HW9

May 17, 2023

1 Exercise Set 9

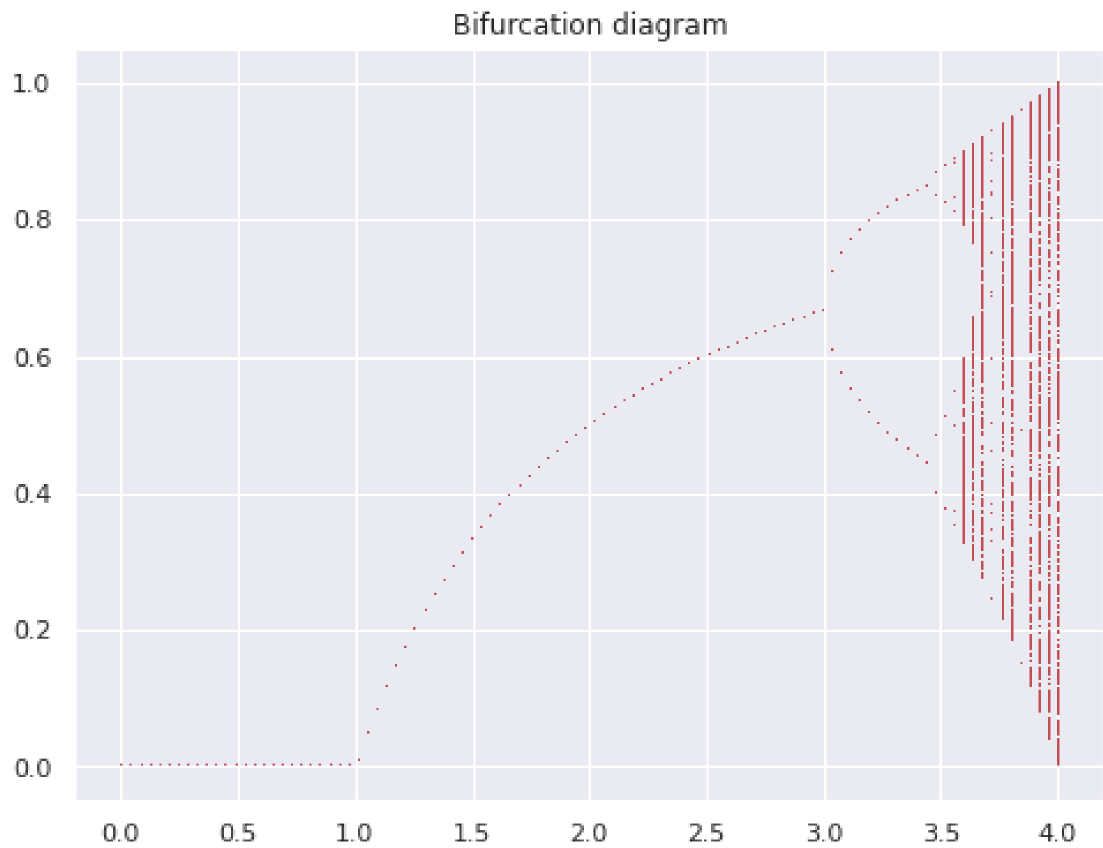
1.1 Mohaddeseh Mozaffari

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.integrate import odeint
from numba import njit
sns.set()
```

2 Q1:

[ ]:
```
Mu = np.linspace(0, 4, 100)
M = 500
N = 1000
```

[ ]:
```
fig, ax = plt.subplots(figsize=(8,6))
for mu in Mu:
    for m in range(M):
        x = np.random.random()
        for n in range(N):
            x = x*mu*(1-x)
        ax.plot(mu, x, ',r', alpha=1)
ax.set_title("Bifurcation diagram")
plt.show()
```



3 Q2:

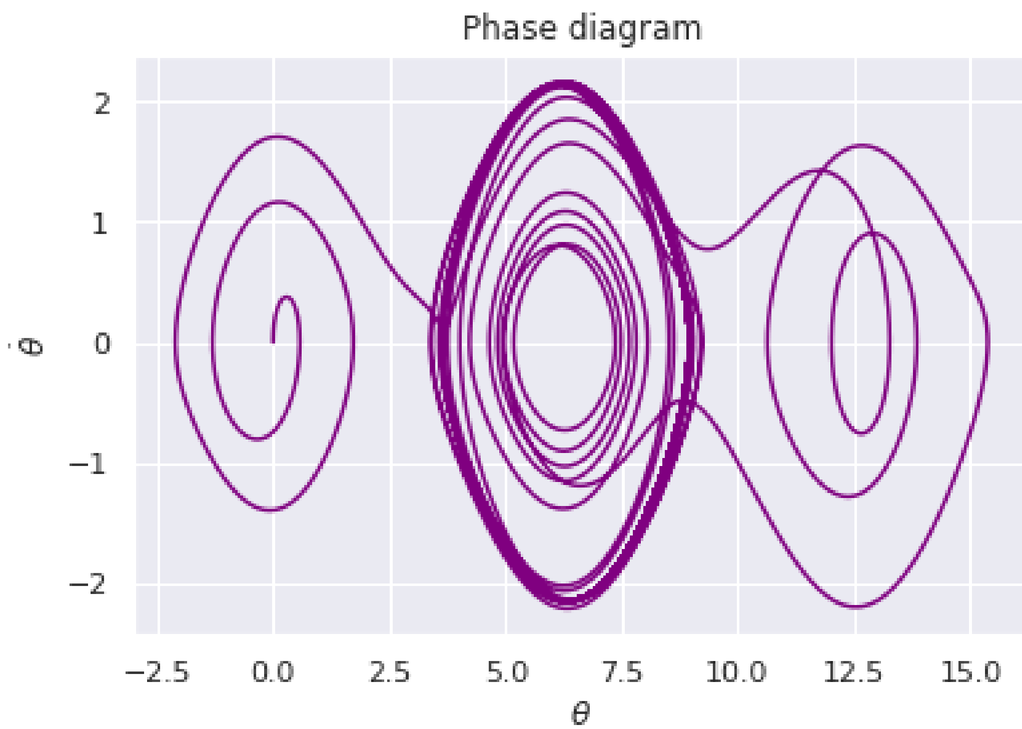
```
[ ]: w0 = 1
     a = 0.2
     f = 0.52
     w = 0.666
     dt = 0.01
     T = np.arange(0, 1000, dt)
     x0 = [0,0]
```

```
[ ]: def theta2(x,t, w0, a, f, w):
         return (x[1], -(w0**2)*np.sin(x[0])- a*x[1]+f*np.cos(w*t))
```

```
[ ]: sol = odeint(theta2, x0, T, args=(w0, a, f, w))
```

```
[ ]: plt.plot(sol[:, 0], sol[:, 1], color="purple")
     plt.xlabel(r"$\theta$")
     plt.ylabel(r"$\dot{\theta}$")
     plt.title("Phase diagram")
```

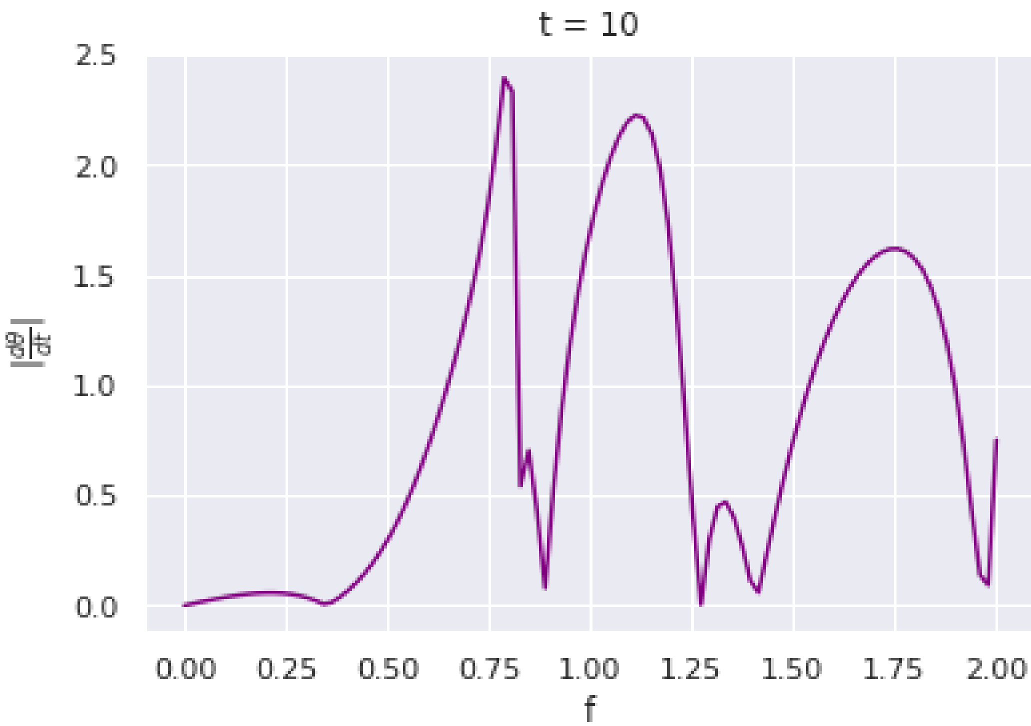
```
plt.show()
```



[ ]:
```
F = np.linspace(0, 2, 100)
```

[ ]:
```
theta_dot = []
for ff in F:
    sol = odeint(theta2, x0, T, args=(w0, a, ff, w))
    theta_dot.append(abs(sol[1000, 1]))
```

[ ]:
```
plt.plot(F, theta_dot, color="purple")
plt.xlabel("f")
plt.ylabel(r"$|\frac{d\theta}{dt}|$")
plt.title("t = 10")
plt.show()
```



4 Q3:

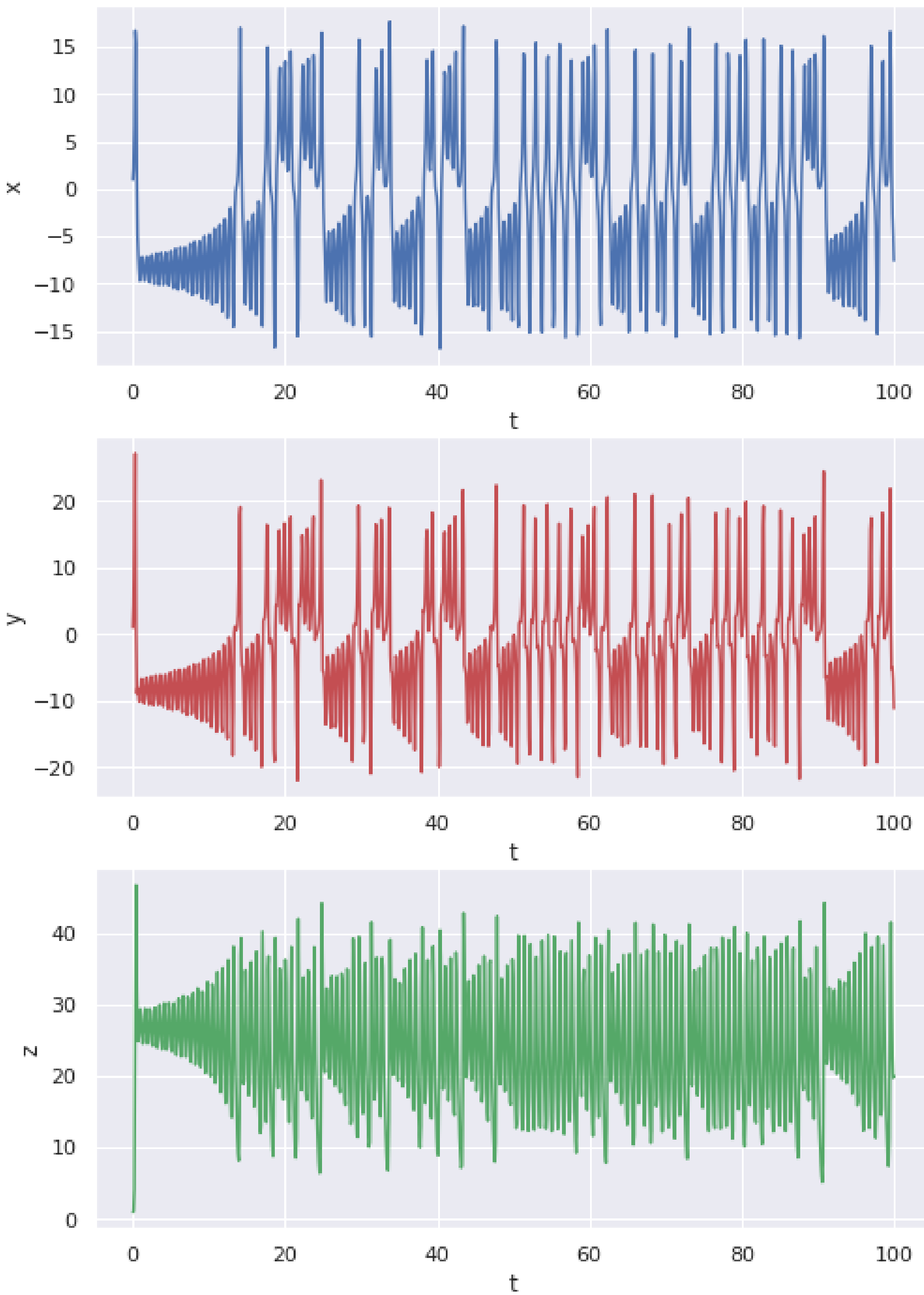
[ ]:
```
def f(x, t):
    dxdt = 10*(x[1] - x[0])
    dydt = -x[0]*x[2] + 28*x[0] - x[1]
    dzdt = x[0]*x[1] - (8/3)*x[2]
    return np.array([dxdt, dydt, dzdt])
```

[ ]:
```
t = np.linspace(0, 100, 1000)
x0 = np.array([1, 1, 1])
```

[ ]:
```
sol = odeint(f, x0, t)
X = sol[:, 0]
Y = sol[:, 1]
Z = sol[:, 2]
```

[ ]:
```
fig, ax = plt.subplots(3, 1, figsize=(8, 12))
ax[0].plot(t, X, 'b')
ax[0].set_xlabel('t')
ax[0].set_ylabel('x')
ax[1].plot(t, Y, 'r')
ax[1].set_xlabel('t')
```

```
ax[1].set_ylabel('y')
ax[2].plot(t, Z, 'g')
ax[2].set_xlabel('t')
ax[2].set_ylabel('z')
plt.show()
```


x
15
10
5
0
−5
−10
−15
0
20
40
60
80
100
t
y
20
10
0
−10
−20
0
20
40
60
80
100
t
z
40
30
20
10
0
0
20
40
60
80
100
t

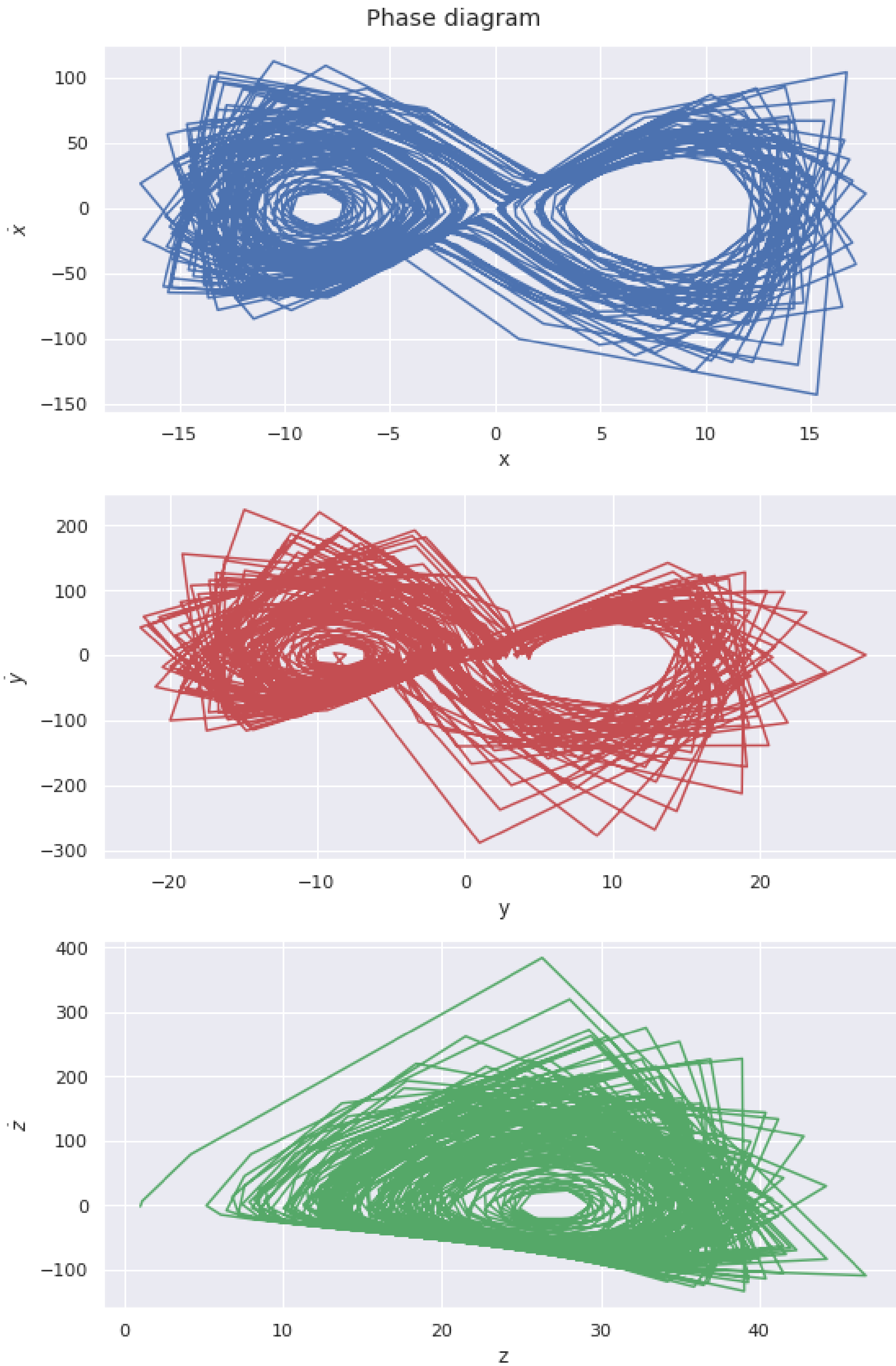
[ ]:
```
dxdt = 10*(Y - X)
dydt = -X*Z + 28*X - Y
dzdt = X*Y - (8/3)*Z
```

[ ]:
```
fig, ax = plt.subplots(3, 1, figsize=(8, 12))
ax[0].plot(X, dxdt, 'b')
ax[0].set_xlabel('x')
ax[0].set_ylabel(r'$\dot{x}$')

ax[1].plot(Y, dydt , 'r')
ax[1].set_xlabel('y')
ax[1].set_ylabel(r'$\dot{y}$')

ax[2].plot(Z, dzdt, 'g')
ax[2].set_xlabel('z')
ax[2].set_ylabel(r'$\dot{z}$')

plt.suptitle("Phase diagram")
plt.tight_layout()
plt.show()
```


Phase diagram
100
50
0
−50
−100
−150
−15
−10
−5
0
5
10
15
x
ẋ
200
100
0
−100
−200
−300
−20
−10
0
10
20
y
ẏ
400
300
200
100
0
−100
0
10
20
30
40
z
ż

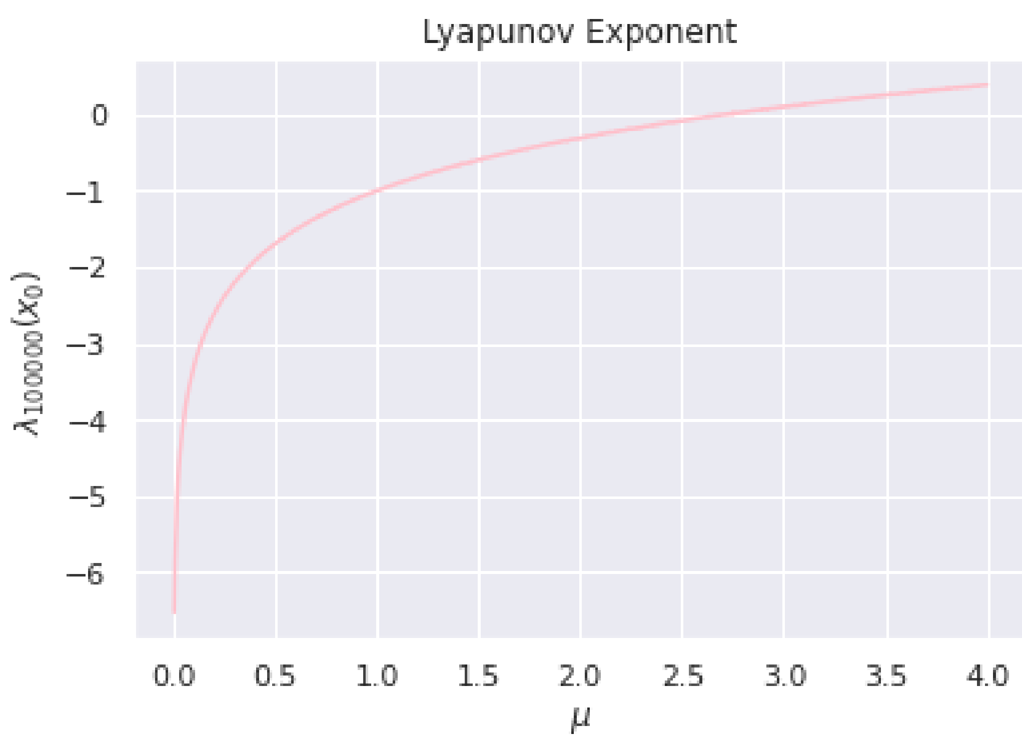
5 Q4:

5.1 A)

[ ]:
```
Mu = np.linspace(0,4, 1000)
N = 100000
```

[ ]:
```
L = []
for mu in Mu:
    l = 0
    for _ in range(N):
        x = np.random.random()
        l += np.log(abs(mu- 2*mu * x))
    l /= N
    L.append(l)
```

[ ]:
```
plt.plot(Mu, L, color="pink")
plt.title('Lyapunov Exponent')
plt.xlabel(r'$\mu$')
plt.ylabel(rf"$\lambda_{ {N} }(x_0)$")
plt.show()
```



5.2 B)

5.2.1 Q2:

```
[ ]: w0 = 1
a = 0.2
f = 0.52
w = 0.666
dt = 0.01
T = np.arange(0, 1000, dt)
x0 = [0.1 ,0.1]
num_initial = 100
epsilon = 1e-6
```

```
[ ]: def theta2(x,t, w0, a, f, w):
    return (x[1], -(w0**2)*np.sin(x[0])- a*x[1]+f*np.cos(w*t))
```

```
[ ]: sol = odeint(theta2, x0, T, args=(w0, a, f, w))
```

```
[ ]: l = 0
for n in range(num_initial):
    x_int = [epsilon*np.random.randn(), 0]
    sol_int = odeint(theta2, x_int, T, args=(w0, a, f, w))
    delta_n = abs(sol_int[:, 0] - sol[:, 0])
    l = np.mean(np.log(delta_n/x_int[0]))
l /= num_initial
```

```
[ ]: print("Lyapunov exponent for Q2 is: ", round(l,3))
```

```
Lyapunov exponent for Q2 is:  0.171
```

5.2.2 Q3:

```
[ ]: def f(x, t):
    dxdt = 10*(x[1] - x[0])
    dydt = -x[0]*x[2] + 28*x[0] - x[1]
    dzdt = x[0]*x[1] - (8/3)*x[2]
    return np.array([dxdt, dydt, dzdt])
```

```
[ ]: t = np.linspace(0, 100, 1000)
x0 = np.array([0.1, 0.1, 0.1])
```

```
[ ]: sol = odeint(f, x0, t)
X = sol[:, 0]
```

[ ]:
```python
num_initial = 1000
epsilon = 1e-6
```

[ ]:
```python
lx , ly, lz = 0, 0, 0
for n in range(num_initial):
    x_int = x0 + epsilon*np.random.randn(3)
    sol_int = odeint(f, x_int, t)
    delta_n_x = abs(sol_int[:, 0] - sol[:, 0])
    delta_n_y = abs(sol_int[:, 1] - sol[:, 1])
    delta_n_z = abs(sol_int[:, 2] - sol[:, 2])
    lx = np.mean(np.log(delta_n_x/x_int[0]))
    ly = np.mean(np.log(delta_n_y/x_int[1]))
    lz = np.mean(np.log(delta_n_z/x_int[2]))
lx /= num_initial
ly /= num_initial
lz /= num_initial
```

[ ]:
```python
print("Lyapunov exponent for Q3 x component is: ",round(lx, 5))
print("Lyapunov exponent for Q3 y component is: ",round(ly, 5))
print("Lyapunov exponent for Q3 z component is: ",round(lz, 5))
```

```
Lyapunov exponent for Q3 x component is:  0.0011
Lyapunov exponent for Q3 y component is:  0.00126
Lyapunov exponent for Q3 z component is:  0.00126
```

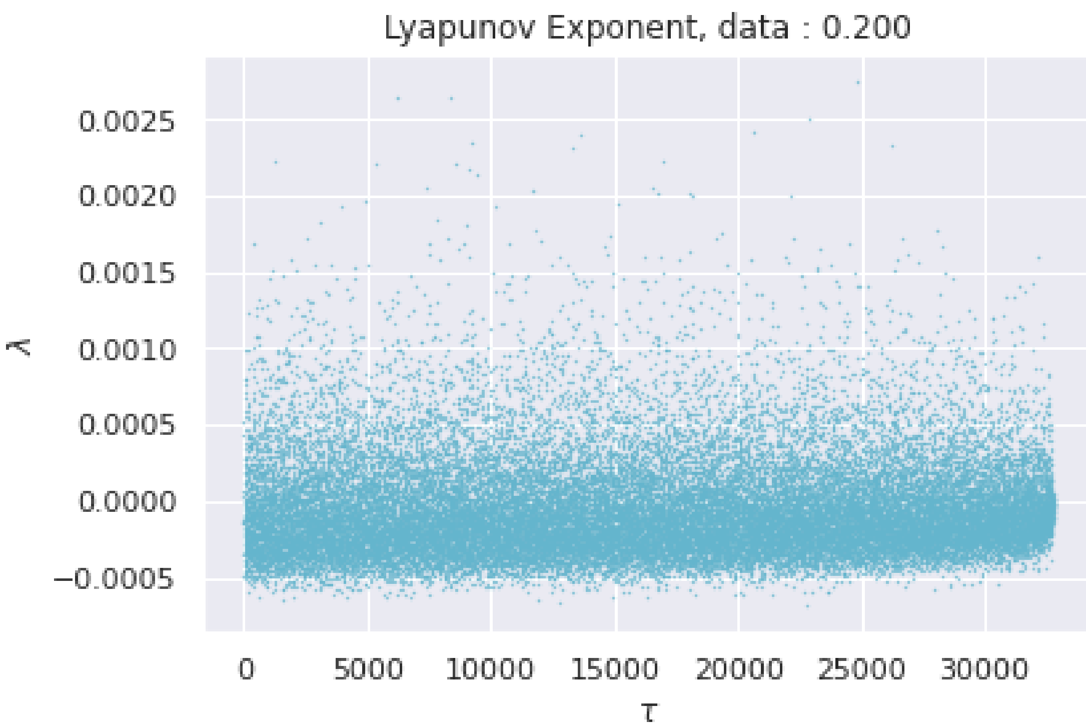
6 Q5:

[ ]:
```python
@njit
def lyapunov_exp(data):
    L = []
    N = len(data)
    def d_n(data, n, tau):
        return abs(data[n+tau] - data[n])
    for tau in range(N):
        l = 0
        d0 = d_n(data, 0, tau)
        if d0 != 0:
            for n in range(1, N-tau):
                dn = d_n(data, n, tau)
                l += np.log(dn/d0)/n
            l /= N
            L.append(l)
    return L
```

[ ]:
```python
data1 = np.loadtxt("/home/mohaddeseh/Documents/Programing/Computational/HW9/
 ↪chaotic_data/0.200.txt")
```

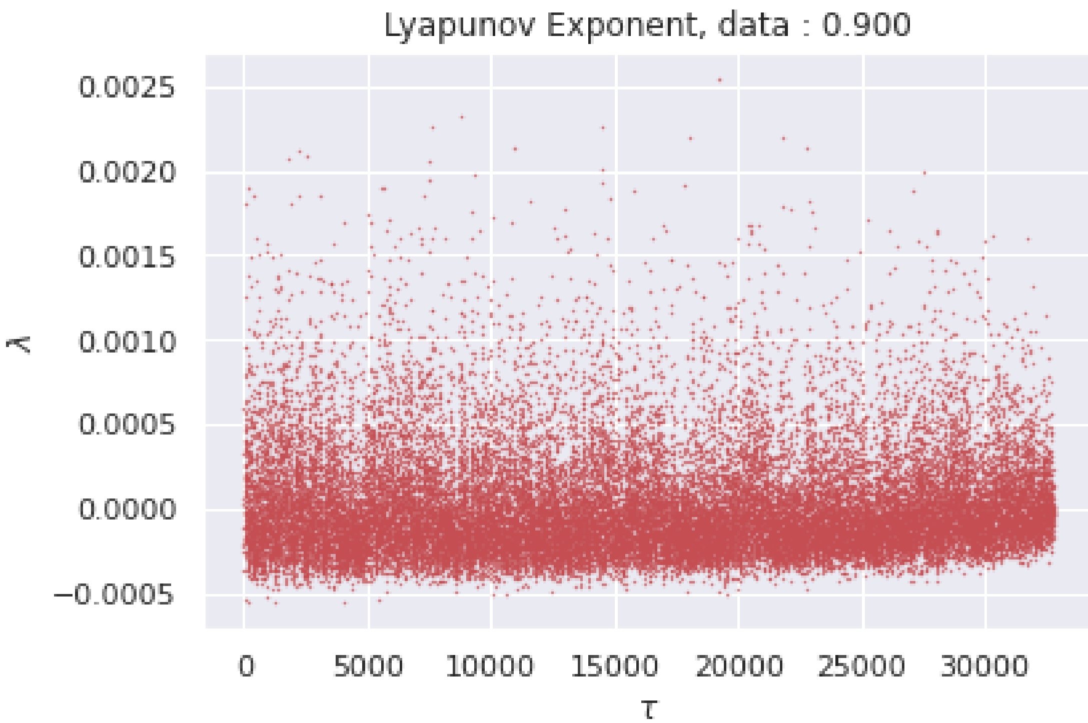
```
data1 = data1[:,1]
l1 = lyapunov_exp(data1)
```

[ ]:
```
plt.scatter(range(len(l1)), l1, s=0.1, color="c")
plt.title('Lyapunov Exponent, data : 0.200')
plt.ylabel('$\lambda$')
plt.xlabel(r'$\tau$')
plt.show()
```



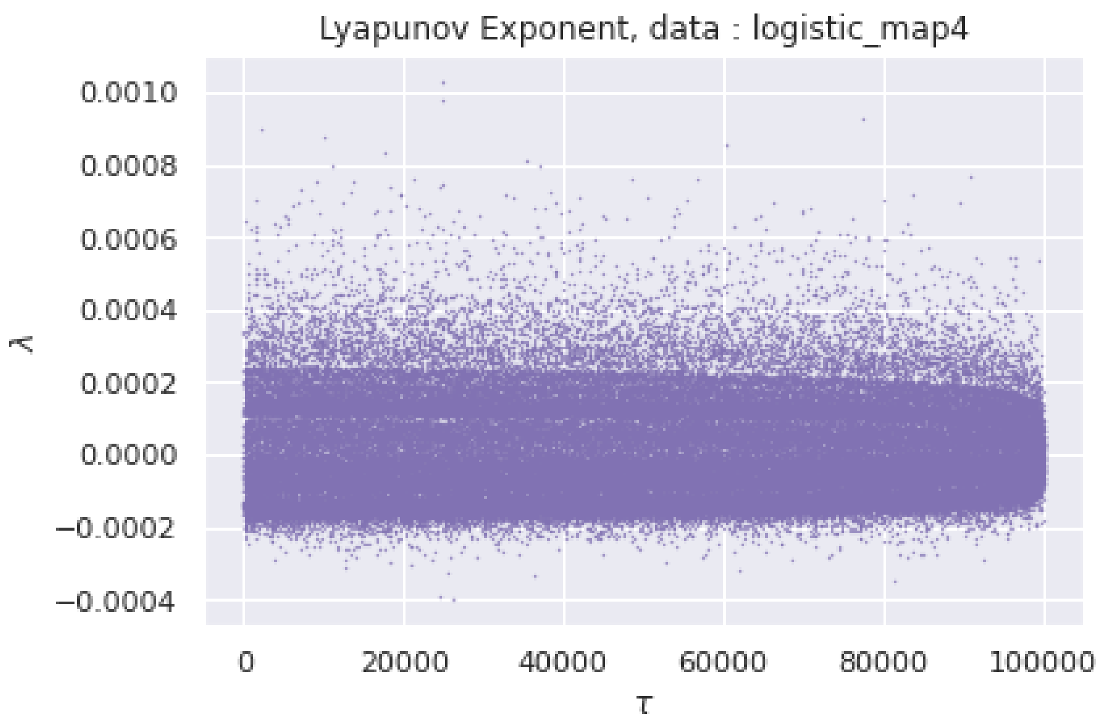
[ ]:
```
data2 = np.loadtxt("/home/mohaddeseh/Documents/Programing/Computational/HW9/
↪chaotic_data/0.900.txt")
data2 = data2[:, 1]
l2 = lyapunov_exp(data2)
```

[ ]:
```
plt.scatter(range(len(l2)), l2, s=0.1, color="r")
plt.title('Lyapunov Exponent, data : 0.900')
plt.ylabel('$\lambda$')
plt.xlabel(r'$\tau$')
plt.show()
```



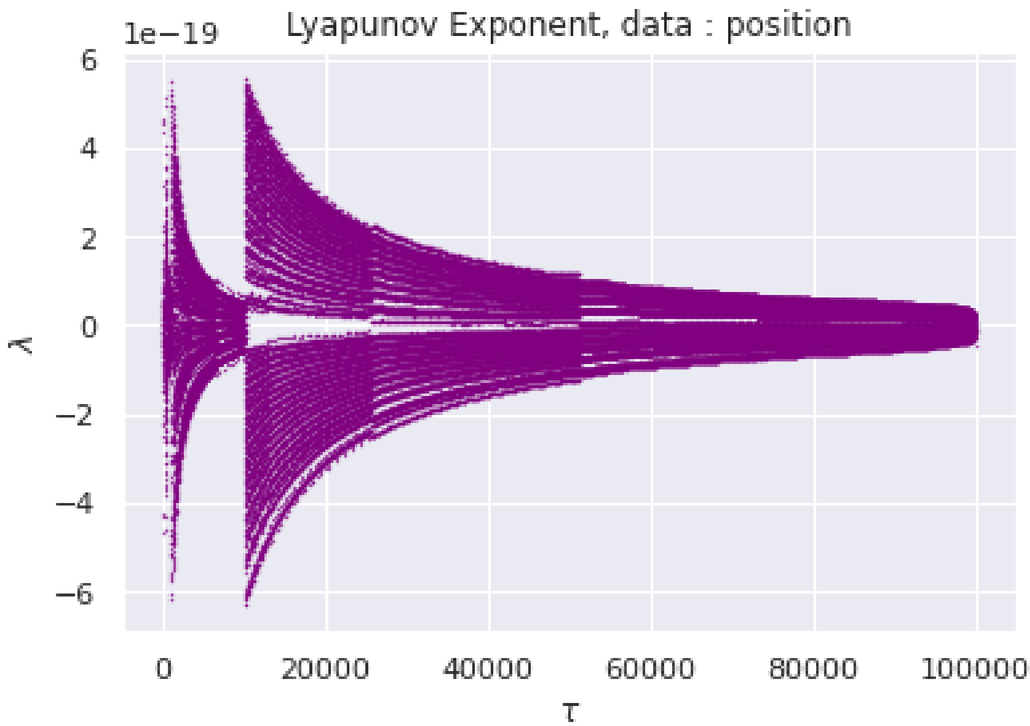

```
[ ]: data3 = np.loadtxt("/home/mohaddeseh/Documents/Programing/Computational/HW9/
     ↪chaotic_data/logistic_map4.txt")
     data3 = data3[:, 1]
     l3 = lyapunov_exp(data3)
```

```
[ ]: plt.scatter(range(len(l3)), l3, s=0.1, color="m")
     plt.title('Lyapunov Exponent, data : logistic_map4')
     plt.ylabel('$\lambda$')
     plt.xlabel(r'$\tau$')
     plt.show()
```

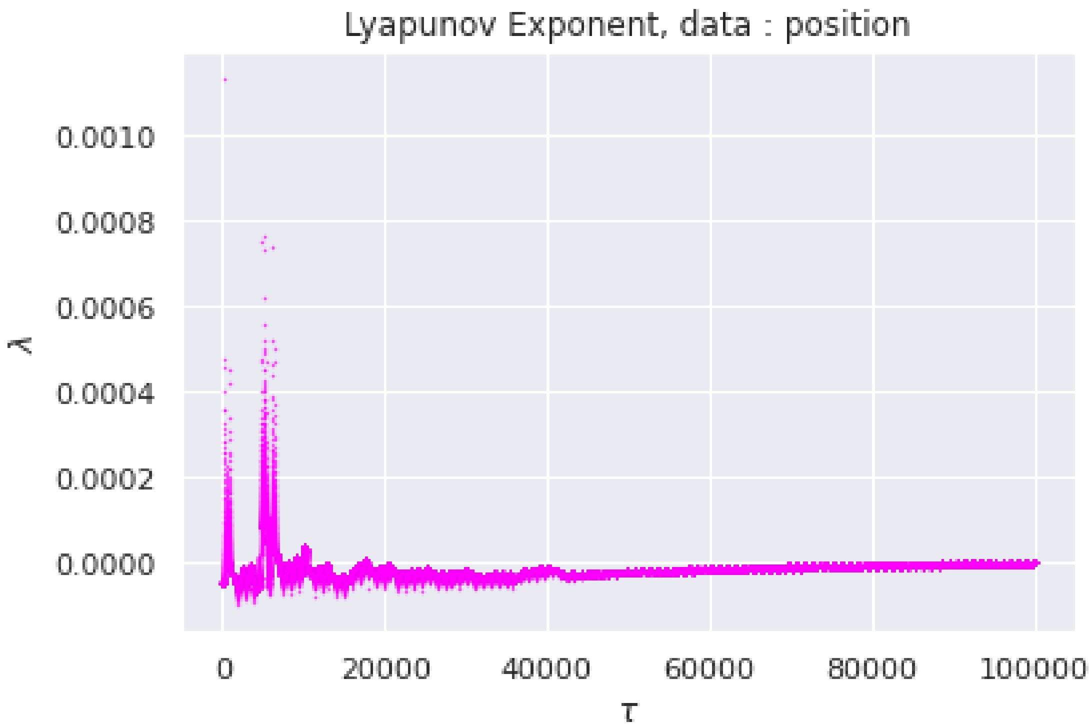


[ ]:
```
data4 = np.loadtxt("/home/mohaddeseh/Documents/Programing/Computational/HW9/
 ↪chaotic_data/position.txt")
data4_0 = data4[:, 0]
data4_1 = data4[:, 1]
l4_0 = lyapunov_exp(data4_0)
l4_1 = lyapunov_exp(data4_1)
```

[ ]:
```
plt.scatter(range(len(l4_0)), l4_0, s=0.1, color="purple")
plt.title('Lyapunov Exponent, data : position')
plt.ylabel('$\lambda$')
plt.xlabel(r'$\tau$')
plt.show()
```

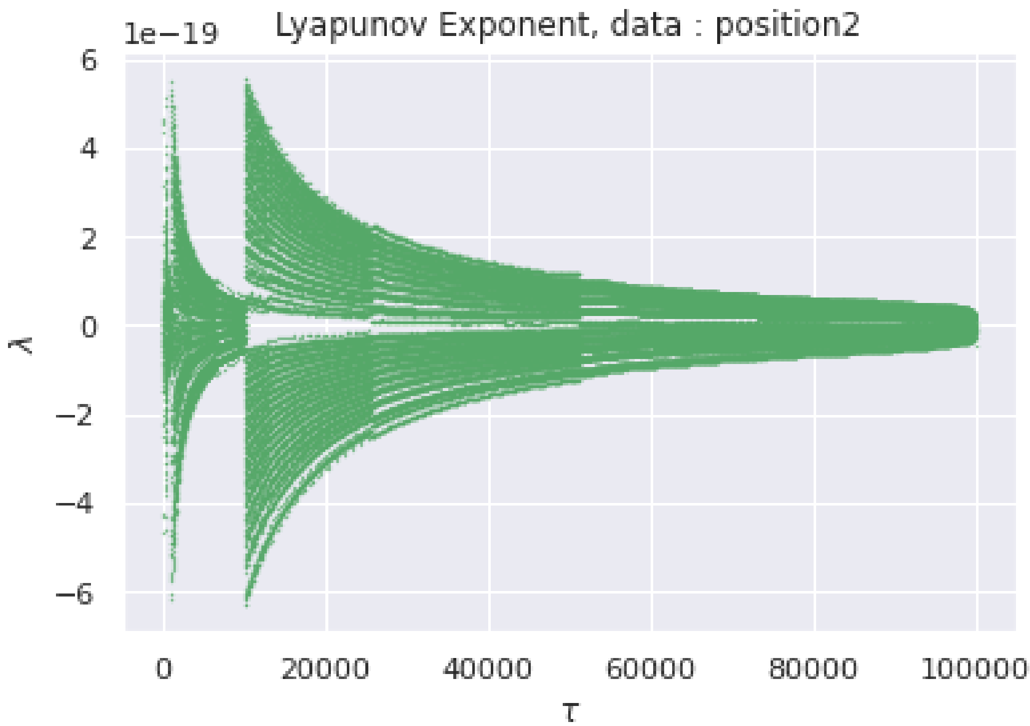


[ ]:
```
plt.scatter(range(len(l4_1)), l4_1, s=0.1, color="magenta")
plt.title('Lyapunov Exponent, data : position')
plt.ylabel('$\lambda$')
plt.xlabel(r'$\tau$')
plt.show()
```

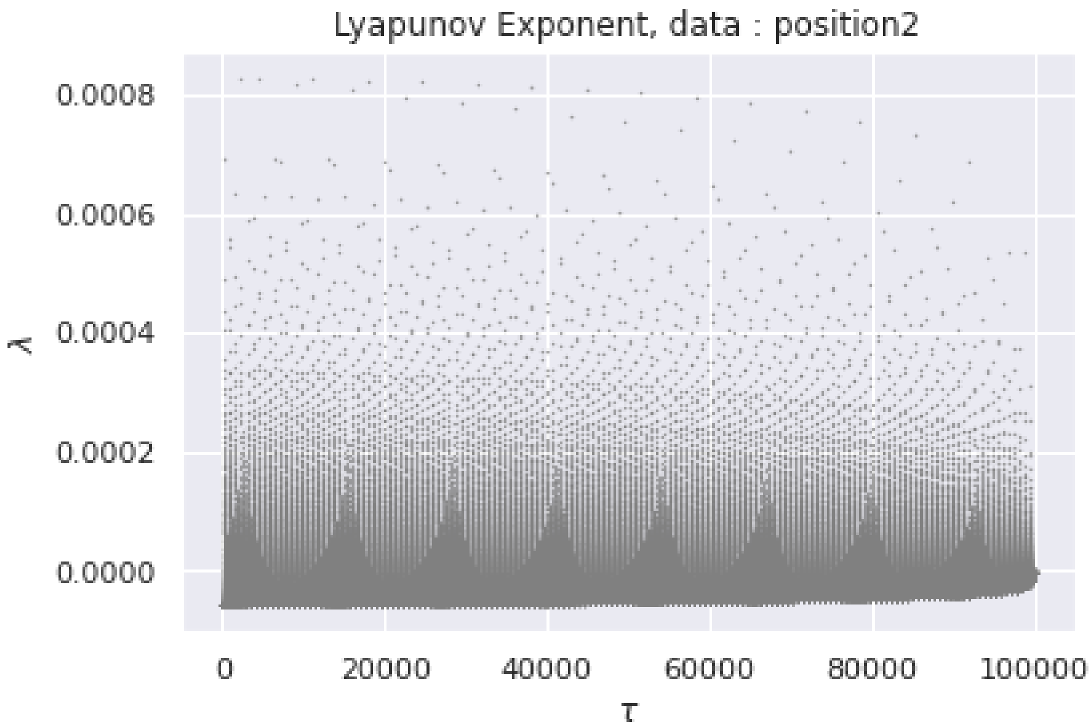



```
[ ]: data5 = np.loadtxt("/home/mohaddeseh/Documents/Programing/Computational/HW9/
      ↪chaotic_data/position2.txt")
     data5_0 = data5[:, 0]
     data5_1 = data5[:, 1]
     l5_0 = lyapunov_exp(data5_0)
     l5_1 = lyapunov_exp(data5_1)
```

```
[ ]: plt.scatter(range(len(l5_0)), l5_0, s=0.1, color="g")
     plt.title('Lyapunov Exponent, data : position2')
     plt.ylabel('$\lambda$')
     plt.xlabel(r'$\tau$')
     plt.show()
```

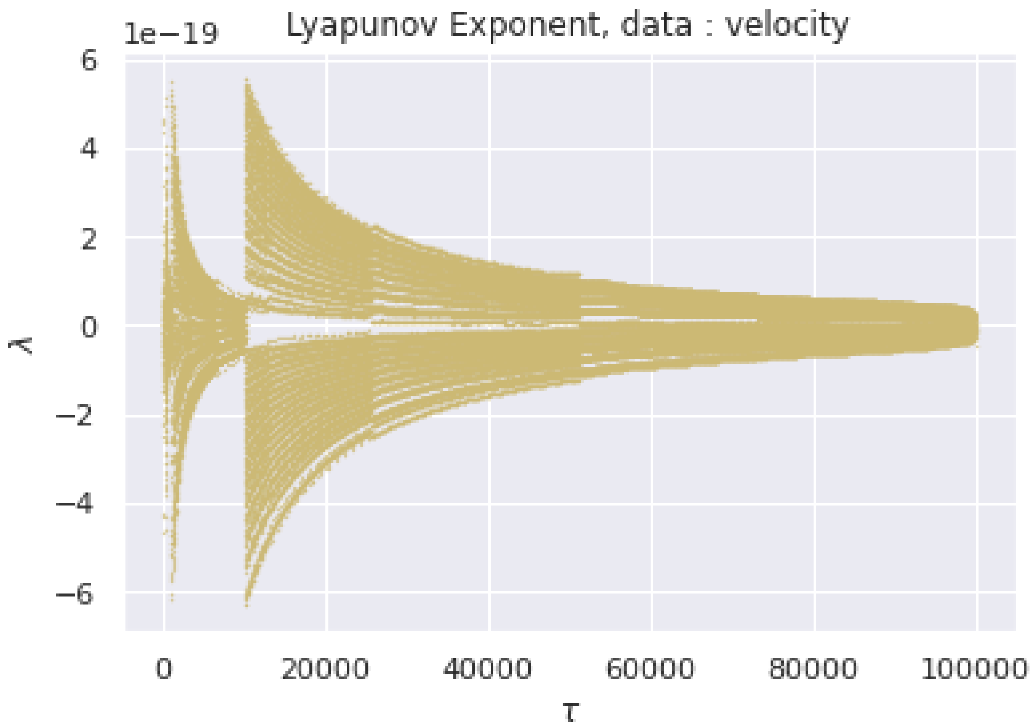


[ ]:
```
plt.scatter(range(len(l5_1)), l5_1, s=0.1, color="grey")
plt.title('Lyapunov Exponent, data : position2')
plt.ylabel('$\lambda$')
plt.xlabel(r'$\tau$')
plt.show()
```

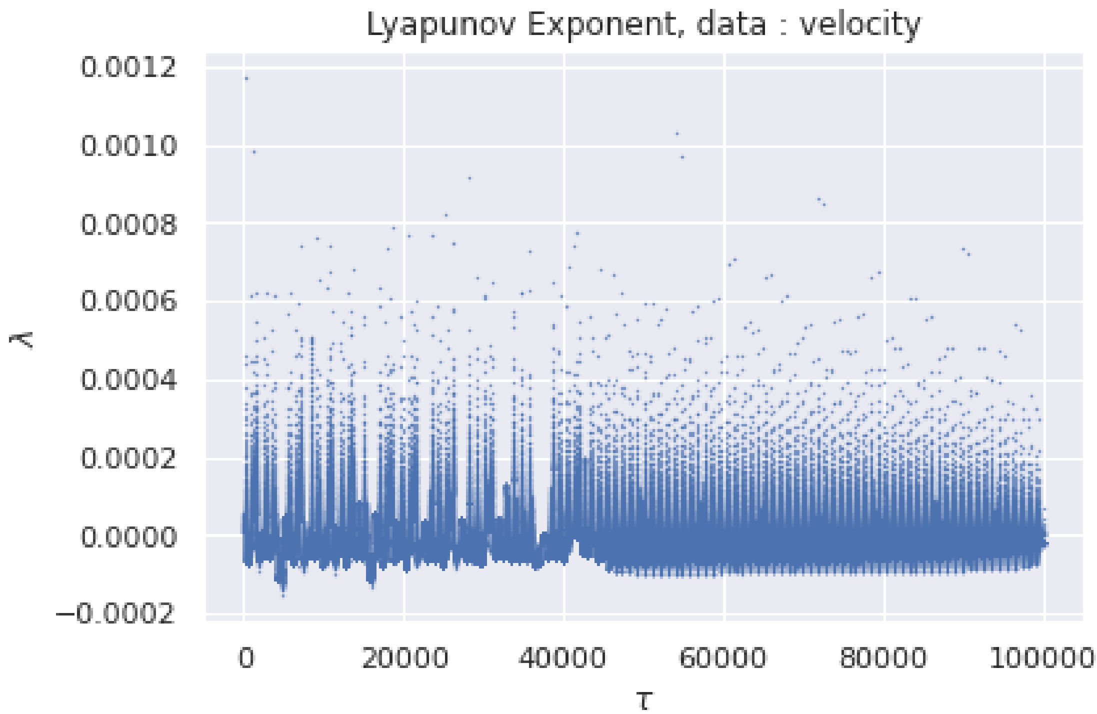


[ ]:
```
data6 = np.loadtxt("/home/mohaddeseh/Documents/Programing/Computational/HW9/
 ↪chaotic_data/velocity.txt")
data6_0 = data6[:, 0]
data6_1 = data6[:, 1]
l6_0 = lyapunov_exp(data6_0)
l6_1 = lyapunov_exp(data6_1)
```

[ ]:
```
plt.scatter(range(len(l6_0)), l6_0, s=0.1, color="y")
plt.title('Lyapunov Exponent, data : velocity')
plt.ylabel('$\lambda$')
plt.xlabel(r'$\tau$')
plt.show()
```



[ ]:
```
plt.scatter(range(len(l6_1)), l6_1, s=0.1, color="b")
plt.title('Lyapunov Exponent, data : velocity')
plt.ylabel('$\lambda$')
plt.xlabel(r'$\tau$')
plt.show()
```


Lyapunov Exponent, data : velocity
0.0012
0.0010
0.0008
0.0006
0.0004
0.0002
0.0000
−0.0002
λ
0
20000
40000
60000
80000
100000
τ

```
[ ]:
```