Actividad 7

Fernando Leyva Cárdenas

April 15, 2018

1 Introduccion

Si suponemos que las fuerzas de restauración no son lineales, lo que sin duda son para grandes vibraciones, podemos modificar el modelo en consecuencia. En lugar de asumir que la fuerza de restauración es de la forma -kx (ley de hook), supongamos que la fuerza de restauración tiene la forma (-kx + nx3) Entonces nuestro modelo se convierte en El rango de movimientos para el modelo no lineal es mucho más complicado que eso para el modelo lineal. Además, surgen preguntas de precisión al resolver estas ecuaciones. No se puede esperar que ningún solucionador numérico se mantenga preciso durante largos intervalos de tiempo. El error de truncamiento local acumulado, error de algoritmo, error de redondeo, propagación error de error, etc., eventualmente fuerza a que la solución numérica sea inexacta.

2 Desarrollo

Ahora mostraremos los siguientes ejercicios (seccion3), con ecuacion diferencial (aplicando phyton):

```
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,c1,c2]
    """
    x1, y1, x2, y2 = w
    m1, m2, k1, k2, L1, L2, b1, b2, c1, c2 = p

    # Create f = (x1',y1',x2',y2'):
    f = [y1,
       (-b1 * y1 - k1 * (x1 - L1) + k2 * (x2 - x1 - L2) + c1 * (x1)**3 + c2 * (x1 - x2)**3) / m1,
         y2,
         (-b2 * y2 - k2 * (x2 - x1 - L2) + c2 * (x2 - x1)**3) / m2]
    return f
```

Este primer ejercicio que se hara es el 3.1, las condiciones para este sistema esta dado con el siguiente codigo con sus especificaciones:

```
# Use ODEINT to solve the differential equations defined by the vector field
from scipy.integrate import odeint
import numpy as np
# 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
#norlinear coefficients
c1= -1/6
c2= -1/10
# Initial conditions
# x1 and x2 are the initial displacements; y1 and y2 are the initial velocities
x1 = 1.0
y1 = 0.0
x2 = -0.5
y2 = 0.0
```

Ahora utilizando el siguiente codigo graficaremos el comportamiento del fenomeno, es decir:

```
from numpy import loadtxt
from pylab import figure, plot, xlabel, grid, hold, legend, title, savefig
from matplotlib.font_manager import FontProperties
%matplotlib inline
import matplotlib.pyplot as plt
t, x1, y1, x2, y2 = loadtxt('ejemplo31.dat', unpack=True)


import numpy as np

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

xlabel('t')
grid(True)
#hold(True)
lw = 1
plt.xlim([-2,2])
plot(x1, y1, 'b', linewidth=lw)

legend((r'$x_1$', r'$x_2$'), prop=FontProperties(size=16))
title('Mass Displacements for the\nCoupled Spring-Mass System')
savefig('ejemplo31.png', dpi=100)

##############################
##############################
```

```
import numpy as np

figure(2, figsize=(6, 4.5))

xlabel('t')
grid(True)
#hold(True)
lw = 1
plt.xlim([-2,2])
plot(x2, y2, 'r', linewidth=lw)

legend((r'$x_1$', r'$x_2$'), prop=FontProperties(size=16))
title('Mass Displacements for the\nCoupled Spring-Mass System')
savefig('ejemplo31.png', dpi=100)
################################
################################

import numpy as np

figure(3, figsize=(6, 4.5))

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

plot(t, x1, 'y', linewidth=lw)

legend((r'$x_1$', r'$x_2$'), prop=FontProperties(size=16))
title('Mass Displacements for the\nCoupled Spring-Mass System')
savefig('ejemplo31.png', dpi=100)
####################################
####################################

import numpy as np

figure(4, figsize=(6, 4.5))

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

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

legend((r'$x_1$', r'$x_2$'), prop=FontProperties(size=16))
title('Mass Displacements for the\nCoupled Spring-Mass System')
savefig('ejemplo31.png', dpi=100)
################################3
################################

import numpy as np
```

```
figure(5, figsize=(6, 4.5))

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

plot(t, x2, 'y', linewidth=lw)
plot(t,x1, 'r', linewidth=lw)

legend((r'$x_1$', r'$x_2$'), prop=FontProperties(size=16))
title('Mass Displacements for the\nCoupled Spring-Mass System')
savefig('ejemplo31.png', dpi=100)
##########################################
##########################################

import numpy as np

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

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


plot(x1,x2, 'r', linewidth=lw)

legend((r'$x_1$', r'$x_2$'), prop=FontProperties(size=16))
title('Mass Displacements for the\nCoupled Spring-Mass System')
savefig('ejemplo31.png', dpi=100)
```

Para el ejemplo 3.2, tenemos las siguientes condiciones y datos:

```
# Use ODEINT to solve the differential equations defined by the vector field
from scipy.integrate import odeint
import numpy as np
# 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
#norlinear coefficients
c1= -1/6
```

```
c2= -1/10
# Initial conditions
# x1 and x2 are the initial displacements; y1 and y2 are the initial velocities
x1 = -0.5
y1 = 0.5
x2 = 3.001
y2 = 5.9
```

Ahora para el ejemplo 3.3, las condiciones son:

```
# Use ODEINT to solve the differential equations defined by the vector field
from scipy.integrate import odeint
import numpy as np
# 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
#norlinear coefficients
c1= -1/6
c2= -1/10
# Initial conditions
# x1 and x2 are the initial displacements; y1 and y2 are the initial velocities
x1 = -0.6
y1 = 0.5
x2 = 3.001
y2 = 5.9
```

Y por ultimo para el ejemplo 4.1, lo primero que debemos hacer es modificar nuestra ecuacion ya que es un problema difrente, lo haremos mediante el siguiente codigo:

```
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,c1,c2]
    """
    x1, y1, x2, y2 = w
    m1, m2, k1, k2, L1, L2, b1, b2, c1, c2 = p
```

```
    # Create f = (x1',y1',x2',y2'):
    f = [y1,
      (-b1 * y1 - k1 * (x1 - L1) + k2 * (x2 - x1 - L2) + c1 * (x1)**3 + c2 * (x1 - x2)**3 + f1 * np.cos(w1*t)) / m
         y2,
         (-b2 * y2 - k2 * (x2 - x1 - L2) + c2 * (x2 - x1)**3 + f2 *np.cos(w2*t)) / m2]
    return f
```

con sus condiciones:

```
# Use ODEINT to solve the differential equations defined by the vector field
from scipy.integrate import odeint
import numpy as np
# Parameter values
# Masses:
m1 = 1.0
m2 = 1.0
# Spring constants
k1 = 2/5
k2 = 1.0
# Natural lengths
L1 = 0.0
L2 = 0.0
# Friction coefficients
b1 = 1/10
b2 = 1/5
#norlinear coefficients
c1= 1/6
c2= 1/10
#forcing amplitudes
f1=1/3
f2=1/5
#forcing frecuencies
w1=1
w2=3/5
# Initial conditions
# x1 and x2 are the initial displacements; y1 and y2 are the initial velocities
x1 = 0.7
y1 = 0.0
x2 = 0.1
y2 = 0.0
```

Podemos ver las graficas del ejemplo 3.1 en la pagina 8, los del ejemplo 3.2 en la pagina 9, los del 3.3 en la pagina 10 y del 4.1 en la pagina 11; y nos podemos dar una idea de su comportamiento.

3 apendice

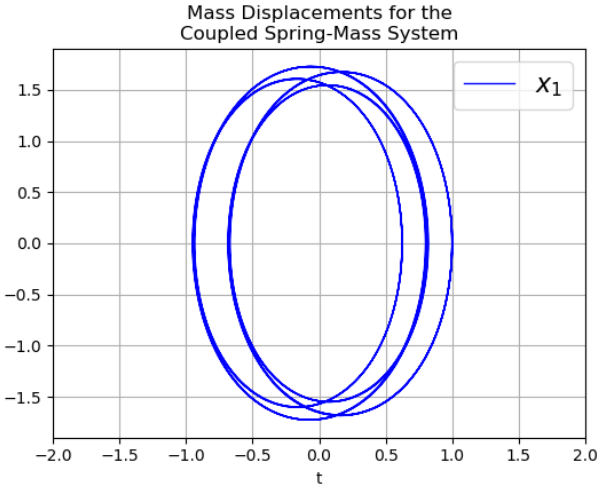
1. **¿Qué más te llama la atención de la actividad completa? ¿Que se te hizo menos interesante?** las graficas fueron inetresantes
2. **¿De un sistema de masas acopladas como se trabaja en esta actividad, hubieras pensado que abre toda una nueva área de fenómenos no lineales?** hay muchos fenomenos en el mundo que no son lineales porque se necesita de mas herramienta para su solucion

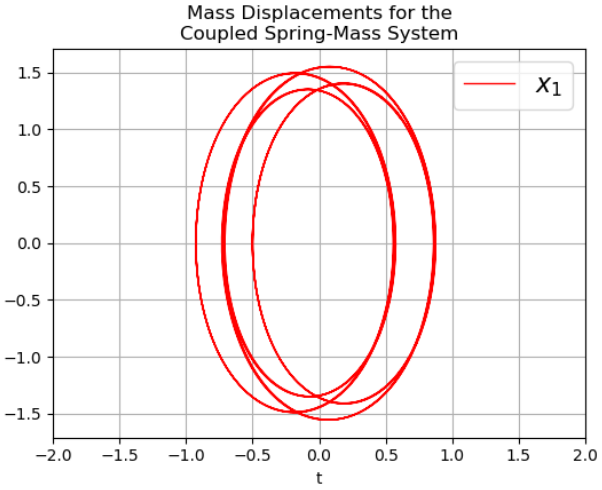
3. **¿Qué propondrías para mejorar esta actividad? ¿Te ha parecido interesante este reto?** fue interesante, lo unico frustrante fue hacer que se acomoden bien las cosas en latex, siempre me pasa :'(

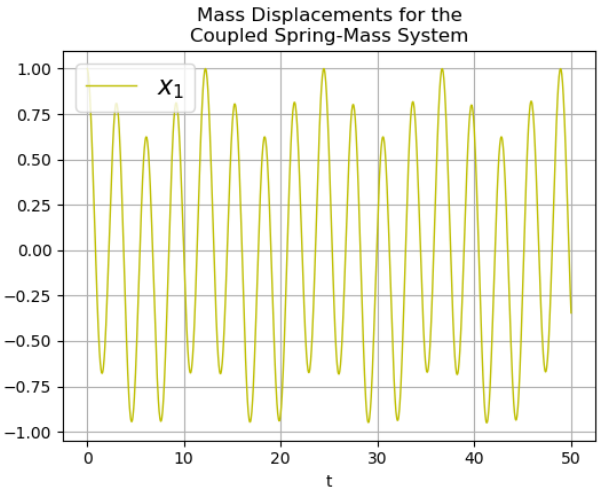
4. **¿Quisieras estudiar mas este tipo de fenómenos no lineales?** si

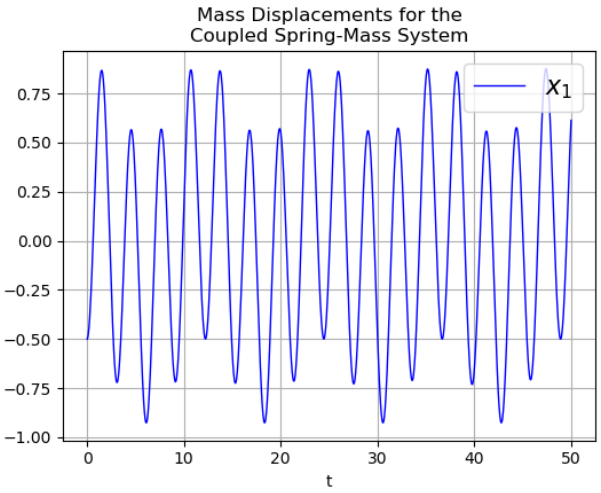
4 bibliografia

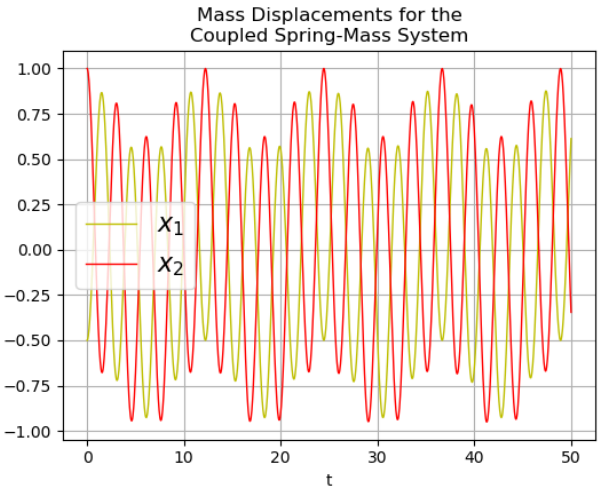
- Temple H. Fay, Sarah Duncan Graham (2003) Coupled Spring Equations. Int. J. Educ. Math. Sci. Tech.. Vol. 34, No. 1, pp. 65-79.

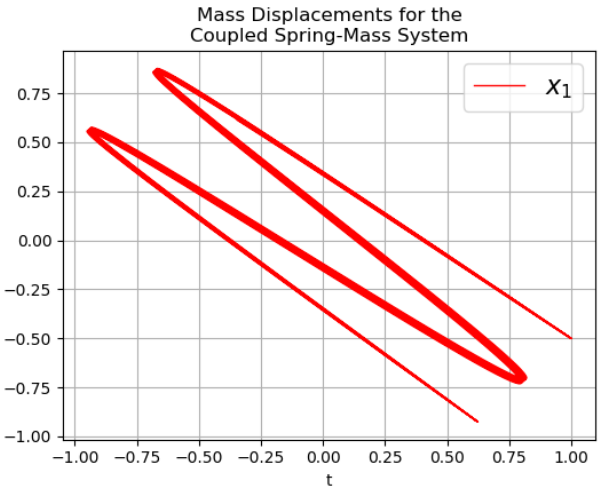

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

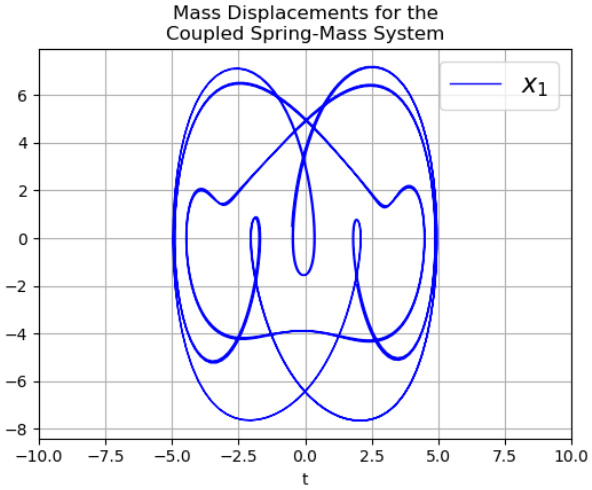

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

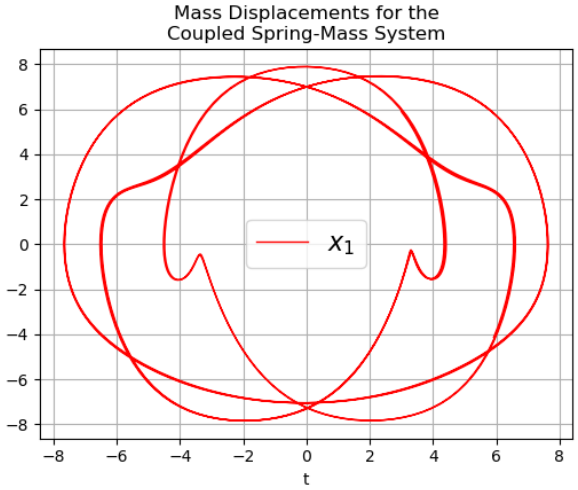

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

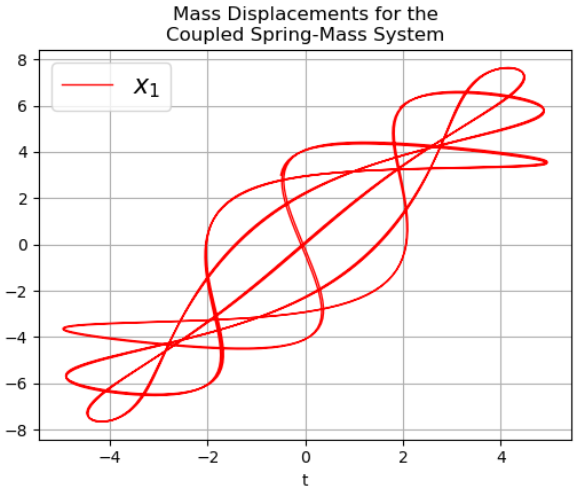

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

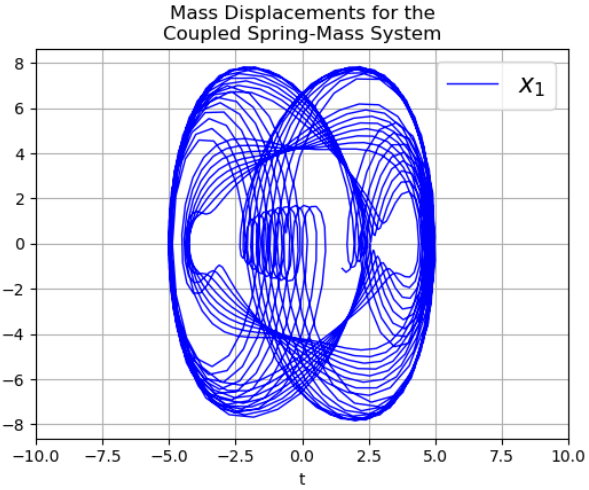

Mass Displacements for the
Coupled Spring-Mass System
x_1
x_2
t

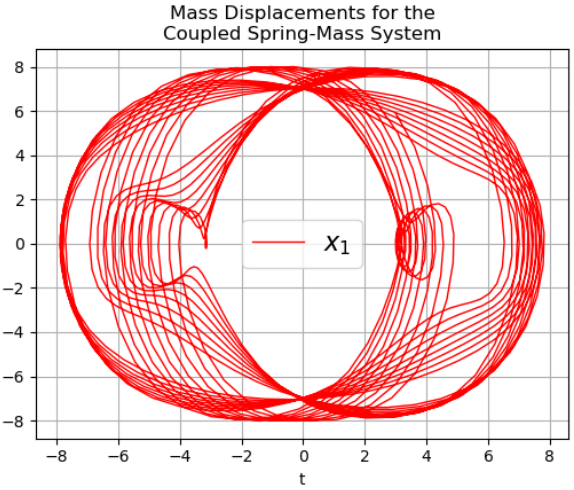

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

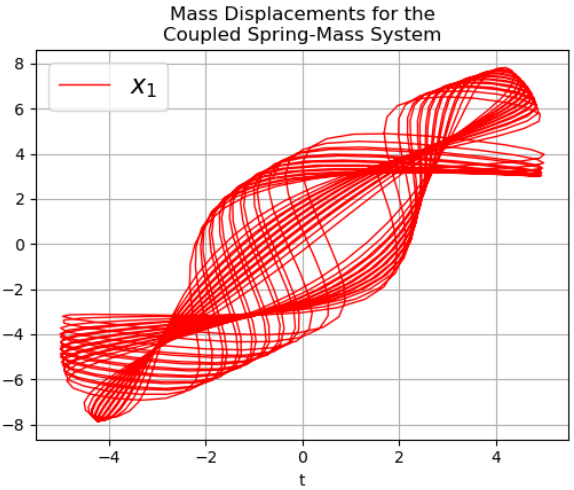

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

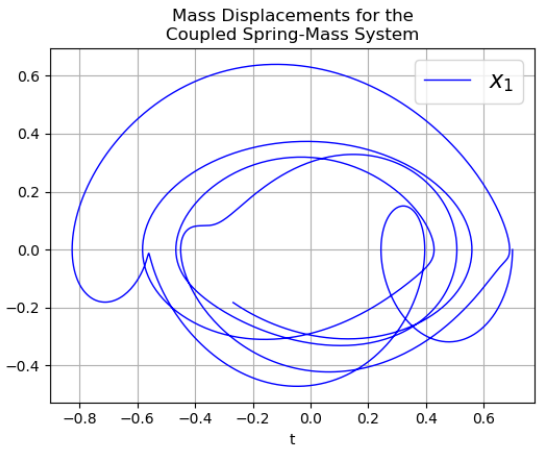

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

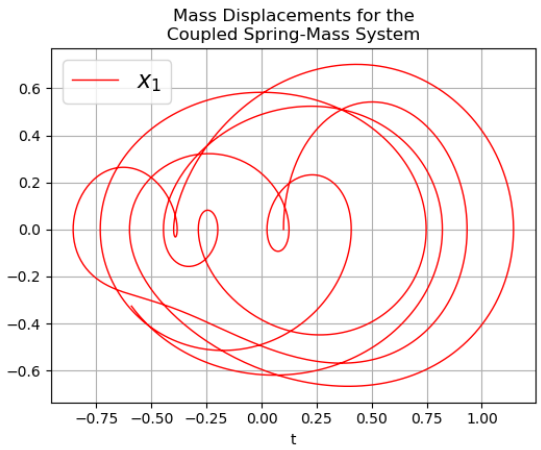

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

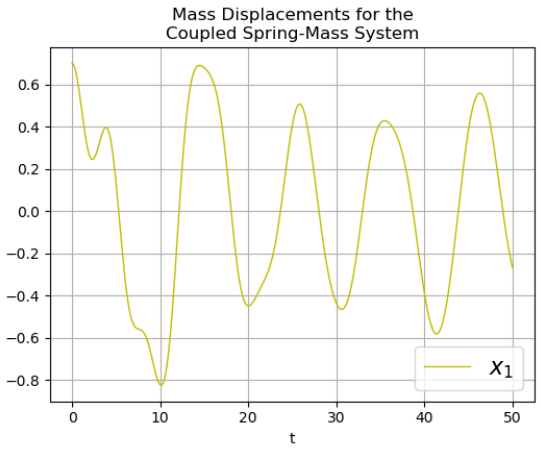

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

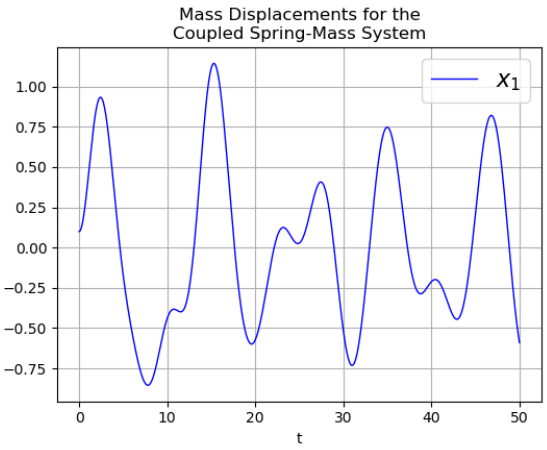

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

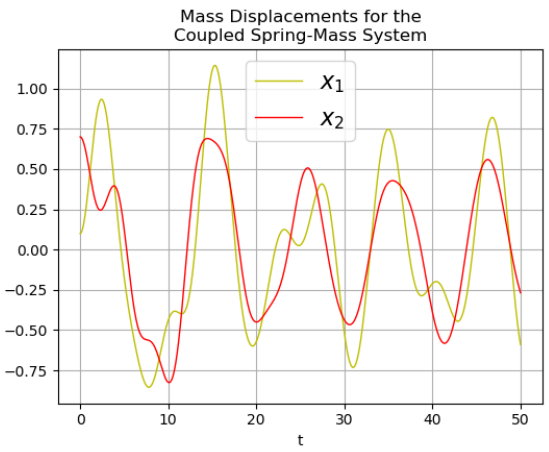

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

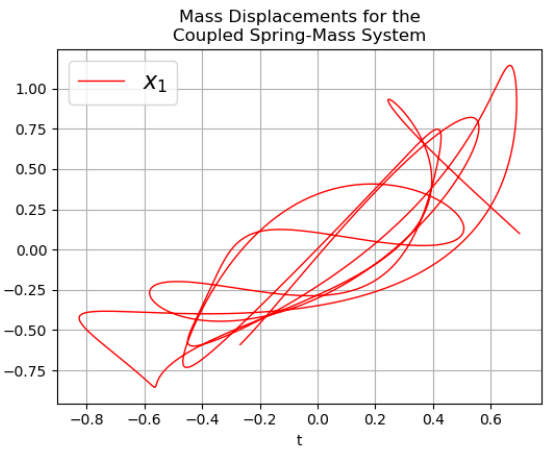

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

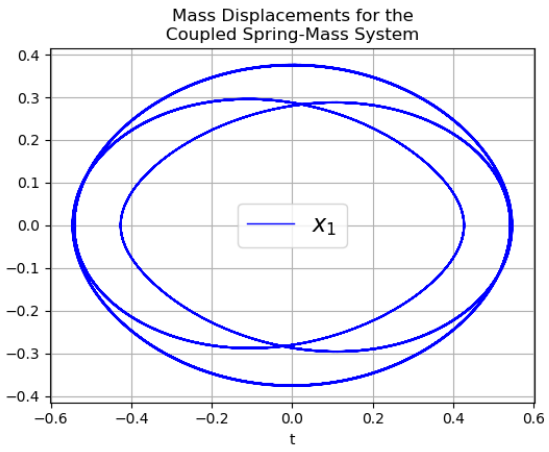

Mass Displacements for the
Coupled Spring-Mass System
x_1
t

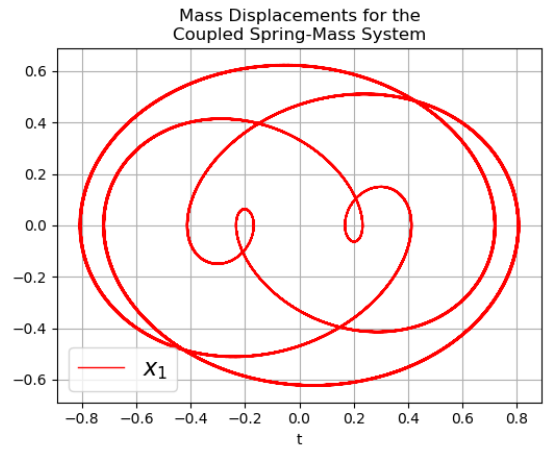

Mass Displacements for the
Coupled Spring-Mass System
x_1
t


Mass Displacements for the
Coupled Spring-Mass System
x_1
t


Mass Displacements for the
Coupled Spring-Mass System
x_1
x_2
t


Mass Displacements for the
Coupled Spring-Mass System
x_1
t


Mass Displacements for the
Coupled Spring-Mass System
x_1
t


Mass Displacements for the
Coupled Spring-Mass System
x_1
t