IDC402 Term Paper: Rössler Attractor

Name: Amlan Nayak
Roll: MS18197
Email: ms18197@iisermohali.ac.in

Contents

---

1 Introduction

Rössler attractor is a system of three non-linear ordinary differential equations which exhibit chaotic dynamics. The system can be described by:

$$\frac{dx}{dt} = -y - z$$
$$\frac{dy}{dt} = x + ay$$
$$\frac{dz}{dt} = b + z(x - c)$$

The fixed points of the system can be found by setting the individual equations to zero and finding the solutions. The solutions are given by:

$$x = \frac{c \pm \sqrt{c^2 - 4ab}}{2}$$
$$y = -\left(\frac{c \pm \sqrt{c^2 - 4ab}}{2a}\right)$$
$$z = \frac{c \pm \sqrt{c^2 - 4ab}}{2a}$$

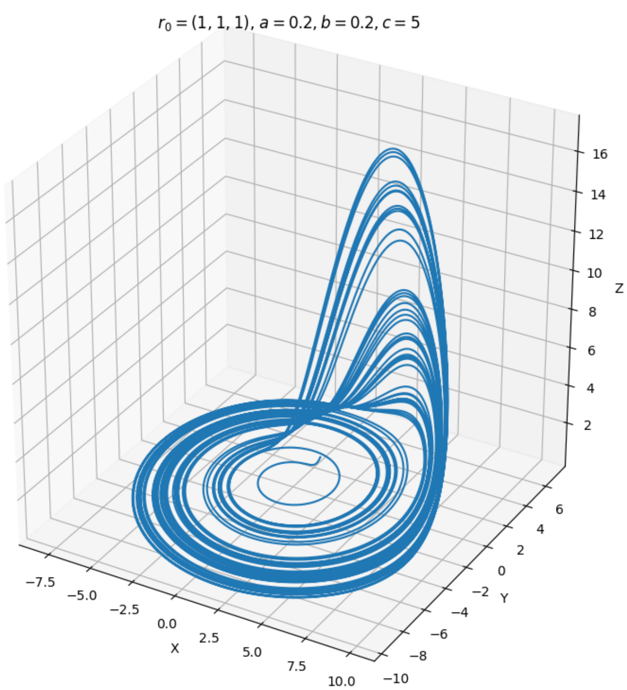

Figure 1: Rössler Attractor

2 Results

2.1 Chaotic Dynamics

For two very similar starting conditions, chaotic dynamics will lead to very different trajectories after evolution of time. Here we choose two initial conditions, r_0=(1,1,1) and r_0=(1.0001,1.0001,1.0001), to investigate the chaotic dynamics of Rösslerr attractor. The 3D trajectories and time series of individual directions has been plotted to show the divergence in behaviour.

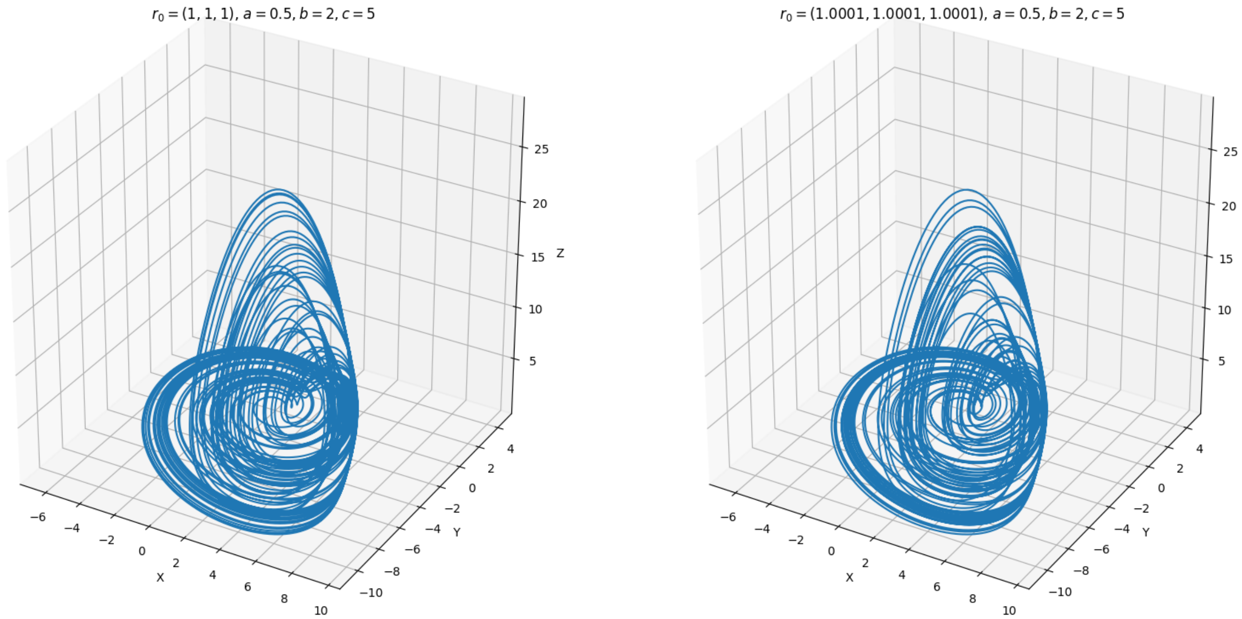

Figure 2: Varying 3D trajectories plots for r_0=(1,1,1) and r_0=(1.0001,1.0001,1.0001)

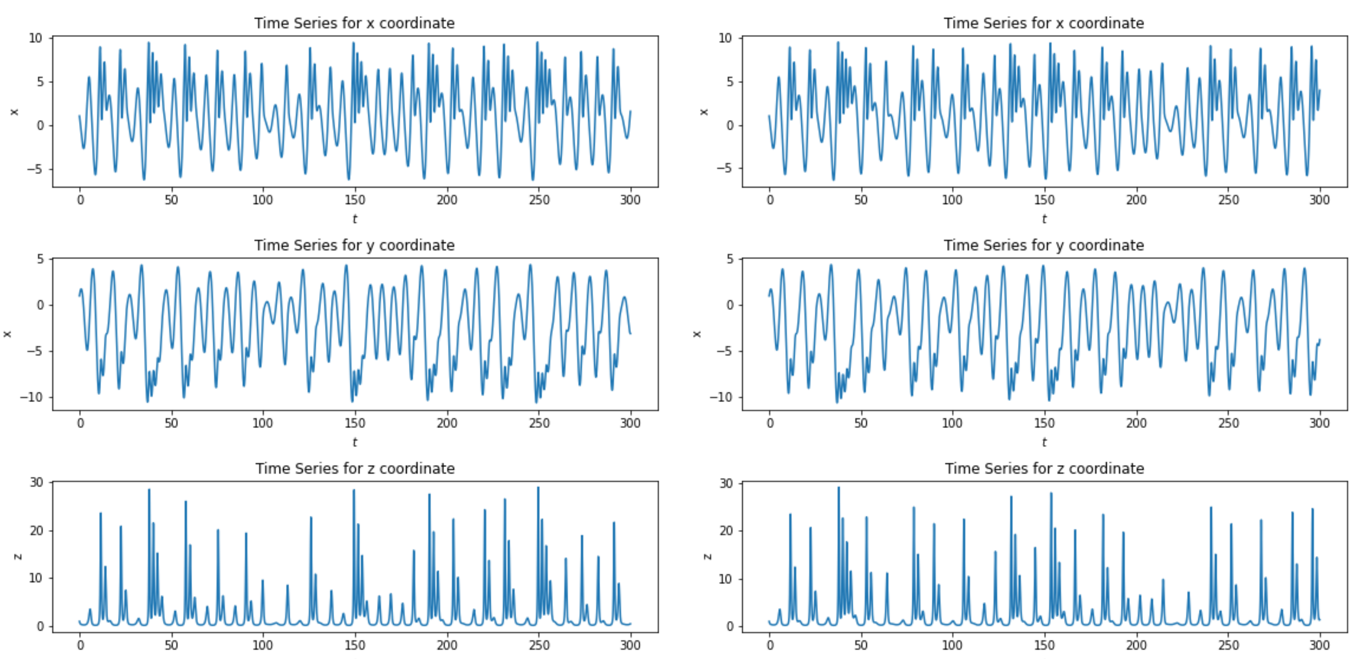

Figure 3: Varying time series plots for r_0=(1,1,1) and r_0=(1.0001,1.0001,1.0001)

2.2 Bifurcation Diagrams

Out of the 3 parameters present in the system, two have been kept constant and the third has been varied to obtain the bifurcation plots.

2.2.1 Varying c while keeping a and b constant

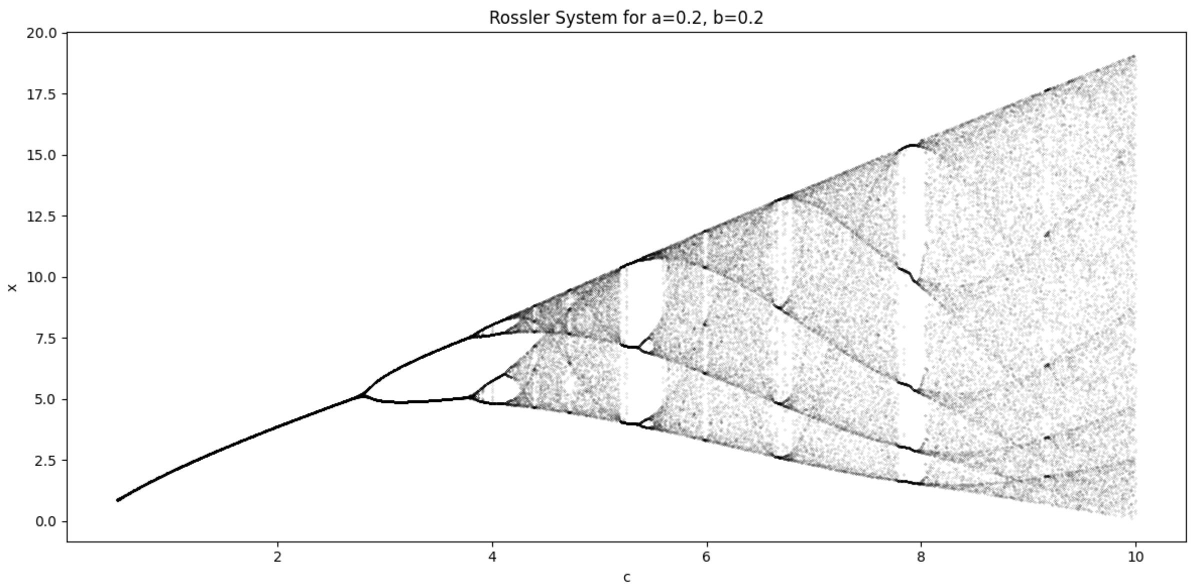

Figure 4: Bifurcation in x coordinate with changing c

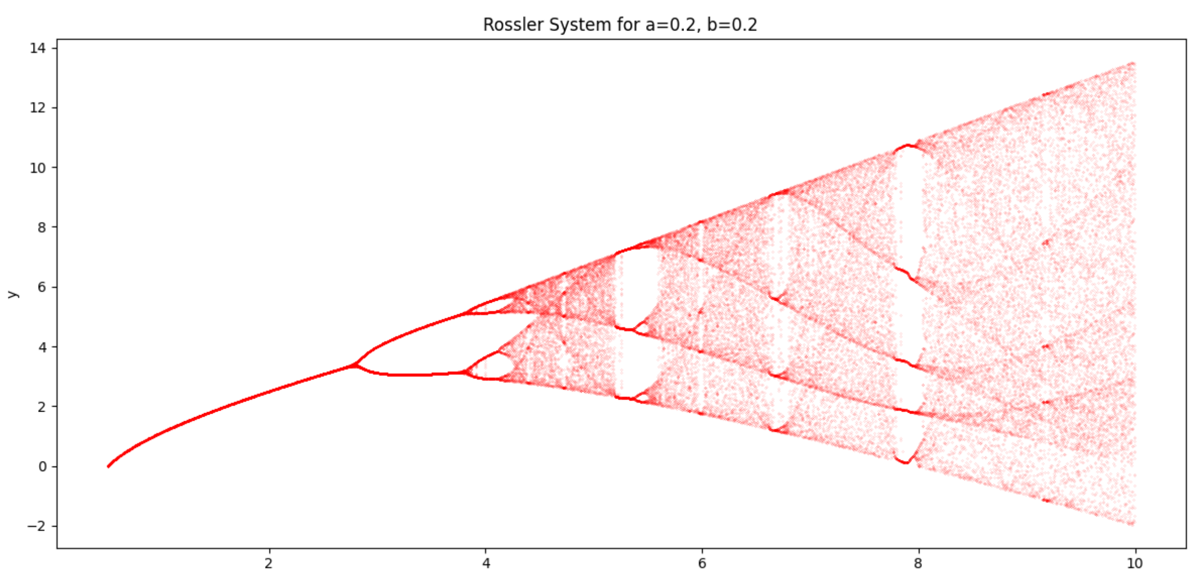

Figure 5: Bifurcation in y coordinate with changing c

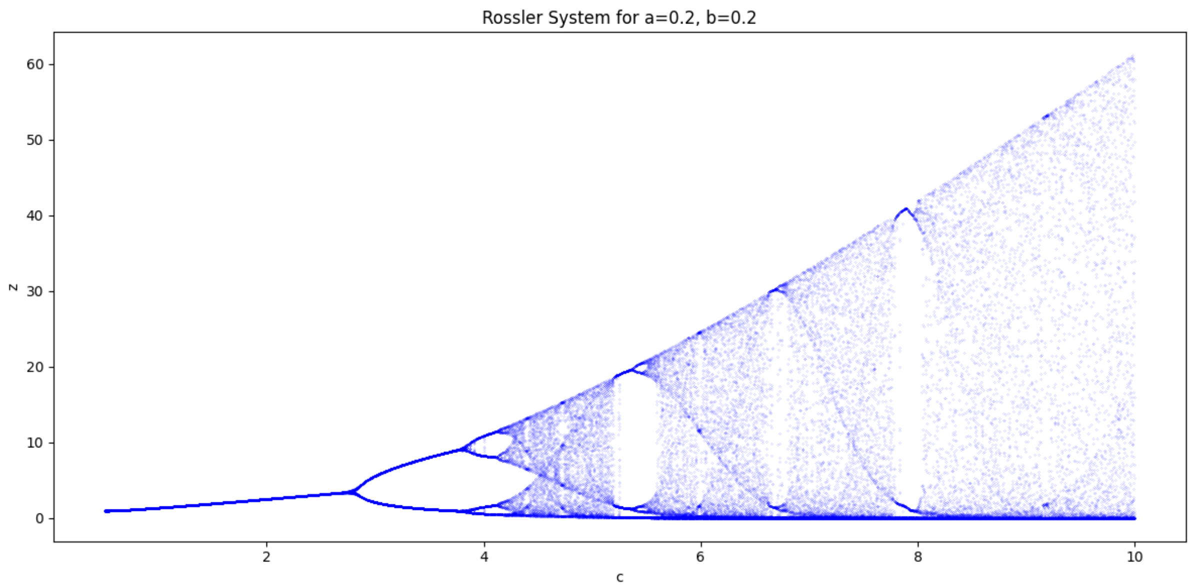

Figure 6: Bifurcation in z coordinate with changing c

2.2.2 Varying a while keeping b and c constant

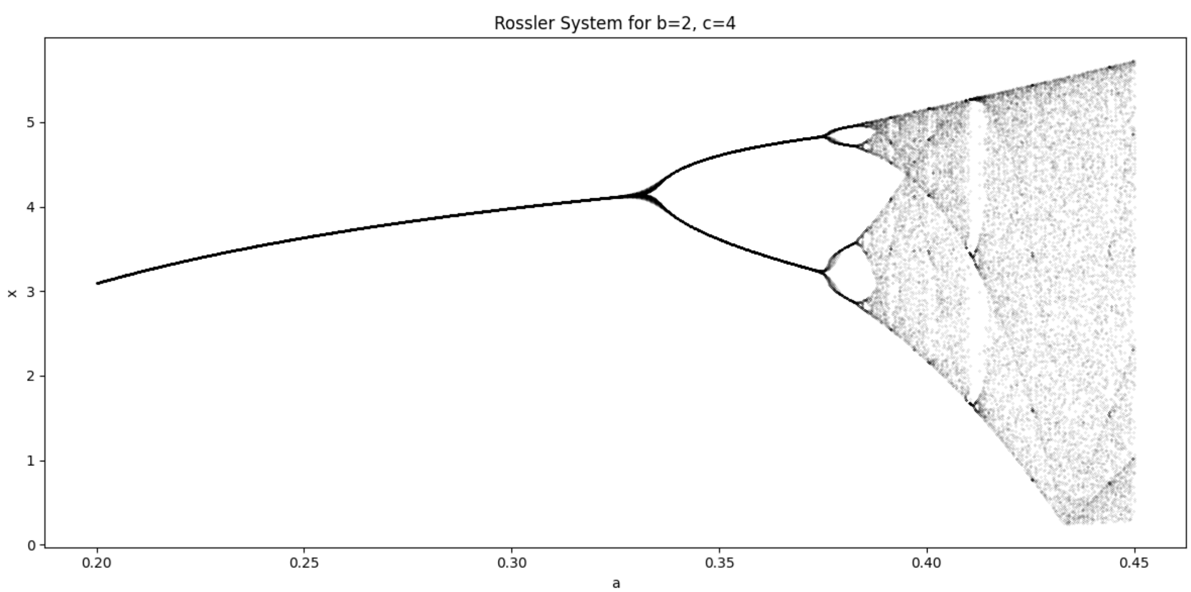

Figure 7: Bifurcation in x coordinate with changing a

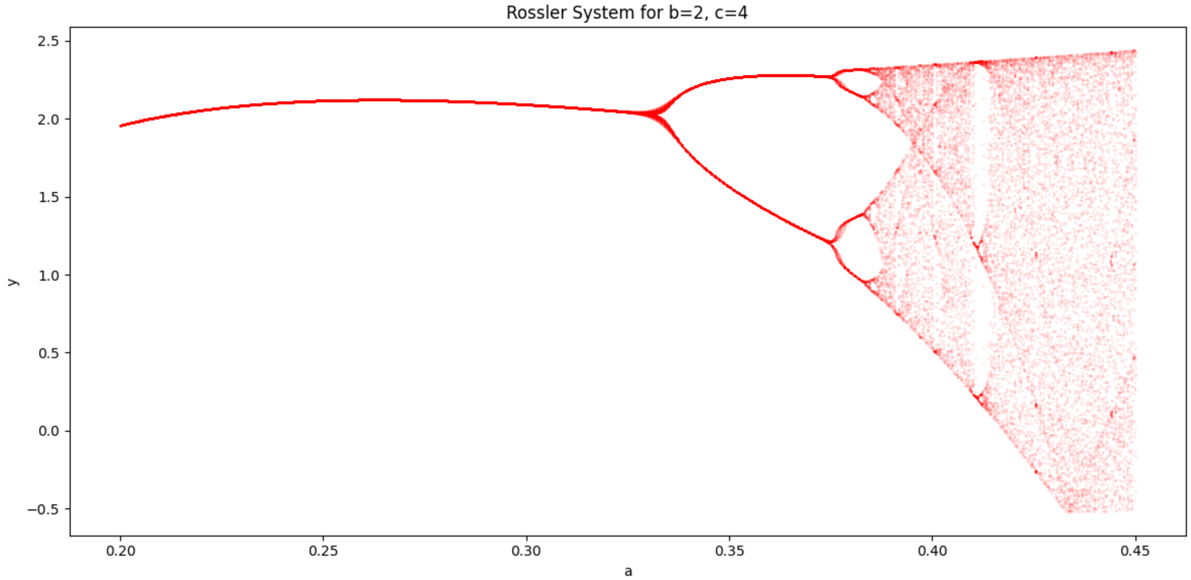

Figure 8: Bifurcation in y coordinate with changing a

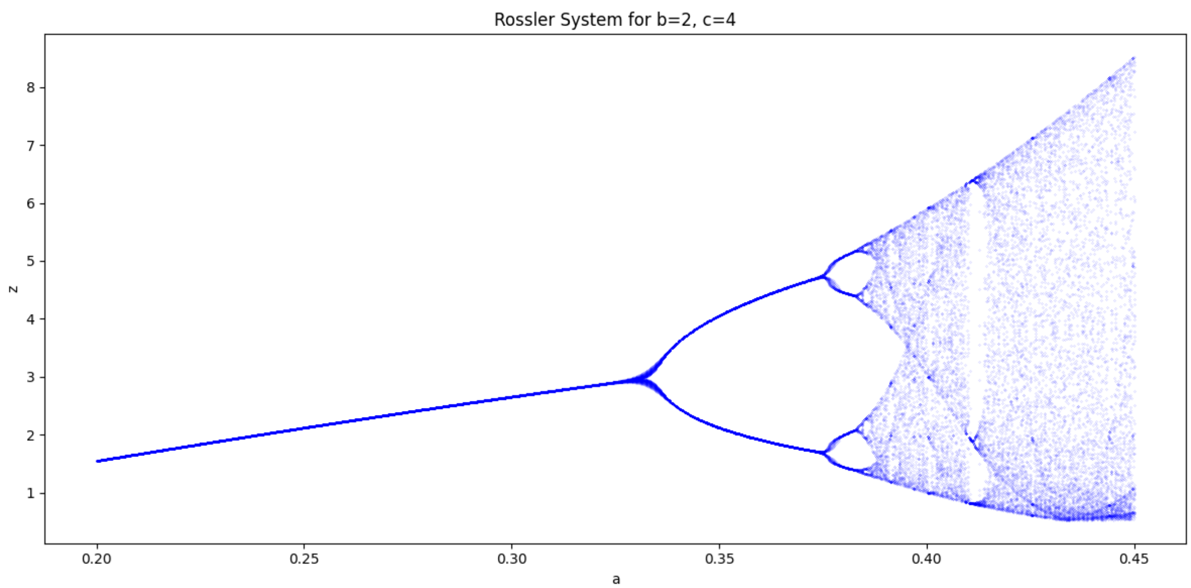

Figure 9: Bifurcation in z coordinate with changing a

2.2.3 Varying b while keeping a and c constant

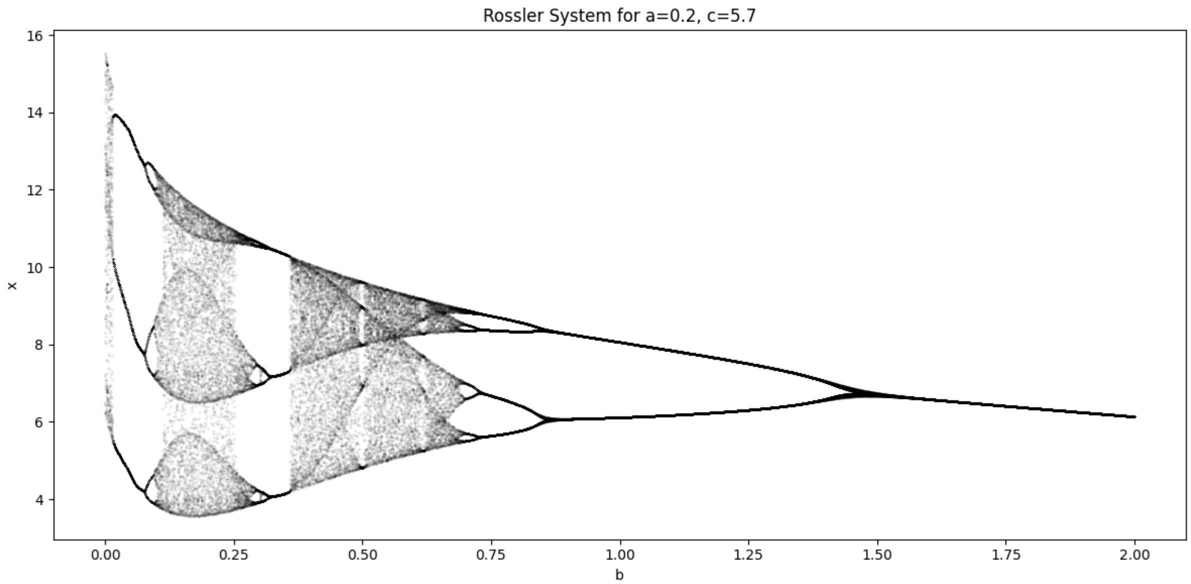

Figure 10: Bifurcation in x coordinate with changing b

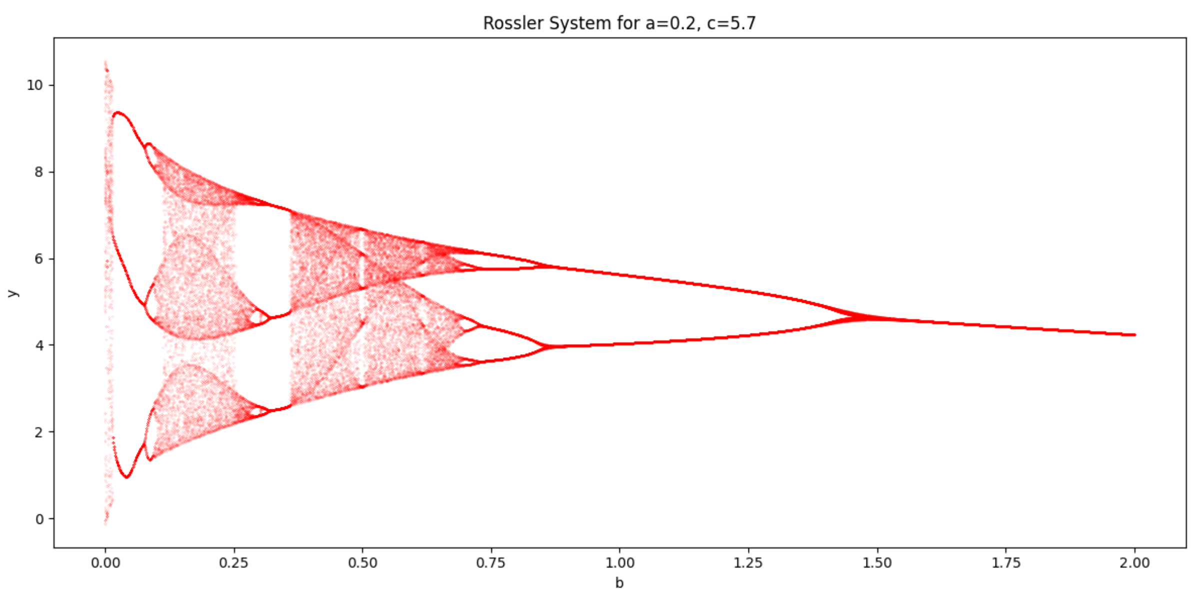

Figure 11: Bifurcation in y coordinate with changing b

2.3 Periodic Orbits

From the bifurcation plots, we can easily see initial periods of stability followed by descent into chaos. Using the values from bifurcation, the x-y trajectories of a starting point for various values of c are plotted.

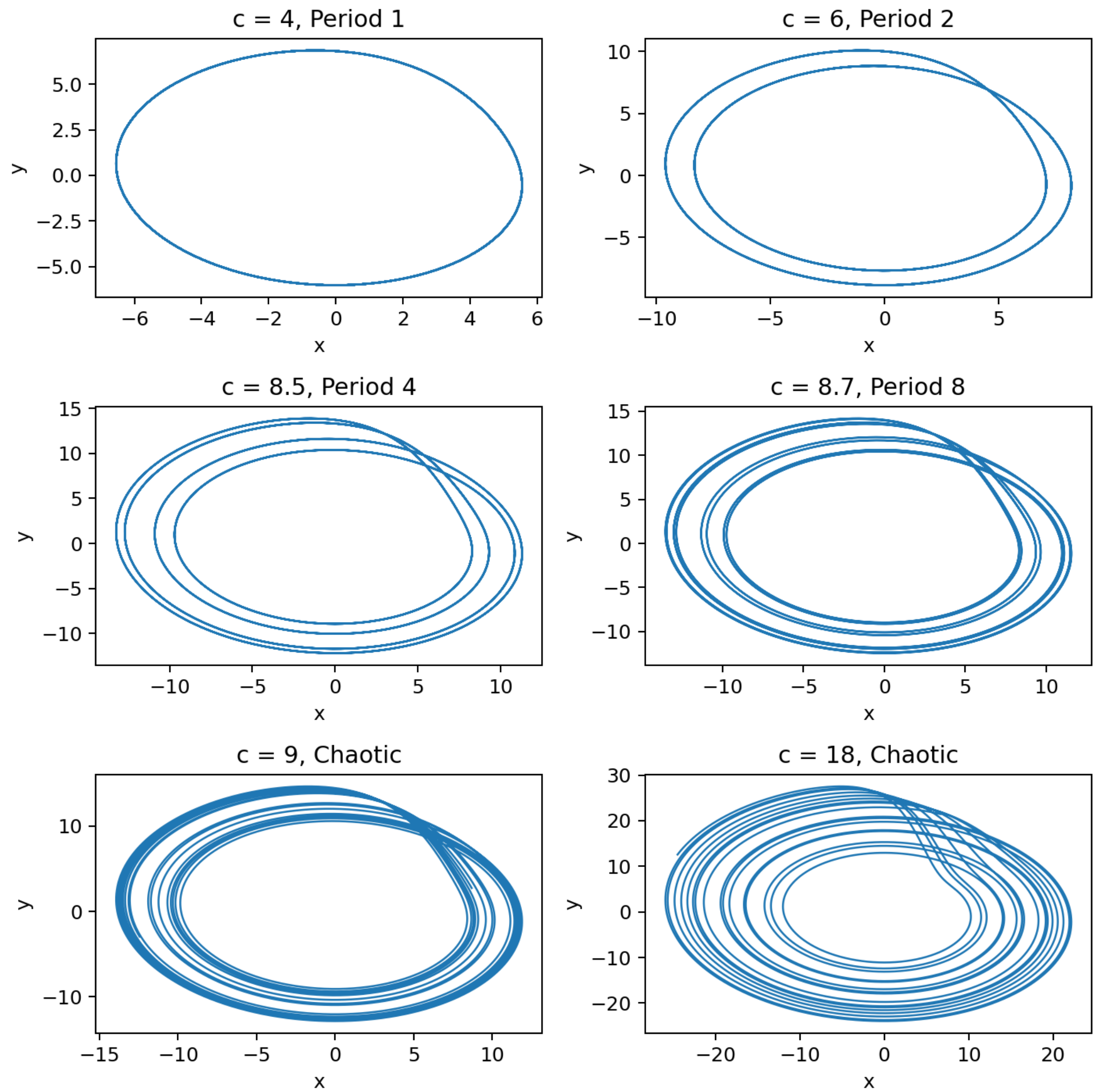

Figure 12: Plots of y vs x for varying c

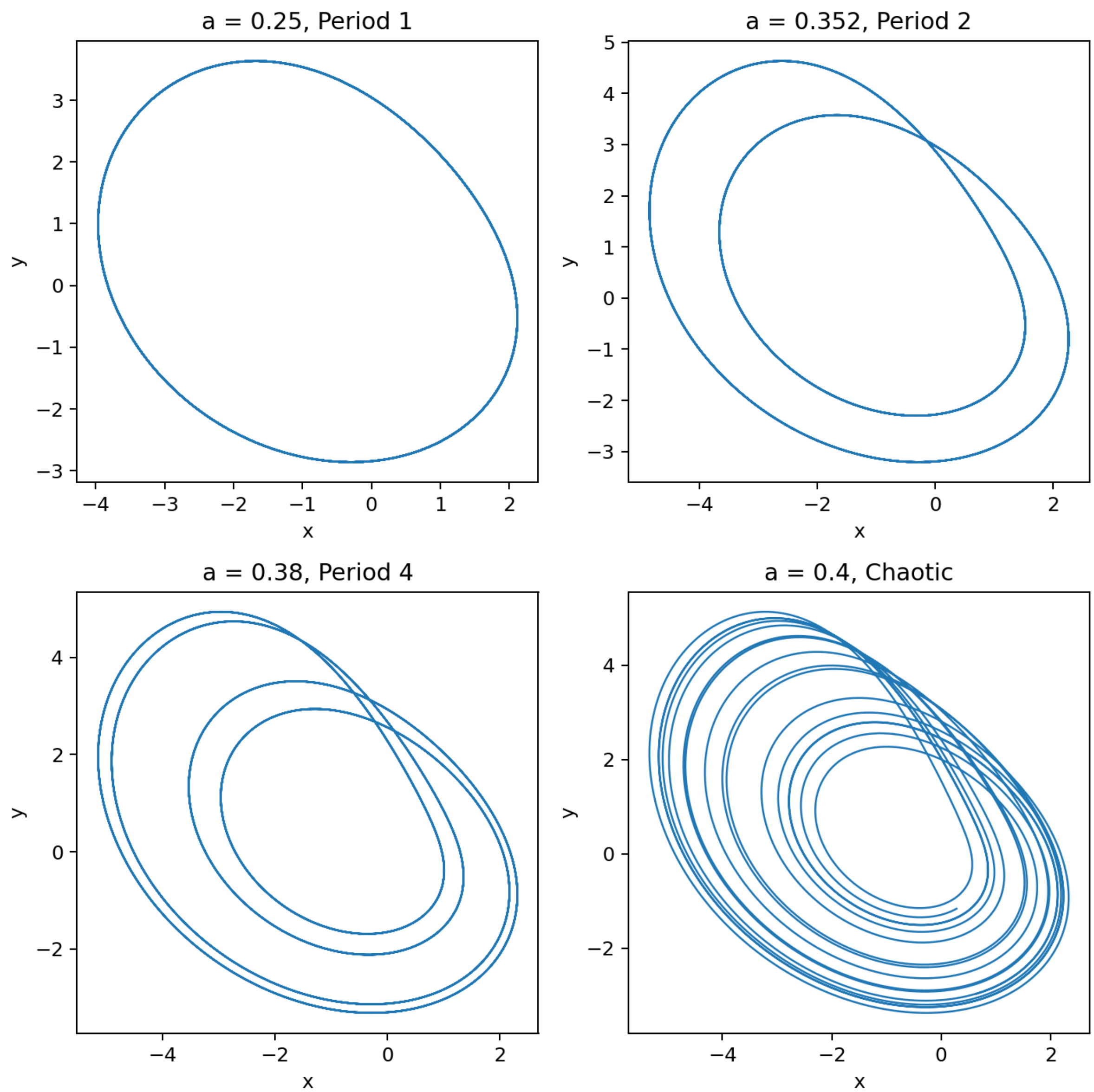

Figure 13: Plots of y vs x for varying a

3 Code

3.1 Chaotic Dynamics

```
import numpy as np
import matplotlib.pyplot as plt
a = 0.2
b = 0.2
c = 5
t = 0
T = 200
h = 0.01
def derivative(r,t):
    x = r[0]
    y = r[1]
    z = r[2]
    return np.array([- y - z, x + a * y, b + z * (x - c)])
time = np.array([])
x = np.array([])
```

```
y = np.array([])
z = np.array([])
r = np.array([1.000, 1.000, 1.000])
while t <= T :

        time = np.append(time, t)
        z = np.append(z, r[2])
        y = np.append(y, r[1])
        x = np.append(x, r[0])

        k1 = h*derivative(r,t)
        k2 = h*derivative(r+k1/2,t+h/2)
        k3 = h*derivative(r+k2/2,t+h/2)
        k4 = h*derivative(r+k3,t+h)
        r += (k1+2*k2+2*k3+k4)/6

        t = t + h
fig = plt.figure(figsize = (8,8),dpi=100)
ax = plt.axes(projection='3d')
ax.grid()

ax.plot3D(x, y, z)
ax.set_title(r' $r_{o} = (1,1,1)$, $a=0.2, b=0.2, c=5$')

ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
fig.tight_layout()
plt.show()
fig.savefig('Rossler_13d.png',layout='tight')

plt.rcParams.update({'font.size': 10})
fig, (ax1, ax2,ax3) = plt.subplots(3, 1,figsize=(8,8))
ax1.plot(np.arange(0,200.01,0.01),x)
ax1.set_title(r'Time Series for x coordinate')
ax1.set_xlabel(r'$t$')
ax1.set_ylabel('x')


ax2.plot(np.arange(0,200.01,0.01),y)
ax2.set_title(r'Time Series for y coordinate')
ax2.set_xlabel(r'$t$')
ax2.set_ylabel('x')
fig.tight_layout()

ax3.plot(np.arange(0,200.01,0.01),z)
ax3.set_title(r'Time Series for z coordinate')
ax3.set_xlabel(r'$t$')
ax3.set_ylabel('z')
fig.tight_layout()
fig.savefig('Rossler_1dt.png')
plt.show()
```

3.2 Bifurcation Diagrams

```

import numpy as np
import matplotlib.pyplot as plt


a = 0.2
b = 0.2
c = 0
x_max = []
x_min= []
y_max = []
y_min = []
z_max = []
z_min = []
r = []

def rk4(x,y):
    k1,l1,m1 = derivative(x,y,z)
```

```
    k2,l2,m2 = derivative(x + k1*h/2, y + l1*h/2, z + m1*h/2)

    k3,l3,m3 = derivative(x + k2*h/2, y + l2*h/2, z + m2*h/2)

    k4,l4,m4 = derivative(x + k3*h, y + l3*h, z+l3*h)

    k = (h/6)*(k1 + 2*k2 + 2*k3 + k4)
    l = (h/6)*(l1 + 2*l2 + 2*l3 + l4)
    m = (h/6)*(m1 + 2*m2 + 2*m3 + m4)

    return k,l,m

def derivative(x,y,z):
  dxdt = -y - z
  dydt = x + a*y
  dzdt = b + z*(x - c)
  return dxdt,dydt,dzdt
while c<10:

  x = 1.0
  y = 1.0
  z = 1.0
  t = 0.0
  h = 0.01

  X,Y,Z,T = [],[],[],[]

  dxdt,dydt,dzdt = derivative(x,y,z)

  while t < 400.0:

    t = t+h
    k,l,m = rk4(x,y)
    x = x+k
    y = y+l
    z = z+m

    X.append(x),Y.append(y),Z.append(z),T.append(t)

  min_x, max_x = [],[]
  min_y, max_y = [],[]
  min_z, max_z = [],[]

  for i in range(20000,(len(X)-1)):
    if(X[i-1] > X[i] < X[i + 1]):
      min_x.append(X[i])
    if(X[i-1] < X[i] > X[i + 1]):
      max_x.append(X[i])

  x_max.append(list(set(max_x)))
  x_min.append(list(set(min_x)))

  for i in range(20000,(len(Y)-1)):
    if(Y[i-1] > Y[i] < Y[i + 1]):
      min_y.append(Y[i])
    if(Y[i-1] < Y[i] > Y[i + 1]):
      max_y.append(Y[i])

  y_max.append(list(set(max_y)))
  y_min.append(list(set(min_y)))

  for i in range(20000,(len(Z)-1)):
    if(Z[i-1] > Z[i] < Z[i + 1]):
      min_z.append(Z[i])
    if(Z[i-1] < Z[i] > Z[i + 1]):
      max_z.append(Z[i])

  z_max.append(list(set(max_z)))
  z_min.append(list(set(min_z)))

  r.append(c)
  c = c+0.005
```

```
  print (c)


fig=plt.figure(figsize = (12,6),dpi=100)

for i in range(len(x_max)):
  plt.scatter([r[i] for j in range(len(x_max[i]))],x_max[i],s=0.03,c='black', marker = '.')
plt.title('Rossler System for a=0.2, b=0.2')
plt.xlabel('c')
plt.ylabel('x')
plt.show()
fig.savefig('X-Bif.png')
```