Universidad de Sonora
Departamento de Ciencias Naturales y Exactas
Licenciatura en Física
Física computacional I

Actividad 6
Sistema de Resortes Acoplados

Alumno:
Gómez García
Manuel Ignacio

Profesor:
Lizarraga Celaya
Carlos

25 de mayo, 2018
Hermosillo, Sonora

"El saber de mis hijos
hará mi grandeza"

1 Introducción

En esta práctica nos enfocaremos en las Ecuaciones Diferenciales Ordinarias pues existen una gran variedad de técnicas a emplear para encontrar una solución. En particular, las ecuaciones no lineales debido a la gran cantidad y poder de algoritmos existentes además de la mínima potencia gráfica que requieren los sistemas de álgebra computacional, como lo son *Mathematica* y *Maple*.

El ejemplo que utilizaremos será el clásico ejercicio de dos masas y dos resortes para armar el sistema que se encuentra colgando del techo. El problema presenta de forma natural ya una Ecuación Diferencial de Segundo Orden. Al diferenciar y sustituir una ecuación en otra podemos obtener una Ecuación Diferencial de Cuarto Orden, haciendo esto que el problema se vuelva interesante puesto que usualmente en la física son de Segundo Orden todas las E.D. que suelen surgir.

La forma en que ahora se nos presenta el ejercicio nos permite explorar aún más fenómenos involucrados en nuestro sistema, siendo estos más interesante al agregar ciertos factores que involucran una oscilación forzada.

2 Modelo de resortes acoplados

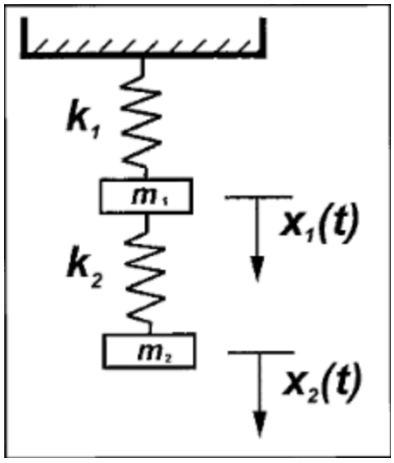

Figure 1: Sistema de resortes acoplados

Nuestro sistema a trabajar será como el mostrado en la figura 1. Los datos a recopilar serán primeramente las posiciones de cada una de las masas en las cuales alcanzan el equilibrio, siendo medidas desde su centro de masa hasta dicho punto donde el sistema se encuentra en reposo, de esta manera para $x_1(t)$ y $x_2(t)$.

ASUMIENDO LA LEY DE HOOKE

Si consideramos pequeñas oscilaciones, las fuerzas de restitución son de la forma $-k_1l_1$ y $-k_2l_2$ donde l_1 y l_2 son la elongación o comprensión de cada resorte. El resorte superior al estar conectado a ambos percibe dos fuerzas distintas; el primer resorte que jala el bloque hacía arriba con una fuerza dada por $-k_1x_1$ y el segundo, que aplica una fuerza que se resiste a ser estirado, la cual es $-k(x_2 - x_1)$. Si despreciamos el amortiguamiento, las ecuaciones de Newton para cada masa serían:

$$\begin{aligned} m_1a_1 &= -k_1x_1 - k_2(x_1 - x_2) \\ m_2a_2 &= -k_2(x_2 - x_1) \end{aligned} \tag{2.1}$$

De aquí, tenemos ahora dos E.D. de Segundo Orden entonces lo siguiente es encontrar una ecuación para x_1 sin involucrar a x_2, resolvemos la primer ecuación para x_2, que sería:

$$x_2 = \tfrac{m_1a_1}{k_2} + \tfrac{k_1+k_2}{k_2}x_1$$

Ahora sustituimos el valor de x_2 en la segunda ecuación de (2.1), donde tenemos m_2, de modo que la sustitución resulta en lo siguiente:

$$m_1m_2x_1^{(4)} + (m_2k_1 + k_2(m_1 + m_2))a_1 + k_1k_2x_1 = 0$$

Dando por resultado que el movimiento de m_1 esta dado por una E.D. de Cuarto Grado.

Bien, lo siguiente será despejar a x_1 para encontrar una E.D. en términos de x_2 y tenemos que:

$$x_1 = \tfrac{m_2}{k_2}a_2 + x_2$$

Tras sustituir en la primer ecuación de (2.1) el valor obtenido de x_1, tendremos:

$$m_1 m_2 x_2^{(4)} + (m_2 k_1 + k_2(m_1 + m_2))a_2 + k_1 k_2 x_2 = 0$$

Como podemos ver, resulta que ambas E.D. se rigen bajo el mismo comportamiento, siendo la única diferencia la posición de cada una de las masas. Como es costumbre, podríamos tomar los valores iniciales de ambas posiciones y velocidades como 0, de modo que para resolver el problema será necesario solucionar dos problemas de valor inicial de Cuarto Orden.

EJEMPLOS CON LA MISMA MASA

Si hacemos que ambos cuerpos posean la misma masa, $m_1 = m_2 = 1$. Y en el caso de que no existe amortiguamiento o una fuerza externa, la ecuación para el movimiento termina siendo:

$$m^4 + (k_1 + 2k_2)m^2 + k_1 k_2 = 0$$

Y esta ecuación anterior tiene como raíces:

$$\pm\sqrt{-\tfrac{1}{2}k_1 + k_2 \pm \tfrac{1}{2}\sqrt{k_1^2 + 4k_2^2}}$$

Ejemplo 2.1 Describiremos el movimiento para el sistema de resortes acoplados con los siguientes valores: $k_1 = 6$, $k_2 = 4$, $x_1 = 1$, $x_2 = 2$, $v_1 = 0$ & $v_2 = 0$.

Podemos ver que las raíces de la ecuación característica serían $\pm\sqrt{2}i$ y $\pm 2\sqrt{3}i$. Con ello obtendremos una ecuación general que sería:

$$x(t) = c_1 Cos\sqrt{2}t + c_2 Sen\sqrt{2}t + c_3 cos 2\sqrt{3}t + c_4 sen 2\sqrt{3}t$$

En conjunto con las ecuaciones anteriores y esta última llegaremos a que la solución para x_1 & x_2 son:

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

Ejemplo 2.2 En este ejemplo, las constantes k_1, k_2, v_1 & v_2 mantienen los valor anterior mientras que $x_1 = -2$ & $x_2 = 1$. Dando ahora como resultado:

$$x_1(t) = -2Cos2\sqrt{3}t$$
$$x_2(t) = Cos2\sqrt{3}t$$

Ejemplo 2.3 Las valores para las constantes del resorte serán $k_1 = 0.4$ & $k_2 = 1.808$, los de posición $x_1 = 0.5$ & $x_2 = -0.5$, mientras que los de velocidad $v_1 = 0$ & $v_2 = 0.7$.

AMORTIGUAMIENTO

El amortiguamiento usualmente encontrado es el de sustancias viscosas; en dichos casos el amortiguamiento es proporcional a la velocidad. El amortiguamiento de cada masa es independiente al de la otra, por lo cual ahora hemos de expresar sus ecuaciones de movimiento añadiendo el siguiente término $-\delta_n a_n$, de modo que si suponemos que los coeficientes δ_n son pequeños, las ecuaciones se reescriben de la siguiente forma:

$$m_1 a_1 = -\delta_1 v_1 - k_1 x_1 - k_2(x_1 - x_2)$$
$$m_2 a_2 = -\delta_2 v_2 - k_2(x_2 - x_1)$$

Para obtener las ecuaciones que describen el movimiento de x_1 & x_2 debemos despejar una de estas incógnitas de la ecuación y sustituir dicha expresión obtenida en la otra ecuación. De tal forma obtenemos para x_1 & x_2, respectivamente:

$$m_1 m_2 x_1^{(4)} + (m_1\delta_1 + m_2\delta_2)x_1^{(3)} + (m_2 k_1 + k_2(m_1 + m_2) + \delta_1\delta_2)a_1 + (k_1\delta_2 + k_2(\delta_1 + \delta_2))v_1 + k_1 k_2 x_1 = 0$$

$$m_1 m_2 x_2^{(4)} + (m_1\delta_1 + m_2\delta_2)x_2^{(3)} + (m_2 k_1 + k_2(m_1 + m_2) + \delta_1\delta_2)a_2 + (k_1\delta_2 + k_2(\delta_1 + \delta_2))v_2 + k_1 k_2 x_2 = 0$$

Como podemos ver, las ecuaciones que describen el movimiento de ambos cuerpos son ecuaciones diferenciales lineales.

Ejemplo 2.4 Asumiendo que $m_1 = m_2 = 1$, $k_1 = 0.4$, $k_2 = 1.808$, $\delta_1 = 0.1$, $\delta_2 = 0.2$, $x_1 = 1$, $x_2 = 2$ & $v_1 = v_2 = 0.5$ describa el movimiento de las masas correspondientes.

3 Creando el Sistema de Ecuaciones Diferenciales Ordinarias

A lo largo de esta sección mostraremos el código utilizado para llevar a cabo los ejemplos 2.1, 2.2, 2.3 & 2.4 que fueron planteados anteriormente.

Para comenzar, debemos definir un arreglo/vector que almacene los datos a utilizar, esto se debe hacer en los 4 códigos que veremos. Este es el bloque de código empleado.

```
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
```

Figure 2: Arreglo que almacena los datos calculados.

El siguiente bloque de código en los programas corresponde a los valores que definiremos (masa, coeficiente de restitución, posiciones y velocidades iniciales, entre otros más).

Para el **ejemplo 2.1**, el código usado se muestra en la **Figura 3**, para el **ejemplo 2.2** la **Figura 4**, para el **ejemplo 2.3** la **Figura 5** y por último, tenemos la **Figura 6** para el **ejemplo 2.4**.

Lo siguiente consiste en escribir el código para generar cada una de las gráficas correspondientes. Para las gráficas del primer ejemplo usamos el código de la Figura 7. En dicho código vemos que se cargan primeramente las librerías que usaremos (esto se hace sólo una vez), después leeremos los datos desde el archivo que los contiene, especificándole a qué variable corresponde cada una de las series de datos, seguido de ello se hacen ajustes del gráfico -Nombre de los márgenes, grosor de las líneas, forma de visualizar, límites

```python
# Use ODEINT to solve the differential equations defined by the vector field
from scipy.integrate import odeint
import numpy as np
import math

# 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 = 250

# 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('Ejemplo2_1.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], np.abs((w1[0]-np.cos(np.sqrt(2)*t1))/np.cos(np.sqrt(2)*t1)),
               np.abs((w1[2]-2*np.cos(np.sqrt(2)*t1))/(2*np.cos(np.sqrt(2)*t1))), sep=" ", end="\n", file=f)
```

Figure 3: Parámetros para el ejemplo 2.1

```
# Use ODEINT to solve the differential equations defined by the vector field
from scipy.integrate import odeint
import numpy as np
import math

# 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 = 1.0
y2 = 0.0

# ODE solver parameters
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 25.0
numpoints = 250

# 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('Ejemplo2_2.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], np.abs((w1[0]+2*np.cos(np.sqrt(12)*t1))/(-2*np.cos(np.sqrt(12)*t1))),
               np.abs((w1[2]-np.cos(np.sqrt(12)*t1))/np.cos(np.sqrt(12)*t1)), sep=" ", end="\n", file=f)
```

Figure 4: Parámetros del ejemplo 2.2

```
# Use ODEINT to solve the differential equations defined by the vector field
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 = 1/2
y1 = 0.0
x2 = -1/2
y2 = 7/10

# ODE solver parameters
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 100.0
numpoints = 1500

# 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('Ejemplo2_3.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], sep=" ", end="\n", file=f)
```

Figure 5: Parámetros del ejemplo 2.3

```
# Use ODEINT to solve the differential equations defined by the vector field
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 = 1/2
x2 = 2.0
y2 = 1/2

# ODE solver parameters
abserr = 1.0e-8
relerr = 1.0e-6
stoptime = 50.0
numpoints = 1500

# 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('Ejemplo2_4.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], sep=" ", end="\n", file=f)
```

Figure 6: Parámetros del ejemplo 2.4

de los ejes, datos a gráficar-, y por último, guardamos nuestro archivo con el nombre que deseemos.
NOTA: Para el ejemplo 2.3 y 2.4, debemos omitir $e1$ y $e2$ de la lectura de datos.

```
# Plot the solution that was generated

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

#Genera la gráfica en el archivo
%matplotlib inline

t, x1, xy, x2, y2, e1, e2 = loadtxt('Ejemplo2_1.dat', unpack=True)

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

xlabel('Tiempo')
grid(True)
#hold(True)
lw = 1

ylim(-2.5,2.5)

plot(t, x1, 'b', linewidth=lw)
plot(t, x2, '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('X1 y X2 moviéndose en sincronía')
savefig('Ejemplo2_1-A.png', dpi=100)
```

Figure 7: Código para la primer gráfica del ejemplo 2.1

Ahora en base al tipo de gráfico que estemos por realizar habrá que variar los datos mencionados.

Por ejemplo, para las gráficas correspondientes al **ejemplo 2.2** van de la Figura 13 a la 16, cambiamos los parámetros a los mencionados anteriormente y listo.

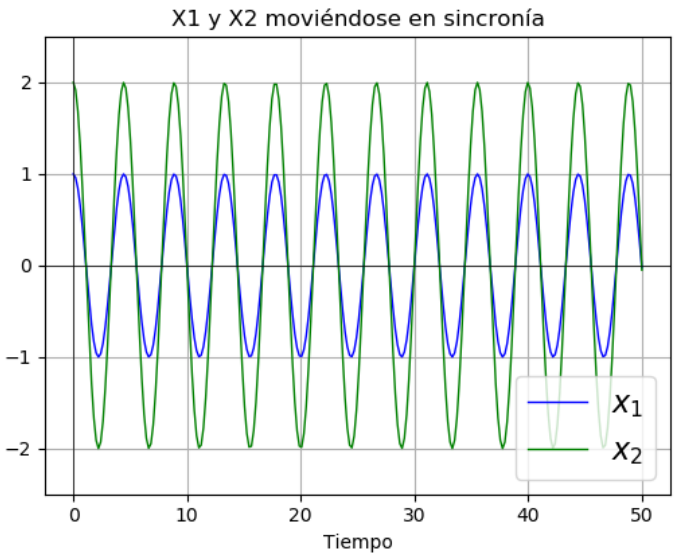

Figure 8: Primer gráfica del ejemplo 2.1

Para el **ejemplo 2.3**, de la Figura 17 a la 22.
Finalmente, para el **ejemplo 2.4** tendremos las Figuras 23 a 28.

4 Apéndice

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

   Me pareció sencilla pues ya habíamos trabajado anteriormente en graficar datos, lo nuevo fue estudiar cómo funciona el código.

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

   La cantidad de material está bien, no fue demasiado ni muy poco.

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

   Me gusta mucho más este aspecto de trabajo a simplemente Jupyter

Notebook, me siento un poco más organizado y con mayor libertad al hacer el trabajo de esta forma.

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

   Habíamos aplicado integración por métodos numéricos más sin embargo, fueron códigos que nosotros creamos y no eran parecidos a los que se aplicaron en esta actividad.

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

   No habíamos resuelto las ecuaciones ni tampoco recuerdo haberlas desarrollado tanto pero de igual forma teníamos la idea de cómo trabajarlas.

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

   Quizás eliminaría o por lo menos reduciría aquello del resumen de los primeros dos capítulos de la lectura.

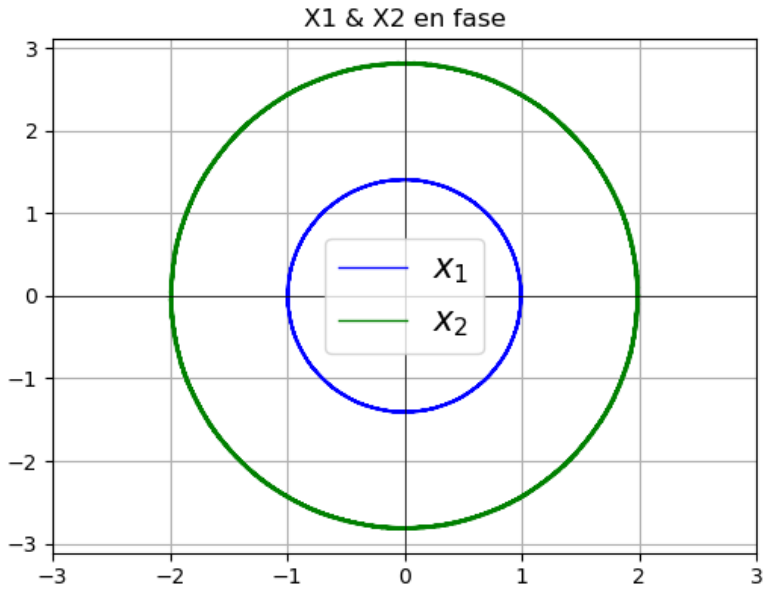

Figure 9: Gráfica del ejemplo 2.1

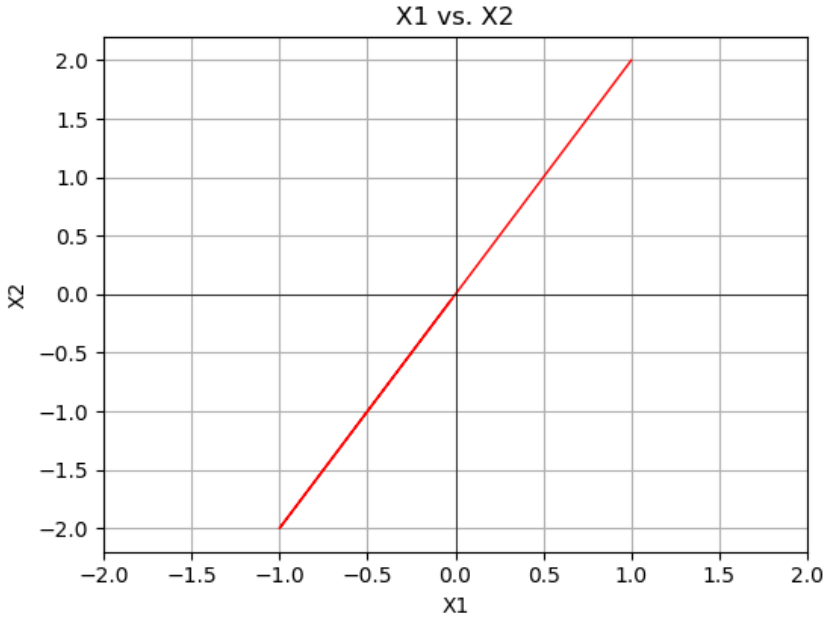

Figure 10: Gráfica del ejemplo 2.1

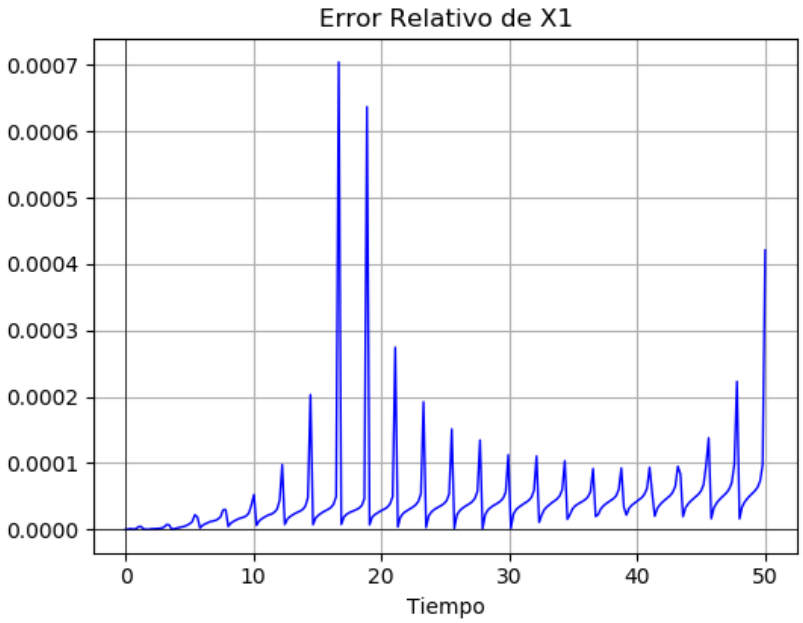

Figure 11: Gráfica del ejemplo 2.1

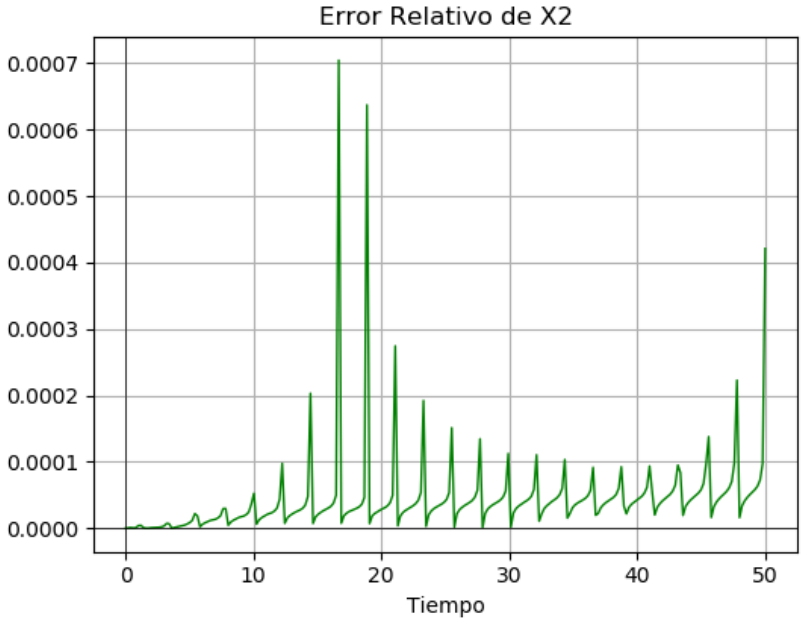

Figure 12: Gráfica del ejemplo 2.1

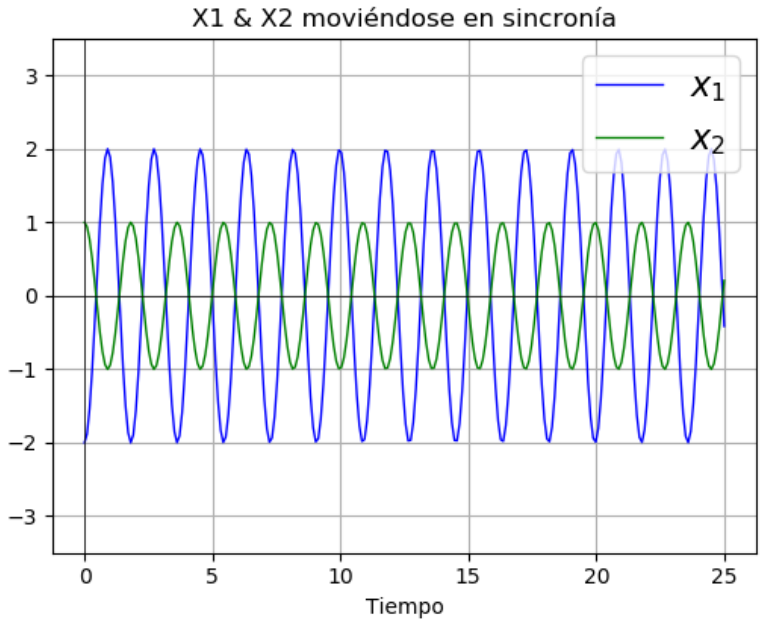

Figure 13: Gráfica del ejemplo 2.2

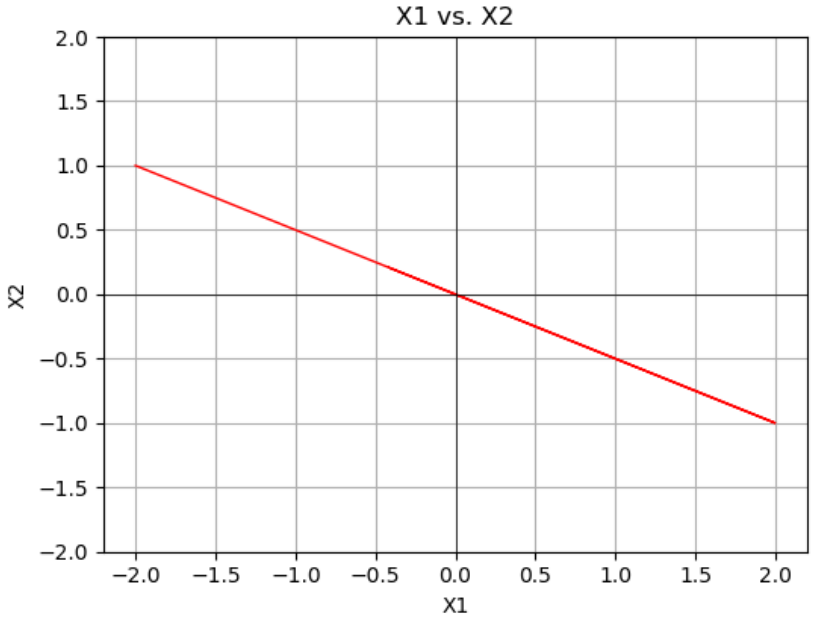

Figure 14: Gráfica del ejemplo 2.2

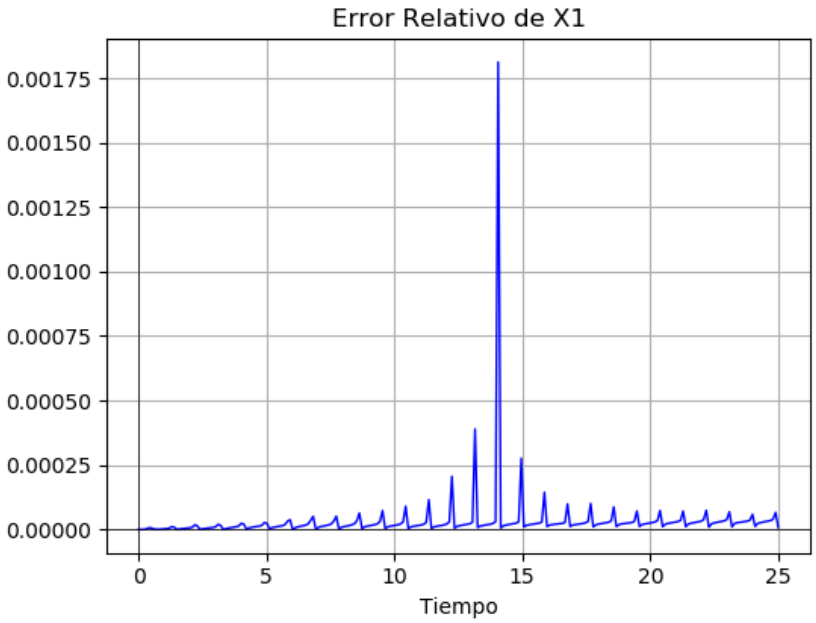

Figure 15: Gráfica del ejemplo 2.2

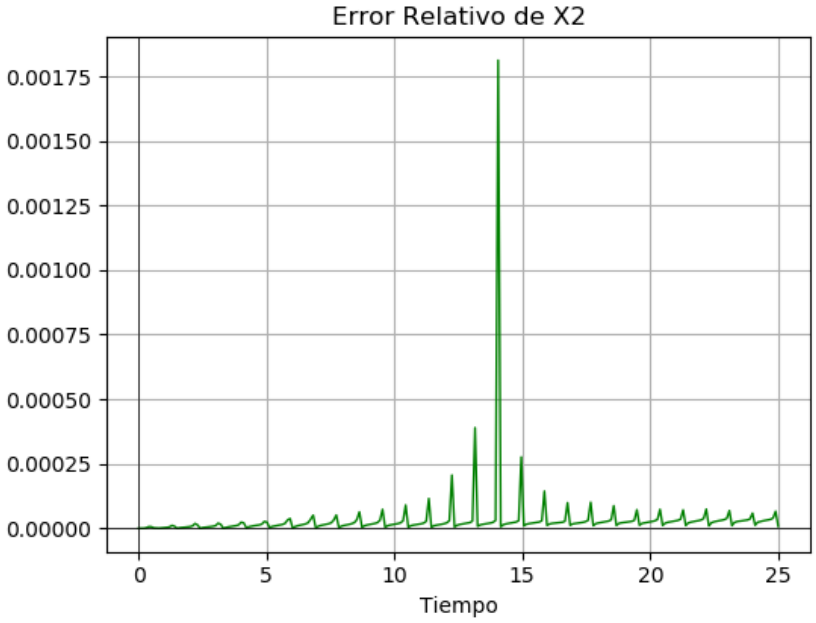

Figure 16: Gráfica del ejemplo 2.2

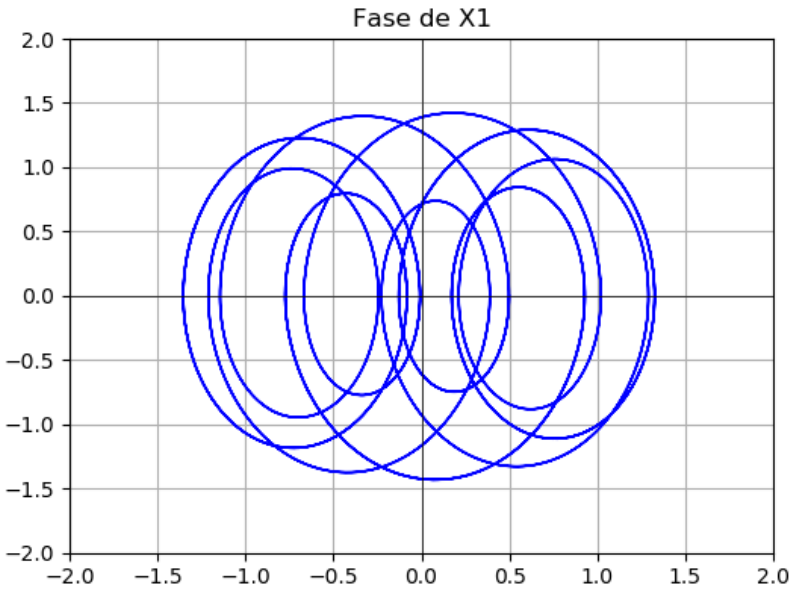

Figure 17: Gráfica del ejemplo 2.3

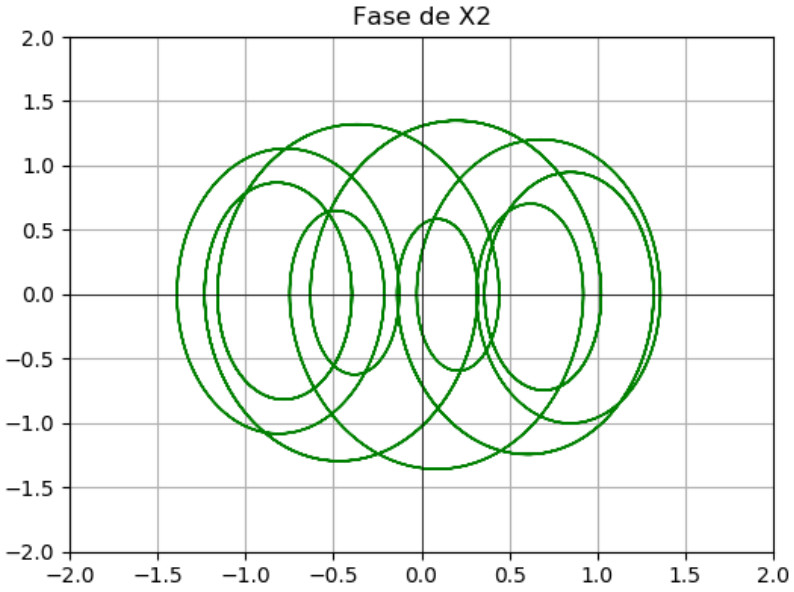

Figure 18: Gráfica del ejemplo 2.3

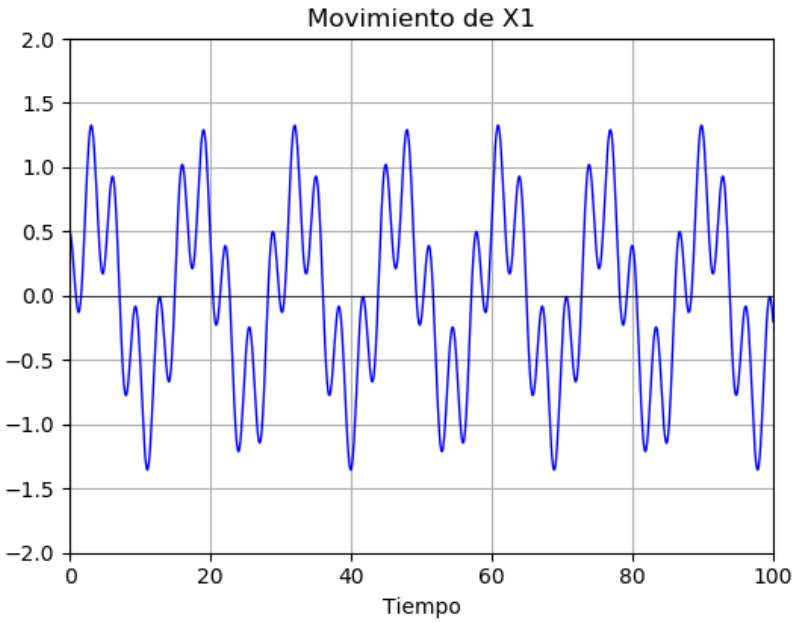

Figure 19: Gráfica del ejemplo 2.3

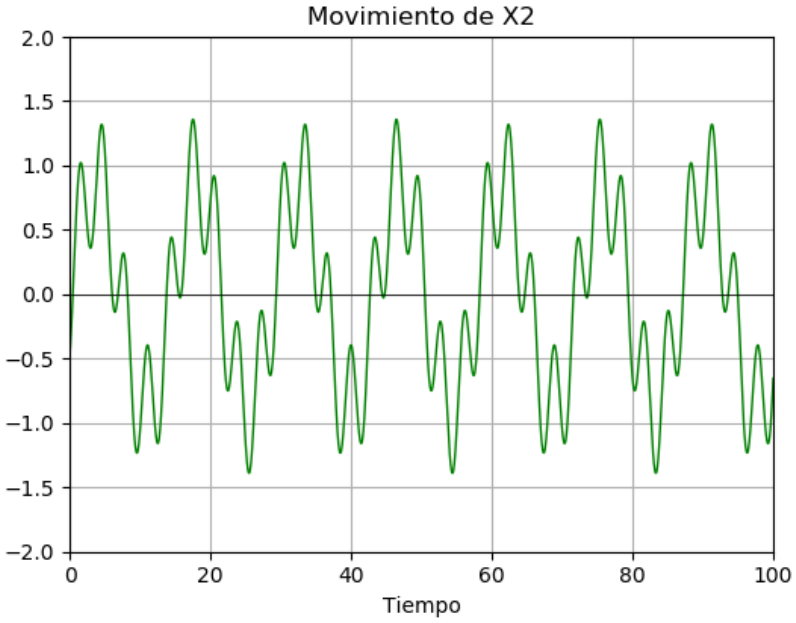

Figure 20: Gráfica del ejemplo 2.3

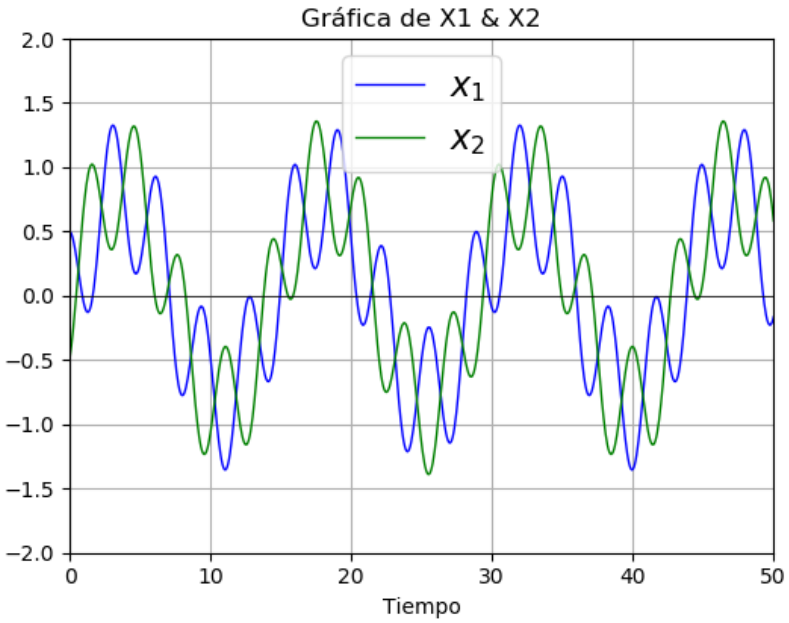

Figure 21: Gráfica del ejemplo 2.3

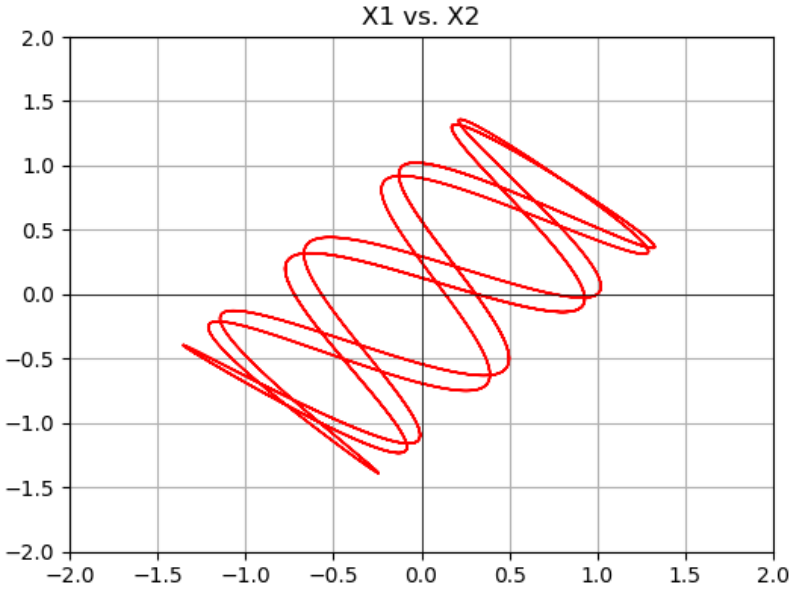

Figure 22: Gráfica del ejemplo 2.3

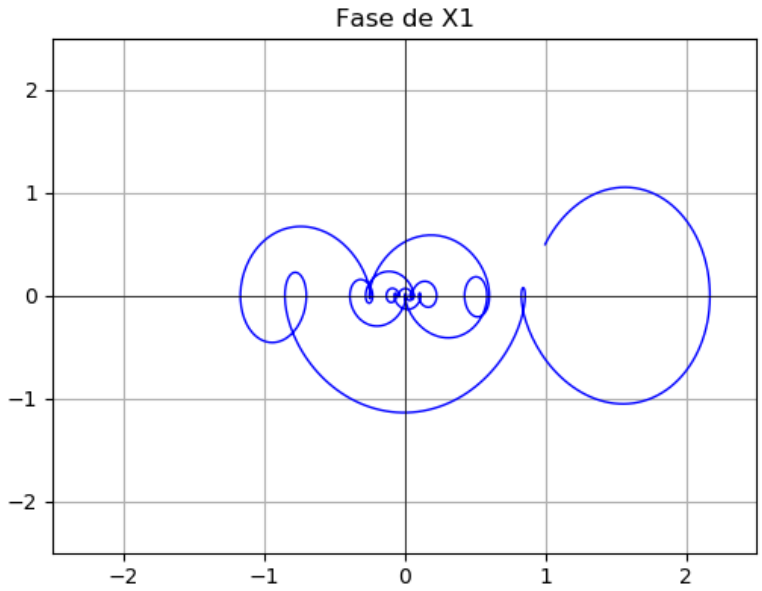

Figure 23: Gráfica del ejemplo 2.4

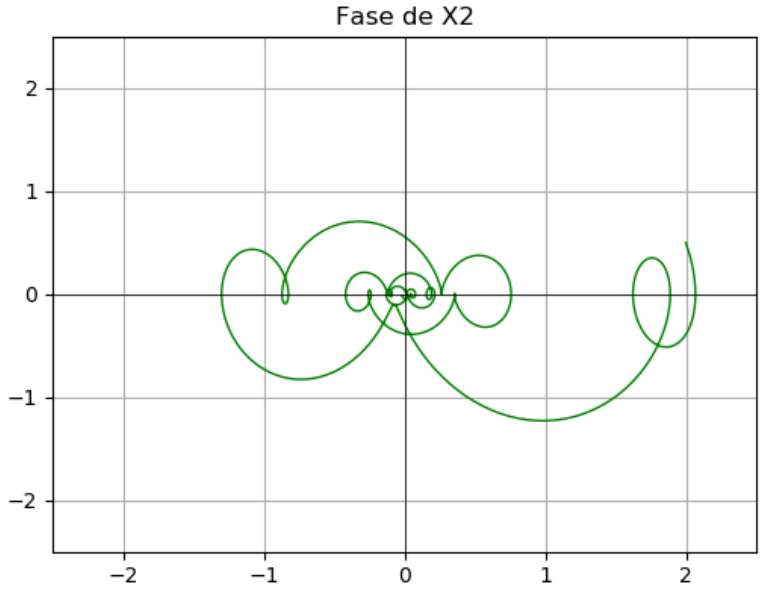

Figure 24: Gráfica del ejemplo 2.4

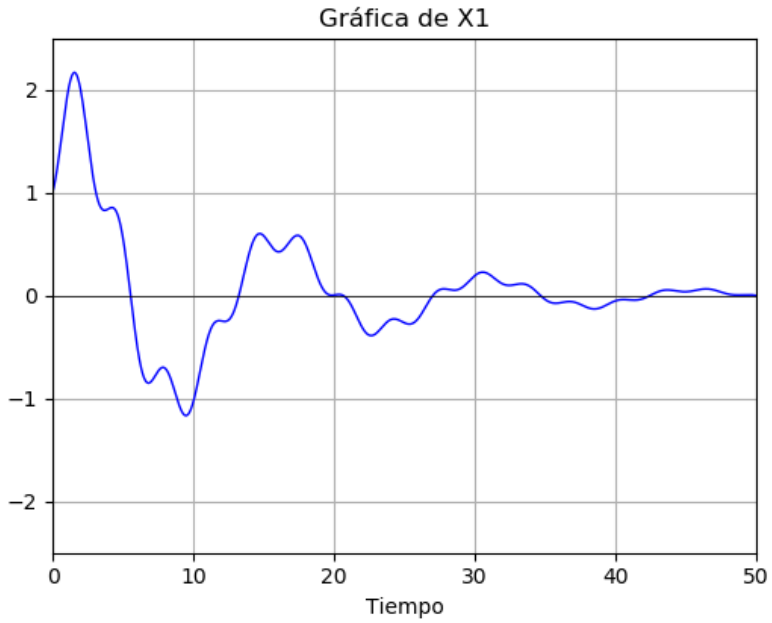

Figure 25: Gráfica del ejemplo 2.4

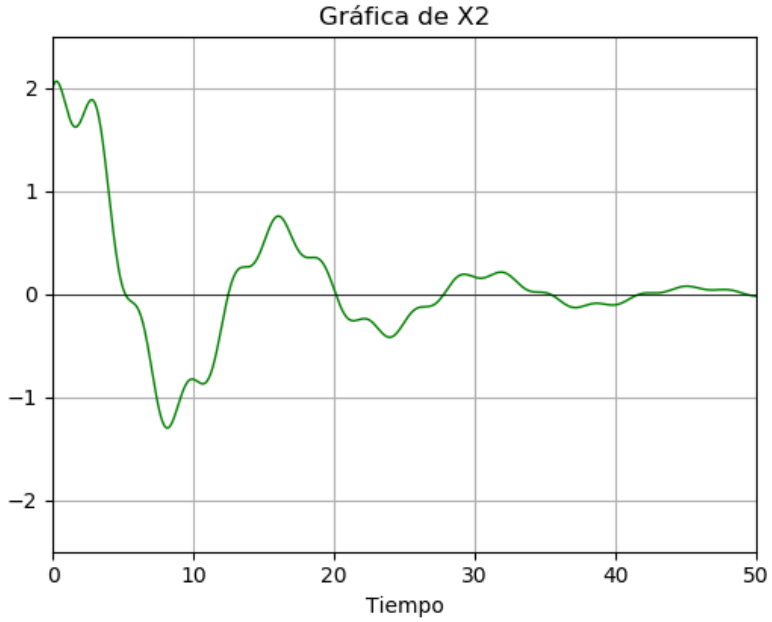

Figure 26: Gráfica del ejemplo 2.4

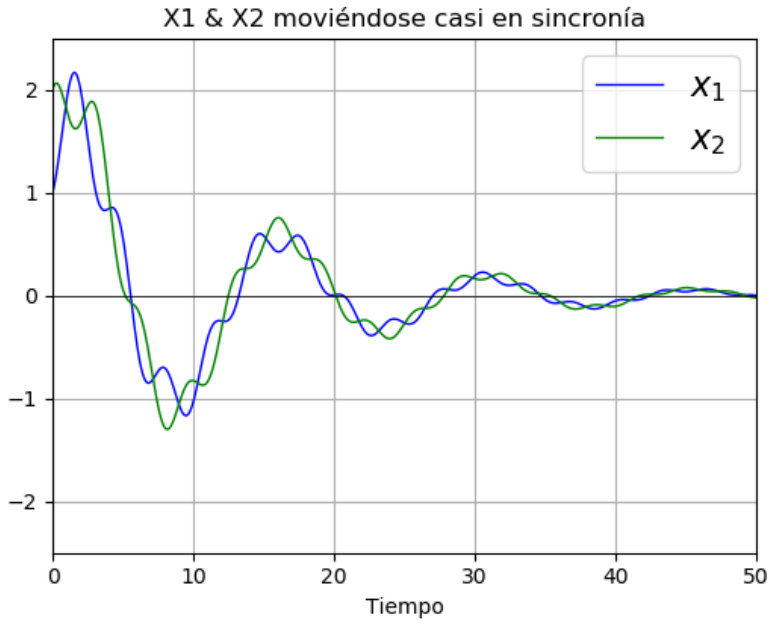

Figure 27: Gráfica del ejemplo 2.4

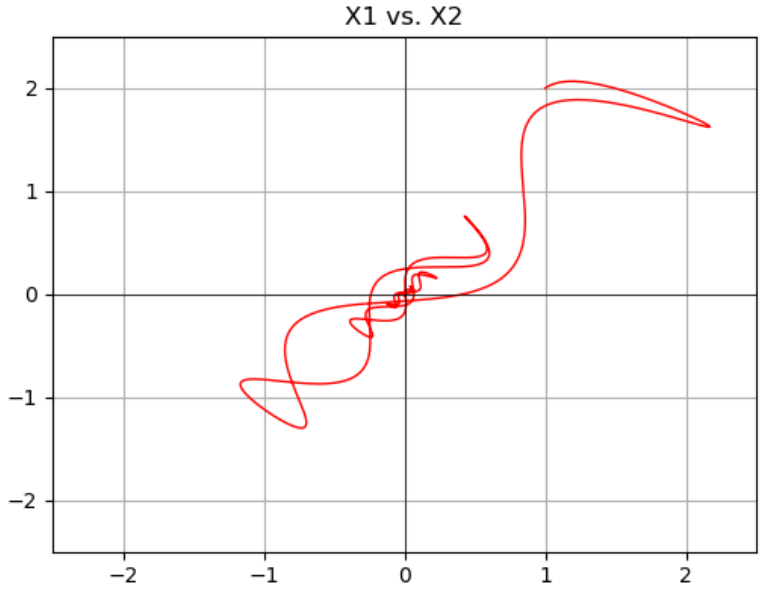

Figure 28: Gráfica del ejemplo 2.4