```
1  ##############################################################################
2  # IMPORTS
3  ##############################################################################
4  import numpy as np
5  import os
6  import matplotlib.pyplot as plt
7  import matplotlib as mpl
8  ##############################################################################
9  # Variable declaration
10 ##############################################################################
11 isPlotReqd = True
12 isPlotPdf = True
13 ##############################################################################
14 # Settings for plot
15 ##############################################################################
16 if (True == isPlotReqd):
17     if (True == isPlotPdf):
18         mpl.use('pdf')
19         fig_width  = 3.487
20         fig_height = fig_width / 1.618
21         rcParams = {
22             'font.family': 'serif',
23             'font.serif': 'Times New Roman',
24             'text.usetex': True,
25             'xtick.labelsize': 8,
26             'ytick.labelsize': 8,
27             'axes.labelsize': 8,
28             'legend.fontsize': 8,
29             'figure.figsize': [fig_width, fig_height]
30             }
31         plt.rcParams.update(rcParams)
32 ##############################################################################
33 # PARAMETERS
34 ##############################################################################
35 T = 1.
36 Q = 1.
37 t_0 = 0
38 t_N = 50
39 t = np.linspace(t_0, t_N, num=t_N+1)
40 N = len(t)
41 num_run = 100
42 x_0 = 0.
43 np.random.seed(1)
44 ##############################################################################
45 # PROCESS NOISE
46 ##############################################################################
47 w_k = np.sqrt(Q) * np.random.normal(0, 1, (N-1, num_run)) # We dont have any process noise at t=0
48 w_k = np.vstack((np.zeros((1, num_run)), w_k))
49 ##############################################################################
50 # SYSTEM STATES
51 ##############################################################################
52 x_k = np.cumsum(w_k, axis=0)
53 ##############################################################################
54 # Plot of error curves
55 ##############################################################################
56 if (True == isPlotReqd):
57     ##########################################################################
58     # Configure axis and grid
59     ##########################################################################
60     fig = plt.figure()
61     ax = fig.add_subplot(111)
```

```
62      fig.subplots_adjust(left=.15, bottom=.16, right=.99, top=.97)
63
64      ax.set_axisbelow(True)
65      ax.minorticks_on()
66      ax.grid(which='major', linestyle='-', linewidth='0.5')
67      ax.grid(which='minor', linestyle=':', linewidth='0.5')
68      for run in range(num_run):
69          ax.plot(t, x_k[:, run])
70
71      ax.set_xlabel(r'$t_k$', fontsize=8)
72      ax.set_ylabel(r'$x_k$', fontsize=8)
73
74      # plt.legend()
75      if (True == isPlotPdf):
76          if not os.path.exists('./generatedPlots'):
77              os.makedirs('generatedPlots')
78          fig.savefig('./generatedPlots/q2_d.pdf')
79      else:
80          plt.show()
81  ##############################################################################
    #
82  # ENSEMBLE standard deviation
83  ##############################################################################
    #
84  print('Ensemble standard deviation at t=5 is ', np.std(x_k[5, :]))
85  print('Ensemble standard deviation at t=25 is ', np.std(x_k[25, :]))
86  print('Ensemble standard deviation at t=50 is ', np.std(x_k[50, :]))
87  ##############################################################################
    #
88  # (e) COVARIANCE ANALYSIS
89  ##############################################################################
    #
90  P = np.ones((N-1, 1))
91  P = np.vstack((np.zeros((1,1)), P))
92  P_k = np.cumsum(P)
93  sigma_k = np.sqrt(P_k)
94  ##############################################################################
95  # Plot of error curves
96  ##############################################################################
97  if (True == isPlotReqd):
98      ##########################################################################
99      # Configure axis and grid
100     ##########################################################################
101     fig = plt.figure()
102     ax = fig.add_subplot(111)
103     fig.subplots_adjust(left=.15, bottom=.16, right=.99, top=.97)
104
105     ax.set_axisbelow(True)
106     ax.minorticks_on()
107     ax.grid(which='major', linestyle='-', linewidth='0.5')
108     ax.grid(which='minor', linestyle=':', linewidth='0.5')
109     for run in range(num_run):
110         ax.plot(t, x_k[:, run])
111     ax.plot(t, sigma_k, color='k', label='$\pm \sigma_k$')
112     ax.plot(t, -sigma_k, color='k')
113     ax.set_xlabel(r'$t_k$', fontsize=8)
114     ax.set_ylabel(r'$x_k$', fontsize=8)
115
116     plt.legend()
117     if (True == isPlotPdf):
118         if not os.path.exists('./generatedPlots'):
```

```
119             os.makedirs('generatedPlots')
120         fig.savefig('./generatedPlots/q 2_e.pdf')
121     else:
122         plt.show()
123
124 print('Covariance analysis: standard deviation at t=5 is ', sigma_k[5])
125 print('Covariance analysis: standard deviation at t=10 is ', sigma_k[10])
```