

```

1 # Exercise 2.1 (a) and (b).
2
3 import numpy as np
4 import matplotlib.pyplot as plt
5
6 left_index = 0
7 middle_index = 1
8 right_index = 2
9
10 # Exercise (a),  $E = 2*k*T$ 
11 prob_left = 1 / (2 + np.exp(-2))
12 prob_right = prob_left
13 prob_middle = np.exp(-2) / (2 + np.exp(-2))
14
15 # Exercise (b),  $E \neq 2*k*T$ 
16 E = 10
17 k = 1
18 T = 200
19 prob_left = 1 / (2 + np.exp(-E/(k*T)))
20 prob_right = prob_left
21 prob_middle = np.exp(-E/(k*T)) / (2 + np.exp(-E/(k*T)))
22
23 no_steps = np.power(10,5) - 1
24 iterations = np.linspace(1,no_steps+1,no_steps+1)
25
26 position = "left"
27
28 position_counter = np.array([1,0,0]) # Left, middle, right
29 transition_counter = np.array([0,0,0]) # Left, middle, right
30
31 position_index = 0
32 transition_index = 1
33
34 left_stat = np.zeros((no_steps+1, 2))
35 right_stat = np.zeros((no_steps+1, 2))
36 middle_stat = np.zeros((no_steps+1, 2))
37 left_stat[0, position_index] = 1
38
39 for i in range(no_steps):
40
41     # MC simulating new position for the unit
42     rand = np.random.uniform()
43     if rand < prob_left:
44         new_position = "left"
45     elif rand < (prob_left + prob_right):
46         new_position = "right"
47     elif rand < (prob_middle + prob_left + prob_right):
48         new_position = "middle"
49
50     # Assigning the unit to the new position if the conditon allows
51     if new_position == "left":
52         if position == "middle" or position == "left":
53             if position == "middle":
54                 transition_counter[left_index] += 1
55             position = new_position
56             position_counter[left_index] += 1
57     else:
58         position = "right"
59         position_counter[right_index] += 1

```

```

60
61     elif new_position == "right":
62         if position == "middle" or position == "right":
63             if position == "middle":
64                 transition_counter[right_index] += 1
65                 position = new_position
66                 position_counter[right_index] += 1
67             else:
68                 position = "left"
69                 position_counter[left_index] += 1
70
71     elif new_position == "middle":
72         if position != "middle":
73             transition_counter[middle_index] += 1
74             position = "middle"
75             position_counter[middle_index] += 1
76
77     # Stats for position frequency
78     no_left = position_counter[left_index]
79     left_stat[i + 1, position_index] = no_left / (i + 2)
80
81     no_right = position_counter[right_index]
82     right_stat[i + 1, position_index] = no_right / (i + 2)
83
84     no_middle = position_counter[middle_index]
85     middle_stat[i + 1, position_index] = no_middle / (i + 2)
86
87     # Stats for transition frequency
88     no_left = transition_counter[left_index]
89     left_stat[i + 1, transition_index] = no_left / (i + 2)
90
91     no_right = transition_counter[right_index]
92     right_stat[i + 1, transition_index] = no_right / (i + 2)
93
94     no_middle = transition_counter[middle_index]
95     middle_stat[i + 1, transition_index] = no_middle / (i + 2)
96
97
98 position_distribution = position_counter / (no_steps + 1)
99 transition_distribution = transition_counter / (no_steps + 1)
100
101 # Output distributions
102 print(f'Probability distribution of positions: {position_distribution}')
103 print(f'Probability distribution of transition: {transition_distribution}')
104
105 fig, axs = plt.subplots(1,2)
106
107 # Plotting position frequency
108 axs[0].plot(iterations, left_stat[:,position_index], 'o', markersize=2)
109 axs[0].plot(iterations, right_stat[:,position_index], 'o', markersize=2)
110 axs[0].plot(iterations, middle_stat[:,position_index], 'o', markersize=2)
111 axs[0].set_title('Position frequency')
112 axs[0].set_xlabel('Time step')
113 axs[0].set_ylabel('Position frequency (no. positions/time step)')
114 axs[0].legend(['Left', 'Right', 'Middle'])
115 axs[0].set_box_aspect(1)
116
117 axs[1].plot(iterations, left_stat[:,transition_index], 'o', markersize=2)
118 axs[1].plot(iterations, right_stat[:,transition_index], 'o', markersize=2)
119 axs[1].plot(iterations, middle_stat[:,transition_index], 'o', markersize=2)

```

```
120 axs[1].set_title('Transition frequency')
121 axs[1].set_xlabel('Time step')
122 axs[1].set_ylabel('Transition frequency (no. transitions/time step)')
123 axs[1].legend(['Left', 'Right', 'Middle'])
124 axs[1].set_box_aspect(1)
125
126 fig.suptitle('2.1a:  $E = 2kT$ ', fontsize=16)
127 plt.show()
```