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

Список иллюстраций

1 Постановка задачи

Построить ломаные кривые. Вычислить для них расстояние Фреше. Показать, на каких элементах реализуется данное расстояние.

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

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

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

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

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

Тогда сопряжение между двумя ломаными - последовательность:

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

Длина сопряжения:

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

Дискретное расстояние Фреше

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

3 Реализация

Реализация осуществлена на языке программирования Python в среде разработке JetBrains PyCharm. Были использованы две бибилотеки:

- 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:

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

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

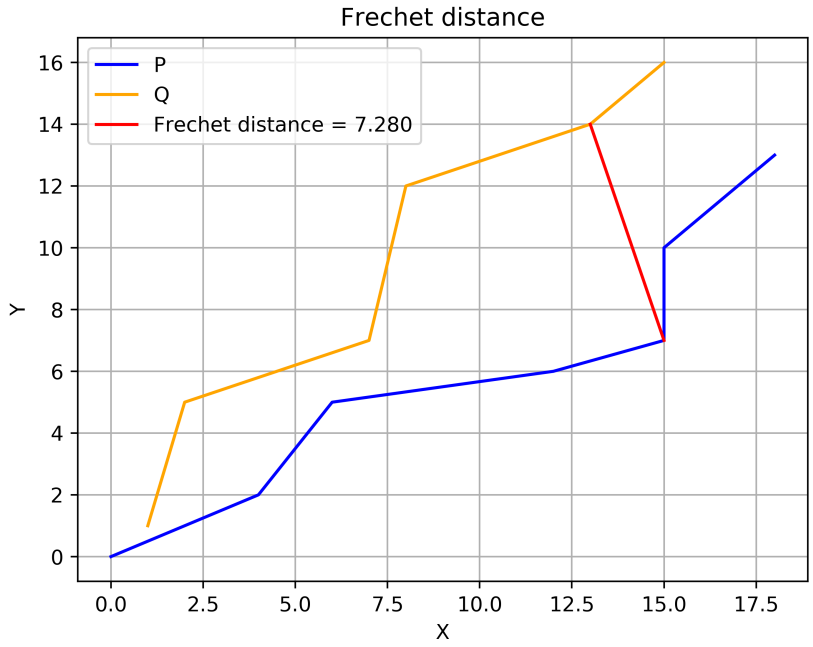

Рис. 1: Расстояние Фреше для 2-х незамкнутых ломаных

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

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

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

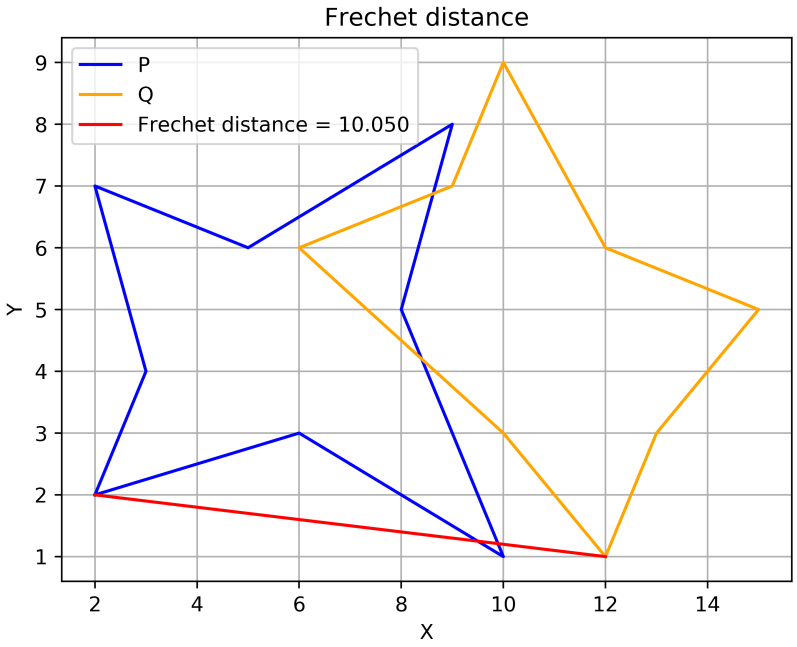

Рис. 2: Расстояние Фреше для 2-х «звездных» множеств

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/LabPractics.pdf (дата обращения: сентябрь-декабрь, 2019 г.)

7 Приложение

7.1 frechet.py

```
import numpy as np
import matplotlib.pyplot as plt


class Frechet:
    def __init__(self, P, Q):
```

```
        self.P = np.array(P)
        self.Q = np.array(Q)

        self.p = len(P)
        self.q = len(Q)

        self.ca = np.full((self.p, self.q), -1.0)


    def frechet_distance(self):
        dist, i, j = self.c(self.p - 1, self.q - 1)
        self.plot(i, j, dist)
        return dist, i, j

    def c(self, i, j):
        n_i = i
        n_j = j
        d = np.linalg.norm(self.P[i] - self.Q[j])
        if self.ca[i][j] > -1:
            return self.ca[i][j], n_i, n_j
        elif i == 0 and j == 0:
            self.ca[i][j] = d
        elif i > 0 and j == 0:
            self.ca[i][j], n_i, n_j = max(
                self.c(i - 1, 0), (d, i, 0)
            )
        elif i == 0 and j > 0:
            self.ca[i][j], n_i, n_j = max(
                self.c(0, j - 1), (d, 0, j)
            )
        elif i > 0 and j > 0:
            self.ca[i][j], n_i, n_j = max(min(
                self.c(i - 1, j), self.c(i - 1, j - 1), self.c(i, j - 1)),
                (d, i, j)
            )
        else:
            self.ca[i][j] = float('inf')

        return self.ca[i][j], n_i, n_j


    def plot(self, i, j, d):
        plt.figure()
        plt.plot(self.P[:, 0], self.P[:, 1], color='blue')
        plt.plot(self.Q[:, 0], self.Q[:, 1], color='orange')
        plt.plot([self.P[i][0], self.Q[j][0]], [self.P[i][1], self.Q[j][1]], color='red')
        plt.legend(['P', 'Q', 'Frechet distance = %.3f' % d])
        plt.title('Frechet distance')
        plt.xlabel('X')
        plt.ylabel('Y')
        plt.grid(True)
        plt.savefig("Frechet_dist%d.png"%i, dpi=500, format='png')
        plt.show()
```

7.2 main.py

```
from frechet import Frechet


def test_frechet_open():
    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)]

    solver = Frechet(P, Q)
    dist, i, j = solver.frechet_distance()

    print("Frechet distance from P to Q is %f (%d, %d)" % (dist, i, j))


def test_frechet_close():
```

```
    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)]

    solver = Frechet(P, Q)
    dist, i, j = solver.frechet_distance()

    print("Frechet distance from P to Q is %f (%d, %d)" % (dist, i, j))



if __name__ == "__main__":
    print("Hello, World!")
    test_frechet_open()
    test_frechet_close()
```