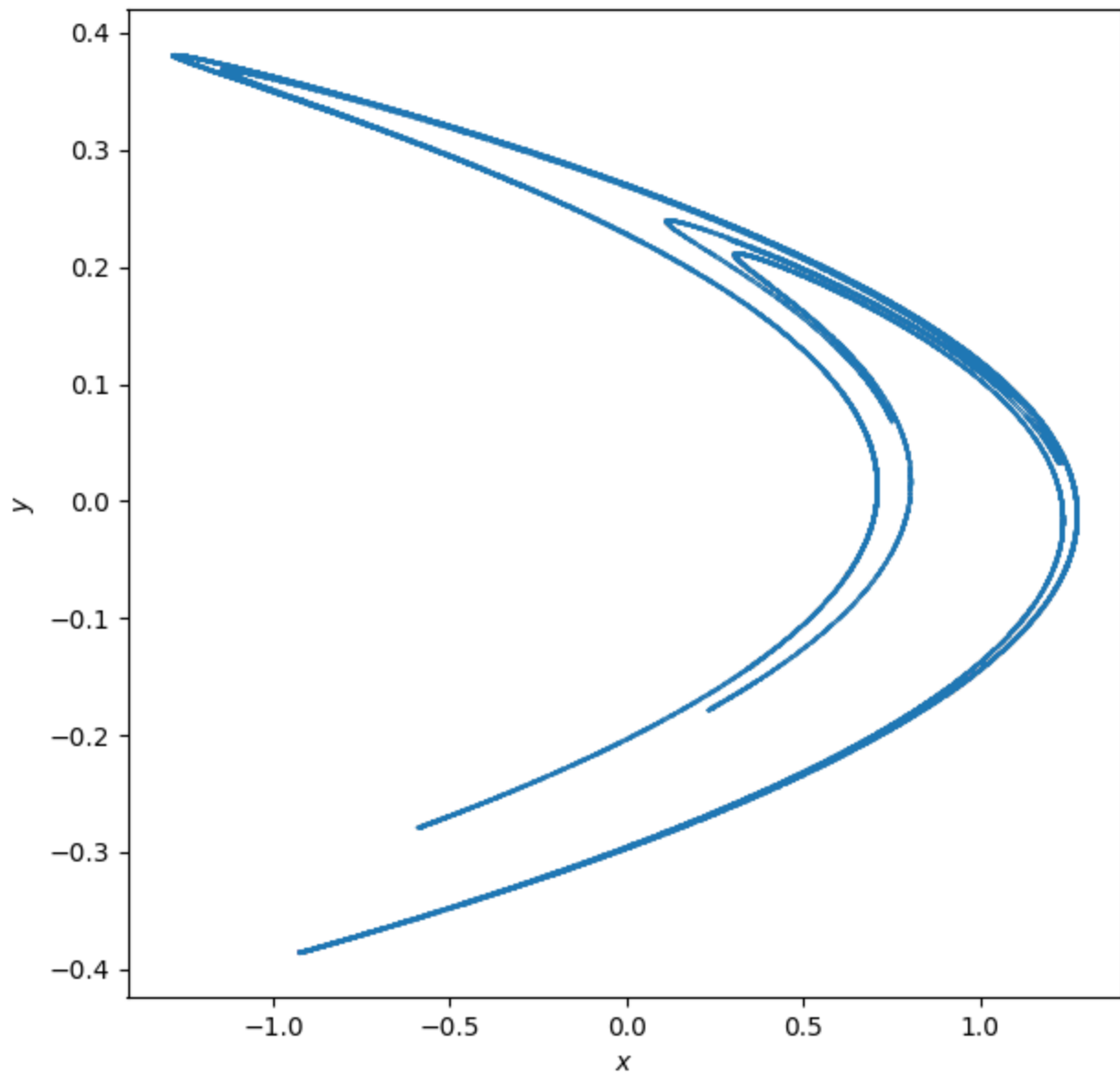
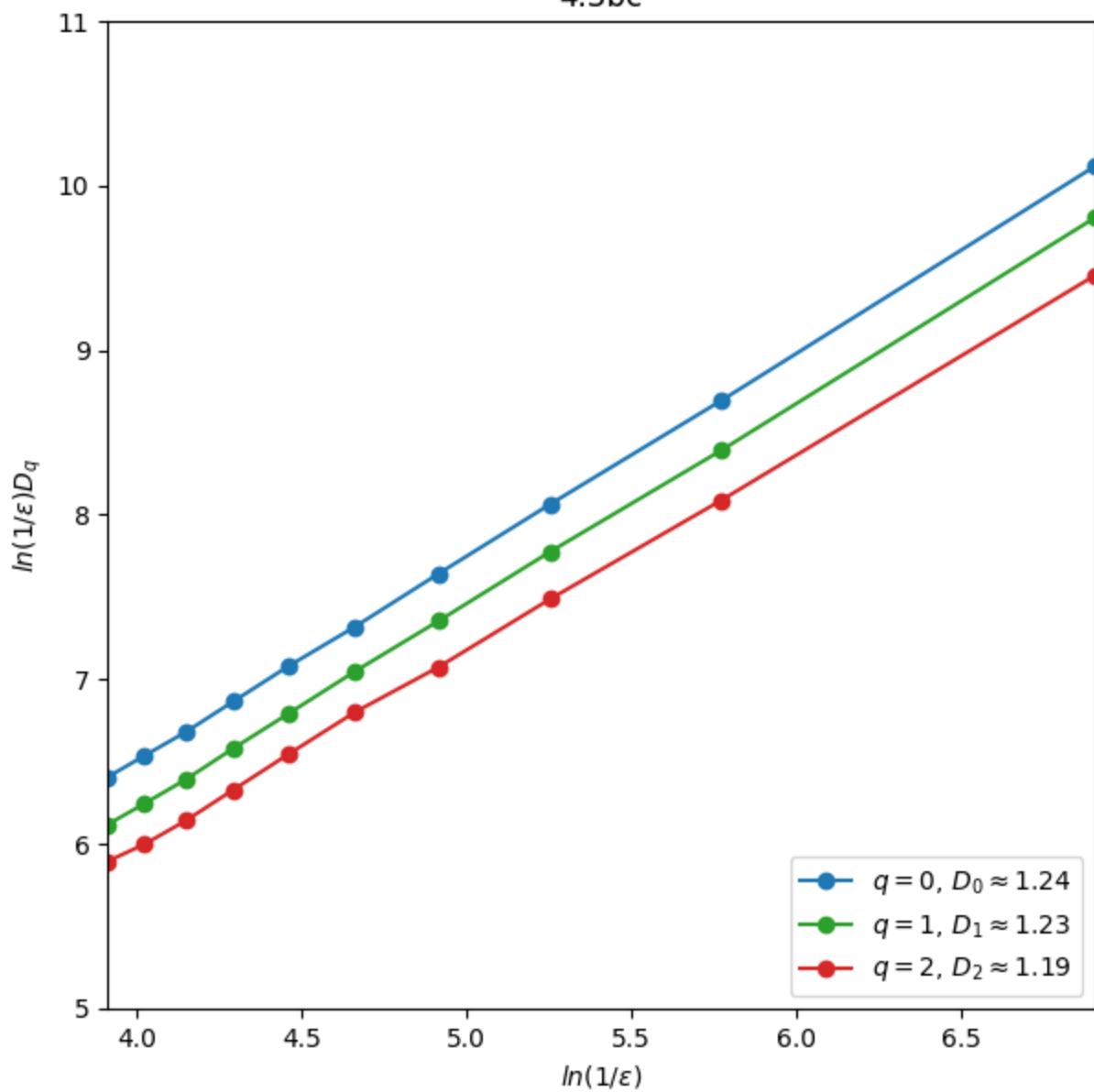


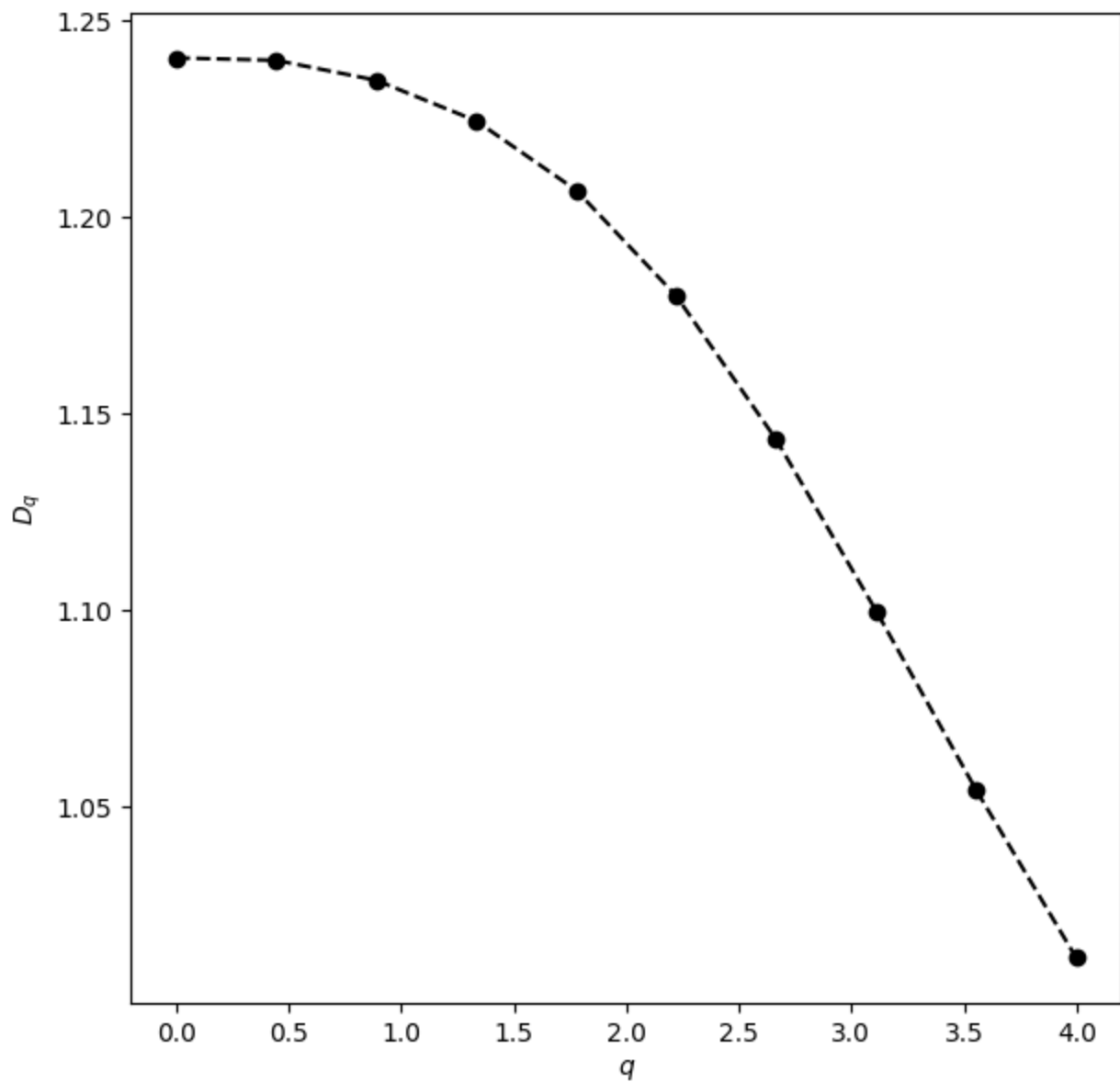
4.3a



4.3bc



4.3d



```
1 # Exercise 4.3a
2
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import sys
6
7 title = '4.3a'
8
9 a = 1.4
10 b = 0.3
11
12 cut_tail = 100
13
14 T = 1000
15 dt = 0.01
16 t = np.arange(0,T,dt)
17
18 x = np.zeros_like(t,dtype=float)
19 y = x.copy()
20
21
22 x[0] = (np.random.uniform()-0.5)
23 y[0] = (np.random.uniform()-0.5)
24
25 for t in range(len(x)-1):
26     x[t+1] = y[t]+1-a*x[t]**2
27     y[t+1] = b*x[t]
28
29 plt.figure(figsize=(7,7))
30 plt.plot(x[cut_tail:],y[cut_tail:],'.',markersize=0.2)
31 plt.title(title)
32 plt.xlabel('$x$')
33 plt.ylabel('$y$')
34 plt.savefig('Dynamical Systems/DS HW4/4.3/'+title+'.png')
35 plt.show()
```

```

1 # Exercise 4.3bc
2
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import sys
6
7 title = '4.3bc'
8
9 a = 1.4
10 b = 0.3
11
12 cut_tail = 100
13
14 T = 10**4
15 dt = 5*10**-3
16 t = np.arange(0,T,dt)
17
18 x = np.zeros_like(t,dtype=float)
19 y = x.copy()
20
21 x[0] = (np.random.uniform()-0.5)
22 y[0] = (np.random.uniform()-0.5)
23
24 for t in range(len(x)-1):
25     x[t+1] = y[t]+1-a*x[t]**2
26     y[t+1] = b*x[t]
27
28 x = x[cut_tail:]
29 y = y[cut_tail:]
30
31 q_vals = np.linspace(0,2,3)
32 epsilon_range = np.linspace(10**-3, 2*10**-2, 10)
33 fig1, axs = plt.subplots(1,3,figsize=(15,15))
34 fig2, ax = plt.subplots(figsize=(7,7))
35 color = ['tab:blue', 'tab:green', 'tab:red']
36
37 for q_i in range(len(q_vals)):
38     q = q_vals[q_i]
39     Iq = np.zeros_like(epsilon_range, dtype=float)
40
41     for epsilon_i in range(len(epsilon_range)):
42         epsilon = epsilon_range[epsilon_i]
43         N_points = len(x)
44
45         xmax = 1.3
46         xmin = -xmax
47         ymax = 0.4
48         ymin = -ymax
49
50         x_bins = np.linspace(xmin, xmax, int((xmax-xmin)/epsilon))
51         y_bins = np.linspace(ymin, ymax, int((ymax-ymin)/epsilon))
52
53         plt.figure()
54         histogram = plt.hist2d(x, y, bins=[x_bins, y_bins])
55         boxes = histogram[0].copy()
56         plt.figure().clear()
57         plt.close()
58         plt.cla()
59         plt.clf()
60
61         for i in range(len(boxes[:,0])):
62             for j in range(len(boxes[0,:])):
63                 if boxes[i,j] != 0:
64                     if q == 1:
65                         N_k = boxes[i,j]
66                         Iq[epsilon_i] += ((N_k/N_points)*np.log(1/(N_k/N_points)))
67                     else:
68                         N_k = boxes[i,j]
69                         Iq[epsilon_i] += (N_k/N_points)**q
70
71     x_axis = np.log(1/epsilon_range)
72     axs[q_i].set_xlabel(r'$\ln(1/\epsilon)$')
73     ax.set_xlabel(r'$\ln(1/\epsilon)$')
74     ax.set_ylabel(r'$\ln(1/\epsilon) D_{\{q\}}$')
75
76
77
78

```

```

79     if q == 1:
80         y_axis = Iq
81         axs[q_i].set_ylabel(r'\sum_{k}^{N_{box}} p_{k} \ln(1/p_{k})$')
82     else:
83         y_axis = np.log(Iq)/(1-q)
84         axs[q_i].set_ylabel(r'$\ln(\sum_{k}^{N_{box}} p_{k}^{q})/(1-q)$')
85
86     coef = np.polyfit(x_axis, y_axis, 1)
87     print(coef)
88     Dq = coef[0]
89
90     axs[q_i].plot(x_axis, y_axis, color=color[q_i])
91     axs[q_i].set_title('$q={}$, $D_{\{}}={}$'.format(int(q),int(q),np.round(Dq,2)))
92     axs[q_i].set_box_aspect(1)
93     axs[q_i].set_ylim([5,11])
94     axs[q_i].set_xlim([np.log(1/epsilon_range[-1]), np.log(1/epsilon_range[0])])
95
96     legend = '$q={}$, $D_{\{}}$'.format(int(q),int(q)) + r'\approx' +
97     '{}$'.format(np.round(Dq,2))
98     ax.plot(x_axis, y_axis, 'o-', color=color[q_i], label=legend)
99     ax.set_title(title)
100    ax.set_box_aspect(1)
101    ax.set_ylim([5,11])
102    ax.set_xlim([np.log(1/epsilon_range[-1]), np.log(1/epsilon_range[0])])
103 plt.subplots_adjust(wspace=1)
104 ax.legend(loc='lower right')
105 fig1.savefig('Dynamical Systems/DS HW4/4.3/'+title+'_v1.png')
106 fig2.savefig('Dynamical Systems/DS HW4/4.3/'+title+'_v2.png')
107 plt.show()

```

```

1 # Exercise 4.3d
2
3 import numpy as np
4 import matplotlib.pyplot as plt
5 import sys
6
7 title = '4.3d'
8
9 a = 1.4
10 b = 0.3
11
12 cut_tail = 100
13
14 T = 10**4
15 dt = 5*10**-3
16 t = np.arange(0,T,dt)
17
18 x = np.zeros_like(t,dtype=float)
19 y = x.copy()
20
21 x[0] = (np.random.uniform()-0.5)
22 y[0] = (np.random.uniform()-0.5)
23
24 for t in range(len(x)-1):
25     x[t+1] = y[t]+1-a*x[t]**2
26     y[t+1] = b*x[t]
27
28 x = x[cut_tail:]
29 y = y[cut_tail:]
30
31 q_vals = np.linspace(0,4,10)
32 Dq_list = []
33 epsilon_range = np.linspace(10**-3, 2*10**-2, 10)
34 fig, ax = plt.subplots(figsize=(7,7))
35 color = ['tab:blue', 'tab:green', 'tab:red']
36
37 for q_i in range(len(q_vals)):
38
39     q = q_vals[q_i]
40     Iq = np.zeros_like(epsilon_range, dtype=float)
41
42     for epsilon_i in range(len(epsilon_range)):
43
44         epsilon = epsilon_range[epsilon_i]
45         N_points = len(x)
46
47         xmax = 1.3
48         xmin = -xmax
49         ymax = 0.4
50         ymin = -ymax
51
52         x_bins = np.linspace(xmin, xmax, int((xmax-xmin)/epsilon))
53         y_bins = np.linspace(ymin, ymax, int((ymax-ymin)/epsilon))
54
55         plt.figure()
56         histogram = plt.hist2d(x, y, bins=[x_bins, y_bins])
57         boxes = histogram[0].copy()
58         plt.figure().clear()
59         plt.close()
60         plt.cla()
61         plt.clf()
62
63         for i in range(len(boxes[:,0])):
64             for j in range(len(boxes[0,:])):
65                 if boxes[i,j] != 0:
66                     if q == 1:
67                         N_k = boxes[i,j]
68                         Iq[epsilon_i] += ((N_k/N_points)*np.log(1/(N_k/N_points)))
69                     else:
70                         N_k = boxes[i,j]
71                         Iq[epsilon_i] += (N_k/N_points)**q
72
73     x_axis = np.log(1/epsilon_range)
74
75     if q == 1:
76         y_axis = Iq
77     else:
78         y_axis = np.log(Iq)/(1-q)

```

```
79
80     coef = np.polyfit(x_axis, y_axis, 1)
81     Dq = coef[0]
82     Dq_list.append(Dq)
83     print('Dq=', Dq, '    q=', q)
84
85 ax.plot(q_vals, Dq_list, 'o--', color='black')
86 ax.set_title(title)
87 ax.set_box_aspect(1)
88 ax.set_xlabel(r'$q$')
89 ax.set_ylabel(r'$D_{q}$')
90
91 fig.savefig('Dynamical Systems/DS HW4/4.3/'+title+'.png')
92 plt.show()
```



```

1 # Exercise 4.3ef
2
3 import numpy as np
4 import matplotlib.pyplot as plt
5 from scipy.linalg import qr
6 import sys
7
8 title = '4.3e'
9
10 a = 1.4
11 b = 0.3
12
13 cut_tail = 100
14
15 T = 2*10**5
16 t_array = np.arange(0,T,1)
17
18 x = np.zeros_like(t_array, dtype=float)
19 y = x.copy()
20
21 x[0] = (np.random.uniform()-0.5)
22 y[0] = (np.random.uniform()-0.5)
23
24 for t in range(len(x)-1):
25     x[t+1] = y[t]+1-a*x[t]**2
26     y[t+1] = b*x[t]
27
28 x = x[cut_tail:]
29 y = y[cut_tail:]
30
31 I = np.identity(2)
32 Q = I.copy()
33 M = I.copy()
34
35 lambda_one = np.zeros_like(x)
36 lambda_two = lambda_one.copy()
37
38 new_T = len(x)
39 new_t_array = t_array[cut_tail:]-cut_tail
40
41 for t in range(len(x)):
42
43     J11 = 2*a*x[t]
44     J12 = 1
45     J21 = b
46     J22 = 0
47
48     J = np.array([[J11,J12],\
49                  [J21,J22]])
50
51     M = J
52     Q,R = qr(np.matmul(M,Q))
53
54     lambda_one[t] = lambda_one[t-1] + np.log(np.absolute(R[0,0]))/new_T
55     lambda_two[t] = lambda_two[t-1] + np.log(np.absolute(R[1,1]))/new_T
56
57 lambda_one_converged = lambda_one[-1]
58 lambda_two_converged = lambda_two[-1]
59
60 if lambda_one_converged < lambda_two_converged:
61     temp = lambda_one_converged
62     lambda_one_converged = lambda_two_converged
63     lambda_two_converged = temp
64
65 lyapunov_dimension = 1 + lambda_one_converged/np.abs(lambda_two_converged)
66
67 print('Lyapunov exponent 1: ', lambda_one_converged)
68 print('Lyapunov exponent 2: ', lambda_two_converged)
69 print('Lyapunov dimension: ', lyapunov_dimension)
70
71 fig2, ax2 = plt.subplots(figsize=(7,7))
72 ax2.plot(new_t_array[1:], np.divide(lambda_one[1:],new_t_array[1:]), '-',
73         linewidth=1.5, label=r'$\lambda_1$ (conv. at $\approx$' +
74         '{}'.format(np.round(lambda_one_converged,2)))
75 ax2.plot(new_t_array[1:], np.divide(lambda_two[1:],new_t_array[1:]), '-',
76         linewidth=1.5, label=r'$\lambda_2$ (conv. at $\approx$' +
77         '{}'.format(np.round(lambda_two_converged,2)))
78 ax2.set_xscale('log')

```

```
75 ax2.set_xlabel('$t$')
76 ax2.set_ylabel('$\sum \lambda_i$ /t')
77 ax2.set_box_aspect(1)
78 ax2.set_title(title)
79
80 plt.legend(loc="lower right", prop={'size': 10})
81 plt.savefig('Dynamical Systems/DS HW4/4.3/'+title+'.png')
82 plt.show()
```