

---

# STOCHASTIC PROCESSES

---

## HW05

**Hanie Hatami(99100614)**

Stochastic processes course

# Contents

|          |                               |          |
|----------|-------------------------------|----------|
| <b>1</b> | <b>Abstract</b>               | <b>3</b> |
| <b>2</b> | <b>Results</b>                | <b>3</b> |
| 2.1      | Binning: . . . . .            | 3        |
| 2.1.1    | $D^{(1)}$ . . . . .           | 3        |
| 2.1.2    | $D^{(2)}$ . . . . .           | 4        |
| 2.2      | Interaction Matrix: . . . . . | 5        |
| 2.2.1    | $D^{(1)}$ . . . . .           | 5        |
| 2.2.2    | $D^{(2)}$ . . . . .           | 6        |
| <b>3</b> | <b>Conclusion</b>             | <b>8</b> |
| <b>4</b> | <b>Code</b>                   | <b>9</b> |

# 1 Abstract

In this exercise, we want to generate the data that we used in the previous exercise and get the Kramer's coefficients, this time we will get the coefficients again and make a small comparison.

## 2 Results

### 2.1 Binning:

#### 2.1.1 $D^{(1)}$

Calculated data:

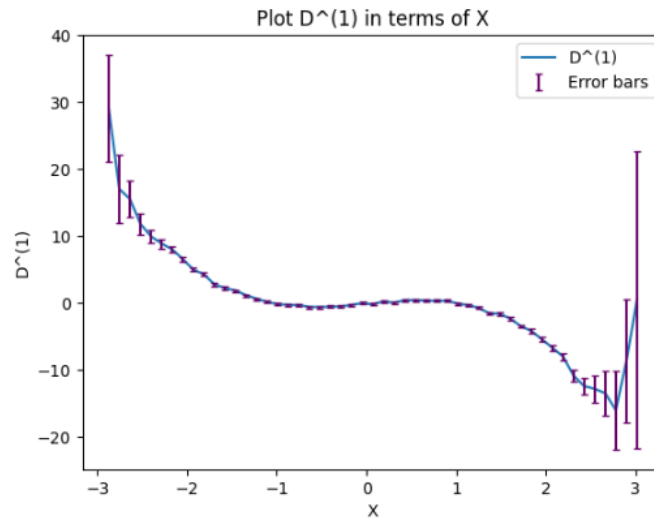


Figure 1:  $D^{(1)}$  in terms of  $X$

Given data(previous exercise):

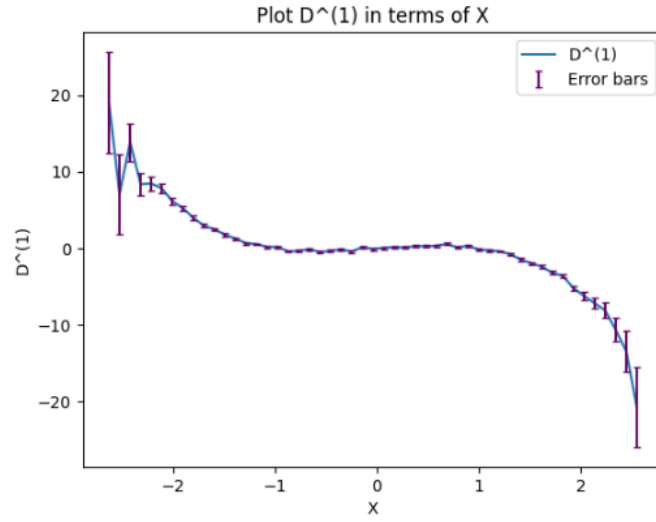


Figure 2:  $D^{(1)}$  in terms of  $X$

### 2.1.2 $D^{(2)}$

Calculated data:

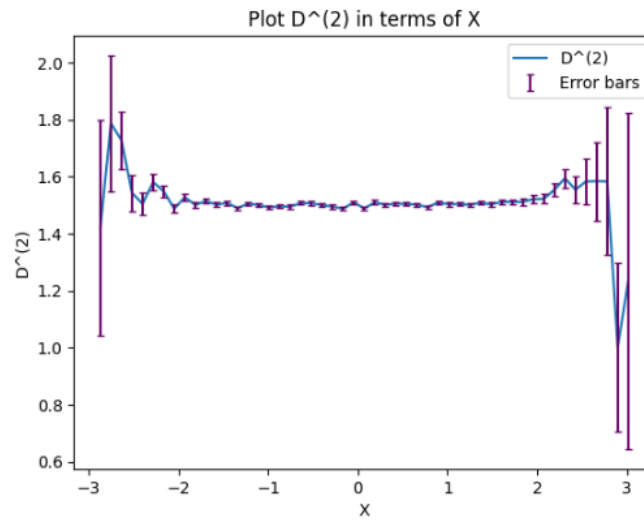


Figure 3:  $D^{(2)}$  in terms of  $X$

Given data(previous exercise):

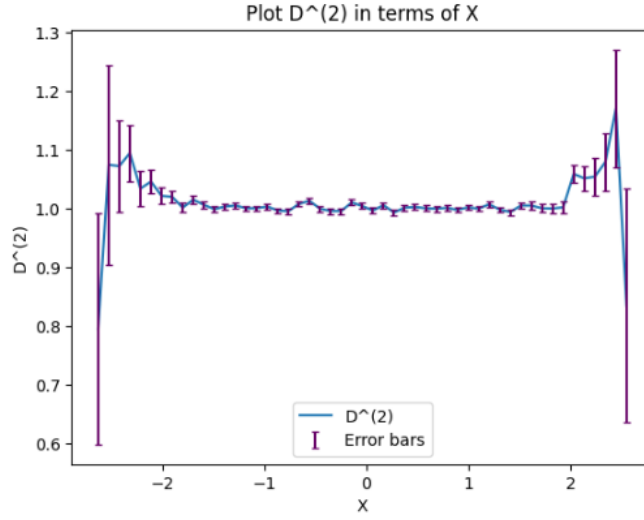


Figure 4:  $D^{(2)}$  in terms of  $X$

## 2.2 Interaction Matrix:

### 2.2.1 $D^{(1)}$

Calculated data:

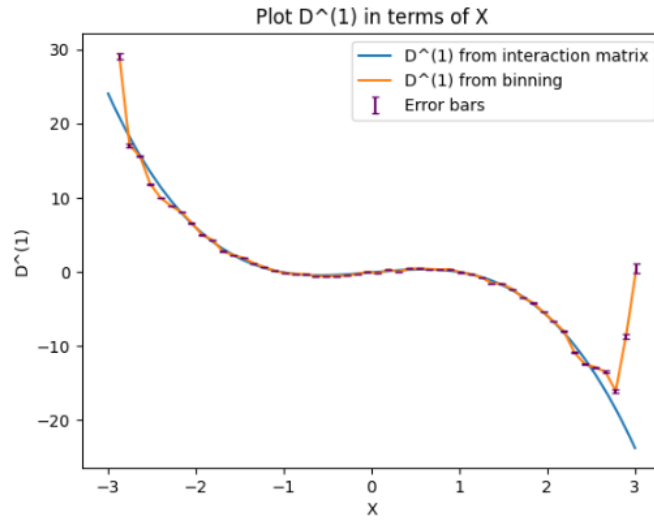


Figure 5:  $D^{(1)}$  in terms of  $X$

Given data(previous exercise):

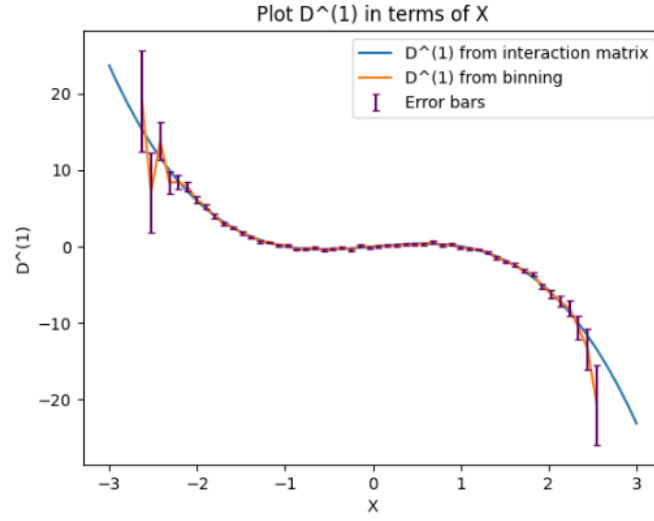


Figure 6:  $D^{(1)}$  in terms of  $X$

### 2.2.2 $D^{(2)}$

Calculated data:

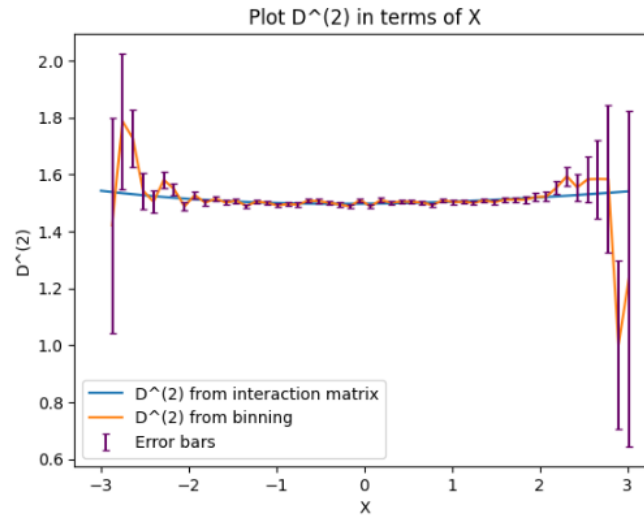


Figure 7:  $D^{(2)}$  in terms of  $X$

Given data(previous exercise):

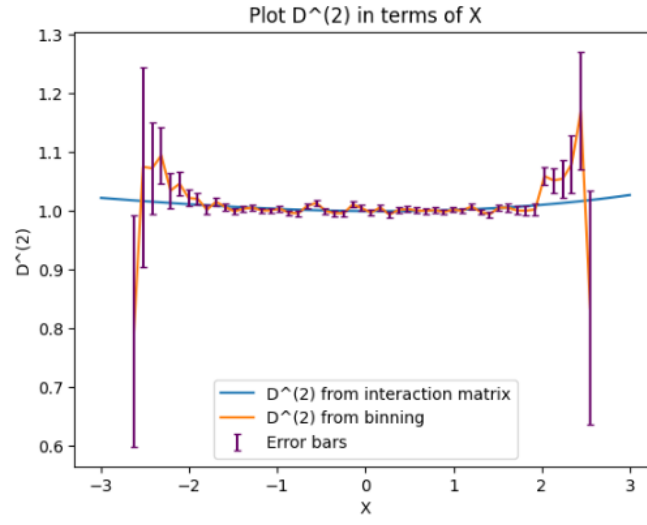


Figure 8:  $D^{(2)}$  in terms of  $X$

### 3 Conclusion

As it is clear, the generated data results are very close to the data we received. (The formula used to produce the same data has only slightly changed the noise.)

As a result, the explanation of this section is the same as the previous exercise.



## 4 Code

```
1 # Import needed libraries
2 import pandas as pd
3 import numpy as np
4 import matplotlib.pyplot as plt
5
6 #Generationg noise
7 n = 3000000
8 etha = np.random.normal(0,1,n)
9
10 #Generating data:
11 x=0.1
12 dt= 0.001
13 data = []
14 for i in range(n):
15     dx = (x-x**3)*dt + ((3*dt)**(1/2))*etha[i]
16     x= x+dx
17     data.append(x)
18
19 #Binning
20 # Calculating drift and diffusion
21 df = data
22 bins = 51
23 hist_values, bin_edges = np.histogram(df, bins=bins)
24 max = np.max(df)
25 min = np.min(df)
26 # Calculation diffrences
27 tool = len(df)
28 differences = []
29 binmid_list = []
30 for i in range(bins):
31     binmid = (bin_edges[i] + bin_edges[i+1]) / 2
32     binmid_list.append(binmid)
33     diff = []
34     for j in range(tool):
35         if df[j]>= bin_edges[i] and df[j] < bin_edges[i+1] and j != tool-1:
36             ekhtelaf = df[j+1] - df[j]
37             diff.append(ekhtelaf)
38     differences.append(diff)
39
40 # Calculating D^(1)
41 D1_dt = []
42 errors_dt =[]
43 for j in range(bins):
44     d1 = (sum(differences[j]))/len(differences[j])
45     res = pd.Series(differences[j]).var()
46     sem = (res/len(differences[j]))**(1/2)
47     errors_dt.append(sem)
48     D1_dt.append(d1)
49
50 D1 = [i * 1000 for i in D1_dt]
51 errors = [i * 1000 for i in errors_dt]
52 plt.plot(binmid_list,D1, label='D^(1)')
53 plt.errorbar(binmid_list, D1, yerr=errors, fmt='none', color='#660066',
54             capsize=2, label='Error bars')
```

```

54 plt.legend()
55 plt.xlabel("X")
56 plt.ylabel("D^(1)")
57 plt.title('Plot D^(1) in terms of X')
58 plt.show()
59
60 # Calculating D^(2)
61 D2_dt = []
62 errors_dt = []
63 for j in range(bins):
64     double_list = np.power(differences[j], 2)
65     d1 = (sum(double_list))/len(differences[j])
66     res = pd.Series(double_list).var()
67     sem = (res/len(double_list))**(1/2)
68     errors_dt.append(sem)
69     D2_dt.append(d1)
70
71 D2 = [i * (1000/2) for i in D2_dt]
72 errors = [i * (1000/2) for i in errors_dt]
73 plt.plot(binmid_list, D2, label='D^(2)')
74 plt.errorbar(binmid_list, D2, yerr=errors, fmt='none', color='#660066',
75             capsize=2, label='Error bars')
76 plt.legend()
77 plt.xlabel("X")
78 plt.ylabel("D^(2)")
79 plt.title('Plot D^(2) in terms of X')
80 plt.show()
81
82 #Interaction matrix
83 data = np.array(data)
84 diff = []
85 for i in range(len(data)-1):
86     diff = data[i+1]-data[i]
87     diff.append(diff)
88
89
90 y1 = [map(lambda x, y: x * y**i, diff, data) for i in range(4)]
91
92 tau=1
93 y = data[tau:] - data[:-tau]
94 data1 = np.ones(len(data))
95 ys = np.zeros(4)
96 for i in range(4):
97     ys[i] = np.mean(y)
98     y *= data[:-1]
99
100
101 mmnts = np.ones(7)
102 xk = np.ones(len(data))
103 for i in range(7):
104     mmnts[i] = xk.mean()
105     xk *= data
106
107 A = np.zeros((4,4))
108 for i in range(4):
109     A[i] = np.roll(mmnts, -i)[:4]

```

```

110
111 A_inv = np.linalg.inv(A)
112 phis = A_inv@ys
113 phis = phis/0.001
114 x = np.linspace(-3,3, 10001)
115
116 plt.plot(x,np.array([phis[i]*x**i for i in range(4)]).sum(0), label='D^(1)
    from interaction matrix')
117 plt.plot(binmid_list,D1, label='D^(1) from binning')
118 plt.errorbar(binmid_list, D1, yerr=errors, fmt='none', color='#660066',
    capsize=2, label='Error bars')
119 plt.legend()
120 plt.xlabel("X")
121 plt.ylabel("D^(1)")
122 plt.title('Plot D^(1) in terms of X')
123 plt.show()
124
125 tau=1
126 y = data[tau:] - data[:-tau]
127 y = y**2
128 ys2 = np.zeros(4)
129 for i in range(4):
130     ys2[i] = np.mean(y)
131     y *= data[:-1]
132
133
134 phis2 = A_inv@ys2
135 phis2 = phis2/0.002
136 x = np.linspace(-3,3, 10001)
137
138 plt.plot(x,np.array([phis2[i]*x**i for i in range(4)]).sum(0), label='D^(2)
    from interaction matrix')
139 plt.plot(binmid_list,D2, label='D^(2) from binning')
140 plt.errorbar(binmid_list, D2, yerr=errors, fmt='none', color='#660066',
    capsize=2, label='Error bars')
141 plt.legend()
142 plt.xlabel("X")
143 plt.ylabel("D^(2)")
144 plt.title('Plot D^(2) in terms of X')
145 plt.show()

```