

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ  
УНИВЕРСИТЕТ им. ПЕТРА ВЕЛИКОГО  
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ И МЕХАНИКИ  
КАФЕДРА ПРИКЛАДНАЯ МАТЕМАТИКА

ЛАБОРАТОРНАЯ РАБОТА №1  
РАССТОЯНИЕ ФРЕШЕ  
по дисциплине: ВЫЧИСЛИТЕЛЬНЫЕ КОМПЛЕКСЫ

Студент группы 3630102/60201

Митрофанова А.Г.

Преподаватель

Баженов А.Н.

Санкт-Петербург  
2019 год

## Содержание

<b>1</b>	<b>Постановка задачи</b>	<b>3</b>
<b>2</b>	<b>Теория</b>	<b>3</b>
<b>3</b>	<b>Реализация</b>	<b>3</b>
<b>4</b>	<b>Результаты</b>	<b>3</b>
4.1	Вычисление расстояния Фреше для 2-х незамкнутых ломаных . . . . .	3
4.2	Вычисление расстояния Фреше для 2-х «звездных» множеств . . . . .	4
<b>5</b>	<b>Обсуждение</b>	<b>5</b>
<b>6</b>	<b>Литература</b>	<b>5</b>
<b>7</b>	<b>Приложение</b>	<b>5</b>
7.1	frechet.py . . . . .	5
7.2	main.py . . . . .	6

## Список иллюстраций

1	Расстояние Фреше для 2-х незамкнутых ломаных . . . . .	4
2	Расстояние Фреше для 2-х «звездных» множеств . . . . .	5

# 1 Постановка задачи

Hello, my name is ALina

## 2 Теория

Рассмотрим метрическое пространство и метрику  $(V, d)$ . Пусть  $A, B$  – две кривые в пространстве  $V$  и  $\mu : AB$  – монотонное непрерывное отображение. Тогда расстояние Фреше между кривыми  $A$  и  $B$  можно найти с помощью формулы

$$DisatnseFrechet(A, B) = \min_{\mu} \max_{a \in A} d(a, \mu(a)) \quad (1)$$

На практике часто возникает необходимость количественной оценки сходства форм областей. Стандартный подход к вычислению Фреше между кривыми – вычисление дискретного расстояния Фреше для ломаных, которые приближают исходные кривые.

Пусть  $P : [0, n] \rightarrow V$  – ломаная и  $P(i), i \in [0, n] \subset V$  – вершина ломаной.

Пусть заданы две ломаные

$$P = (u_1, \dots, u_p)$$

$$Q = (v_1, \dots, v_q)$$

Тогда сопряжение между двумя ломаными – последовательность:

$$L : (u_{a_1}, \dots, u_{b_1}), \dots, (u_{a_m}, \dots, u_{b_n}), a_1 = b_1, a_m = p, b_n = q, a_{i+1} = a_i | (a_i + 1), b_{i+1} = b_i | (b_i + 1) \quad (2)$$

Длина сопряжения:

$$\|L\| = \max_{i=1, m} d(u_{a_i}, u_{b_i}) \quad (3)$$

Дискретное расстояние Фреше

$$\delta_{dF}(P, Q) = \min \|L\| \quad (4)$$

## 3 Реализация

Реализация осуществлена на языке программирования Python в среде разработки JetBrains PyCharm. Были использованы две библиотечки:

- NumPy – для работы с числами и массивами
- matplotlib – для построения графиков

## 4 Результаты

### 4.1 Вычисление расстояния Фреше для 2-х незамкнутых ломаных

Даны две незамкнутые ломаные:

$$P = \{(0, 0), (4, 2), (6, 5), (12, 6), (15, 7), (15, 10), (18, 13)\}$$

$$Q = \{(1, 1), (2, 5), (7, 7), (8, 12), (13, 14), (15, 16)\}$$

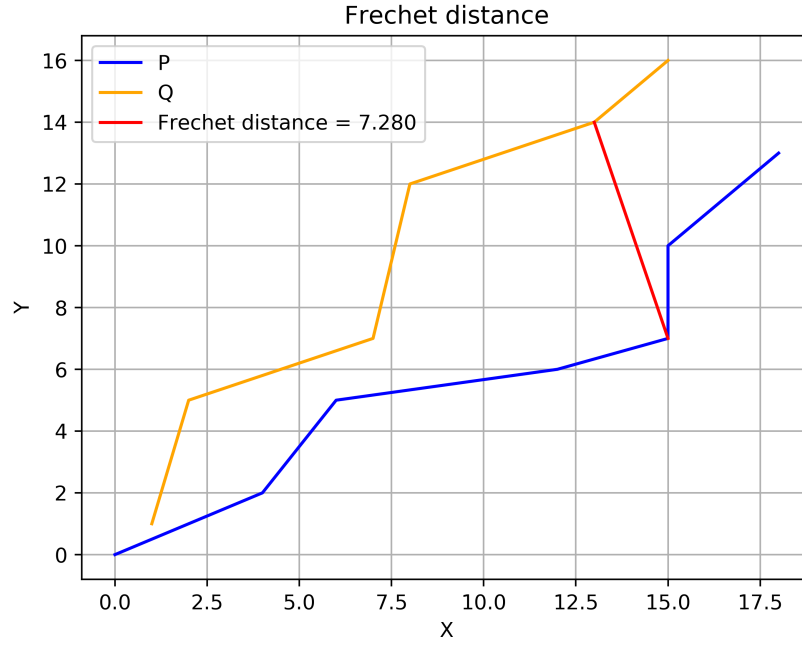


Рис. 1: Расстояние Фреше для 2-х незамкнутых ломаных

Расстояние Фреше  $\delta_{dF}(P, Q) = 7.280$  между точками  $P[4] = (15, 7)$  и  $Q[4] = (13, 14)$ .

## 4.2 Вычисление расстояния Фреше для 2-х «звездных» множеств

Даны две замкнутые пересекающиеся кривые, конкретно - «звездные» множества

$$P = \{(2, 2), (3, 4), (2, 7), (5, 6), (9, 8), (8, 5), (10, 1), (6, 3), (2, 2)\}$$

$$Q = \{(12, 1), (10, 3), (6, 6), (9, 7), (10, 9), (12, 6), (15, 5), (13, 3), (12, 1)\}$$

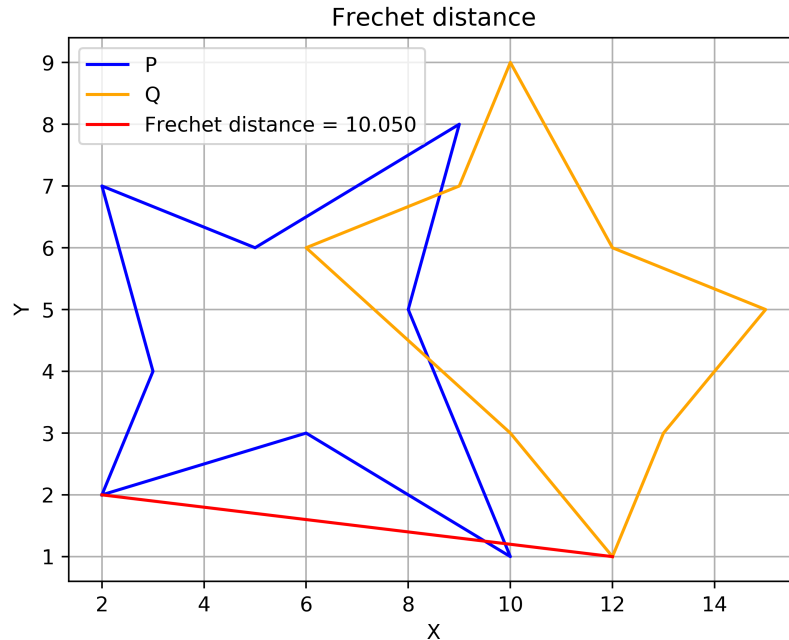


Рис. 2: Расстояние Фреше для 2-х «звездных» множеств

Расстояние Фреше  $\delta_{dF}(P, Q) = 10.050$  между точками  $P[0] = (2, 2)$  и  $Q[0] = (12, 1)$ .

## 5 Обсуждение

Результаты вычислений совпадают с ожидаемыми. Наиболее трудоемкая операция - перебор всех вариантов при вычислении

$$\max(\min(d(a_{k_{i-1}}, b_{m_j}), d(a_{k_{i-1}}, b_{m_{j-1}}), d(a_{k_i}, b_{m_{j-1}}), d(a_{k_i}, b_{m_j})))$$

## 6 Литература

### Список литературы

- [1] А.Н. Баженов, Лабораторный практикум. Методический материал. «Вычислительные комплексы» [Электронный ресурс, облачное хранилище]. Режим доступа: <https://cloud.mail.ru/public/4ra6/5wwqBzMBC/LabPractices.pdf> (дата обращения: сентябрь-декабрь, 2019 г.)

## 7 Приложение

### 7.1 frechet.py

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4
5 class Frechet:
6     def __init__(self, P, Q):
```

```

7         self.P = np.array(P)
8         self.Q = np.array(Q)
9
10        self.p = len(P)
11        self.q = len(Q)
12
13        self.ca = np.full((self.p, self.q), -1.0)
14
15
16    def frechet_distance(self):
17        dist, i, j = self.c(self.p - 1, self.q - 1)
18        self.plot(i, j, dist)
19        return dist, i, j
20
21    def c(self, i, j):
22        n_i = i
23        n_j = j
24        d = np.linalg.norm(self.P[i] - self.Q[j])
25        if self.ca[i][j] > -1:
26            return self.ca[i][j], n_i, n_j
27        elif i == 0 and j == 0:
28            self.ca[i][j] = d
29        elif i > 0 and j == 0:
30            self.ca[i][j], n_i, n_j = max(
31                self.c(i - 1, 0), (d, i, 0)
32            )
33        elif i == 0 and j > 0:
34            self.ca[i][j], n_i, n_j = max(
35                self.c(0, j - 1), (d, 0, j)
36            )
37        elif i > 0 and j > 0:
38            self.ca[i][j], n_i, n_j = max(min(
39                self.c(i - 1, j), self.c(i - 1, j - 1), self.c(i, j - 1)),
40                (d, i, j)
41            )
42        else:
43            self.ca[i][j] = float('inf')
44
45        return self.ca[i][j], n_i, n_j
46
47
48    def plot(self, i, j, d):
49        plt.figure()
50        plt.plot(self.P[:, 0], self.P[:, 1], color='blue')
51        plt.plot(self.Q[:, 0], self.Q[:, 1], color='orange')
52        plt.plot([self.P[i][0], self.Q[j][0]], [self.P[i][1], self.Q[j][1]], color='red')
53        plt.legend(['P', 'Q', 'Frechet distance = %.3f' % d])
54        plt.title('Frechet distance')
55        plt.xlabel('X')
56        plt.ylabel('Y')
57        plt.grid(True)
58        plt.savefig("Frechet_dist%d.png"%i, dpi=500, format='png')
59        plt.show()

```

## 7.2 main.py

```

1 from frechet import Frechet
2
3
4 def test_frechet_open():
5     P = [(0, 0), (4, 2), (6, 5), (12, 6), (15, 7), (15, 10), (18, 13)]
6     Q = [(1, 1), (2, 5), (7, 7), (8, 12), (13, 14), (15, 16)]
7
8     solver = Frechet(P, Q)
9     dist, i, j = solver.frechet_distance()
10
11     print("Frechet distance from P to Q is %f (%d, %d)" % (dist, i, j))
12
13
14 def test_frechet_close():

```

```

15 P = [(2, 2), (3, 4), (2, 7), (5, 6), (9, 8), (8, 5), (10, 1), (6, 3), (2, 2)]
16 Q = [(12, 1), (10, 3), (6, 6), (9, 7), (10, 9), (12, 6), (15, 5), (13, 3), (12, 1)]
17
18 solver = Frechet(P, Q)
19 dist, i, j = solver.frechet_distance()
20
21 print("Frechet distance from P to Q is %f (%d, %d)" % (dist, i, j))
22
23
24
25 if __name__ == "__main__":
26     print("Hello, World!")
27     test_frechet_open()
28     test_frechet_close()

```