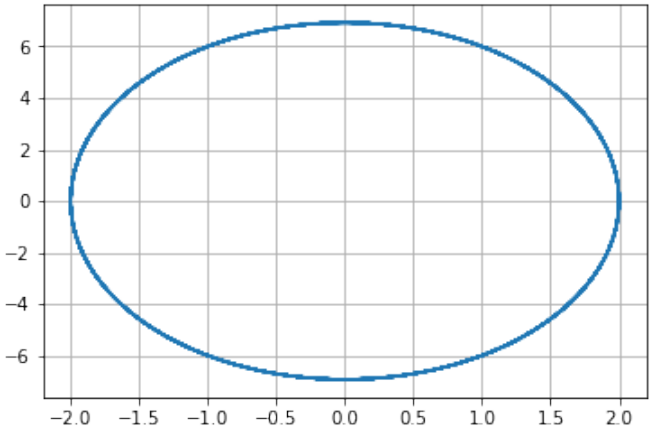

Mass Displacements for the
Coupled Spring-Mass System
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0
10
20
30
40
50
t
x_1
x_2

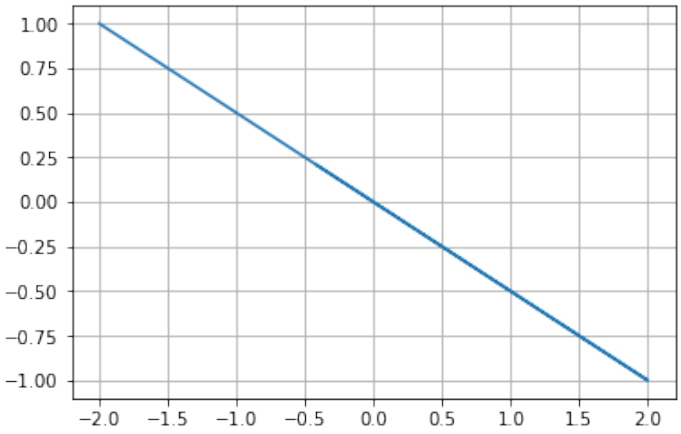

Phase Portraits for x1 and x2
3
2
1
0
−1
−2
−3
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
x_1
x_2


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

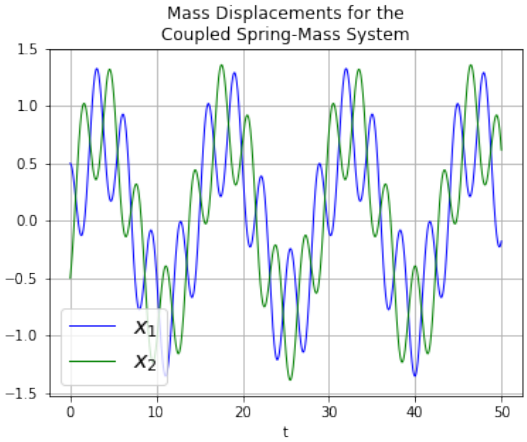
Ejemplo 2.2:

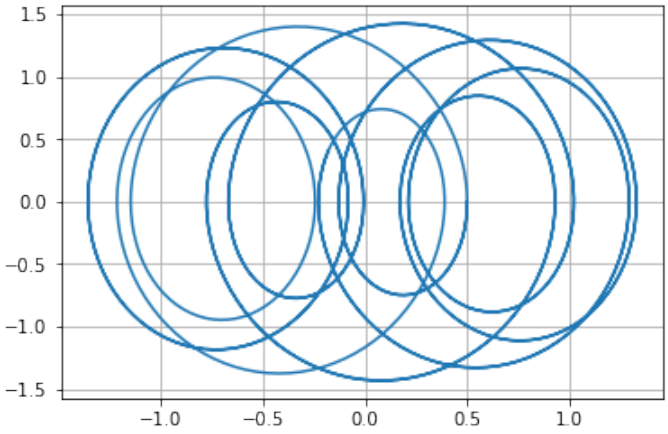

Mass Displacements for the
Coupled Spring-Mass System
x_1
x_2
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0
5
10
15
20
25
t

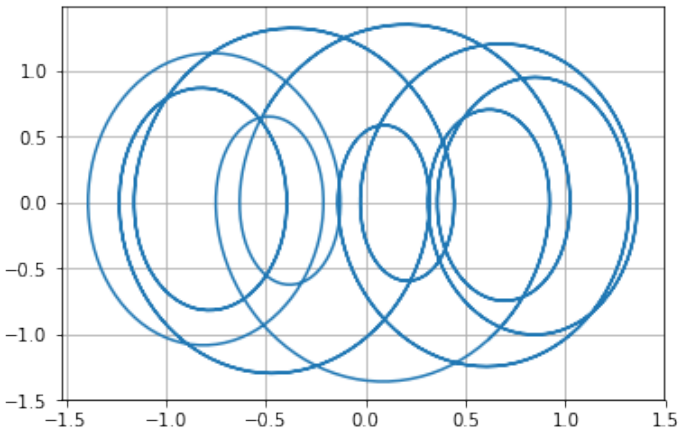

6
4
2
0
−2
−4
−6
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

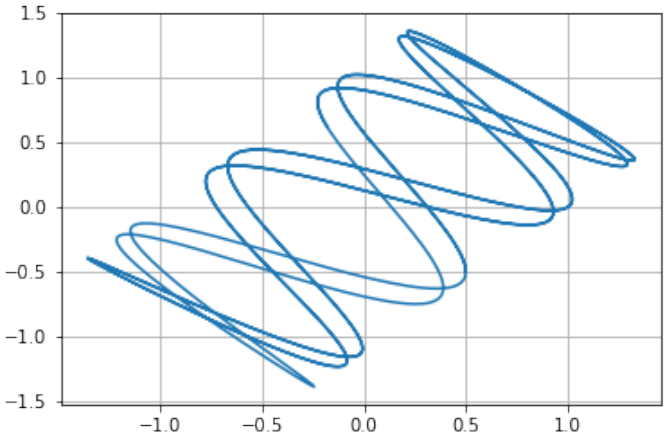

1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

Ejemplo 2.3:

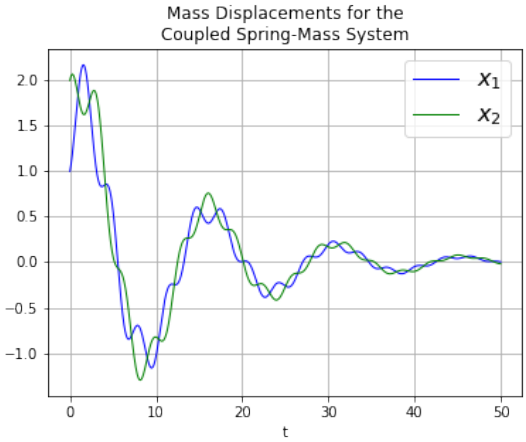

Mass Displacements for the
Coupled Spring-Mass System
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
10
20
30
40
50
t
x_1
x_2


1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.0
−0.5
0.0
0.5
1.0


1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.0
−0.5
0.0
0.5
1.0

Ejemplo 2.4:



4 Apéndice

1. ¿En general te pareció interesante esta actividad de modelación matemática? ¿Qué te gustó mas? ¿Qué no te gustó?
   Me parece muy interesante ya que es darle enfoque a la carrera. Me gustó analizar las diferencias entre el ejemplo y la actividad a realizar.

2. La cantidad de material te pareció ¿bien?, ¿suficiente?, ¿demasiado?
   Me parece una cantidad normal de material.

3. ¿Cuál es tu primera impresión de Jupyter Lab?
   No lo veo tan diferente al Jupyter Notebook.

4. Respecto al uso de funciones de SciPy, ¿ya habías visto integración numérica en tus cursos anteriores? ¿Cuál es tu experiencia?
   No lo había visto en FORTRAN, pero sí en otras materias.

5. El tema de sistema de masas acopladas con resortes, ¿ya lo habías resuelto en tu curso de Mecánica 2?
   Lo habíamos resuelto. Pero no con ecuaciones diferenciales.

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