```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
from scipy.linalg import qr
import sys

title = '3.3cd'
sigma = 10
r = 28
b = 3

def dynamical_system(IC, t):
    x = IC[0]
    y = IC[1]
    z = IC[2]

    dxdt = sigma*(y-x)
    dydt = r*x-y-x*z
    dzdt =  x*y-b*z

    return [dxdt, dydt, dzdt]

# Numerical integration
T = 10**3
dt = 10**-3
t_array = np.arange(0,T,dt)

x = 0.01
y = 0.01
z = 0.01

IC = [x,y,z]
eqs = odeint(dynamical_system, IC, t_array)

cut_tail = 2000

x = eqs[cut_tail:,0]
y = eqs[cut_tail:,1]
z = eqs[cut_tail:,2]

new_t_array = t_array[cut_tail:]-cut_tail*dt
new_T = T-cut_tail*dt

I = np.identity(3)
Q = I.copy()

lambda_one = np.zeros_like(new_t_array)
lambda_two = lambda_one.copy()
lambda_three = lambda_one.copy()

print('No memory error')

for t in range(len(new_t_array)):

    J11 = -sigma
    J12 = sigma
    J13 = 0

    J21 = r-z[t]
    J22 = -1
    J23 = -x[t]

    J31 = y[t]
    J32 = x[t]
    J33 = -b

    J = np.array([[J11,J12,J13],\
                  [J21,J22,J23],\
                  [J31,J32,J33]])

    M = I+J*dt
    Q,R = qr(np.matmul(M,Q))

    lambda_one[t] = lambda_one[t-1] + np.log(np.absolute(R[0,0]))/new_T
    lambda_two[t] = lambda_two[t-1] + np.log(np.absolute(R[1,1]))/new_T
    lambda_three[t] = lambda_three[t-1] + np.log(np.absolute(R[2,2]))/new_T

    if t % 100000 == 0:
```

```
        print(t)

print('Done: QR')

lambda_one_converged = lambda_one[-1]
lambda_two_converged = lambda_two[-1]
lambda_three_converged = lambda_three[-1]

print('Lambda 1: ', lambda_one_converged)
print('Lambda 2: ', lambda_two_converged)
print('Lambda 3: ', lambda_three_converged)
print('Sum of lambdas: ', lambda_one[-1]+lambda_two[-1]+lambda_three[-1])

# # Trajectory
# fig1, ax1 = plt.subplots(subplot_kw={"projection": "3d"}, figsize=(7,7))
# ax1.plot(x, y, z, '-', linewidth=0.1)

# Lambdas
fig2, ax2 = plt.subplots(figsize=(7,7))
ax2.plot(new_t_array[1:], np.divide(lambda_one[1:],new_t_array[1:]), '-',
linewidth=1.5, label=r'$\lambda_1$ (conv. at $\approx$' +
'{})'.format(np.round(lambda_one_converged,2)))
ax2.plot(new_t_array[1:], np.divide(lambda_two[1:],new_t_array[1:]), '-',
linewidth=1.5, label=r'$\lambda_2$ (conv. at $\approx$' +
'{})'.format(np.round(lambda_two_converged,2)))
ax2.plot(new_t_array[1:], np.divide(lambda_three[1:],new_t_array[1:]), '-',
linewidth=1.5, label=r'$\lambda_3$ (conv. at $\approx$' +
'{})'.format(np.round(lambda_three_converged,2)))
ax2.set_xscale('log')
ax2.set_xlabel('$t$')
ax2.set_ylabel('$\sum \lambda_i$ /t')
ax2.set_box_aspect(1)
ax2.set_title(title)

plt.legend(loc="lower right", prop={'size': 10})
plt.savefig('Dynamical systems/DS HW3/3.3/' + title + '.png', bbox_inches='tight')
# plt.show()
```