Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ Фундаментальные науки

КАФЕДРА Прикладная математика

КУРСОВАЯ РАБОТА

НА ТЕМУ:

Построение триангуляции Делоне методом заметающей прямой в системе Wolfram Mathematica

Студент ФН2-51Б (Группа) _____ (Подпись, дата) В. А. Лосев (И. О. Фамилия)

Руководитель курсовой работы _____ (Подпись, дата) А. Ю. Попов (И. О. Фамилия)

2021 г.

Оглавление

Введение 3

1. Постановка задачи. 3

2. Диаграмма Вороного и триангуляция Делоне 4

2.1. Преобразование * 4

2.2. Основные понятия 5

3. Реализация метода заметающей прямой 6

3.1. Поиск областей и точек пересечения 6

3.2. Обработка случая точки пересечения гипербол 7

3.3. Общий алгоритм метода заметающей прямой 7

3.4. Основные функции и структуры в программе 9

3.5. Обратное преобразование 10

4. Экспериментальная проверка скорости работы алгоритма 11

5. Заключение 11

Список использованных источников 12

Введение

При решении различных задач механики сплошной среды (механика деформируемого твердого тела, задачи гидро- и газодинамики и другие) применяются различные численные методы. Очень многие из широко распространенных методов предполагают построение сетки в расчетной области. При построении сетки обычно стараются обеспечить ее приемлемое качество, которое можно оценить соотношением между размерами элементов, углами и другими параметрами. Особенный интерес представляет триангуляция Делоне ввиду своих особых свойств.

Триангуляция Делоне — триангуляция для заданного множества точек S на плоскости, при которой для любого треугольника все точки из S за исключением точек, являющихся его вершинами, лежат вне окружности, описанной вокруг треугольника [1]. Приведем основные свойства и теоремы.

1. Триангуляция Делоне обладает максимальной суммой минимальных углов всех своих треугольников среди всех возможных триангуляций.
2. Триангуляция Делоне обладает минимальной суммой радиусов окружностей, описанных около треугольников, среди всех возможных триангуляций.
3. Если никакие четыре точки не лежат на одной окружности, триангуляция Делоне единственна.

Также триангуляция Делоне имеет тесную связь с диаграммой Вороного (оба объекта показаны на рисунке 1). Диаграмма Вороного конечного множества точек S на плоскости представляет такое разбиение плоскости, при котором каждая область этого разбиения образует множество точек, более близких к одному из элементов множества S, чем к любому другому элементу множества. Если соединить рёбрами точки, области Вороного которых граничат друг с другом, полученный граф будет являться триангуляцией Делоне.

1. Постановка задачи.

Существует несколько видов алгоритмов для построения диаграммы Вороного (и, соответственно, триангуляции Делоне), к ним относятся перечисленные ниже алгоритмы, с указанием их сложности:

1) инкрементальный алгоритм, сложность $O(n^2)$;

2) алгоритм «Разделяй и властвуй», сложность $O(n \log n)$;

3) метод заметающей прямой (алгоритм Форчуна), сложность $O(n \log n)$.

Последний алгоритм обладает преимуществами над другими алгоритмами, по-

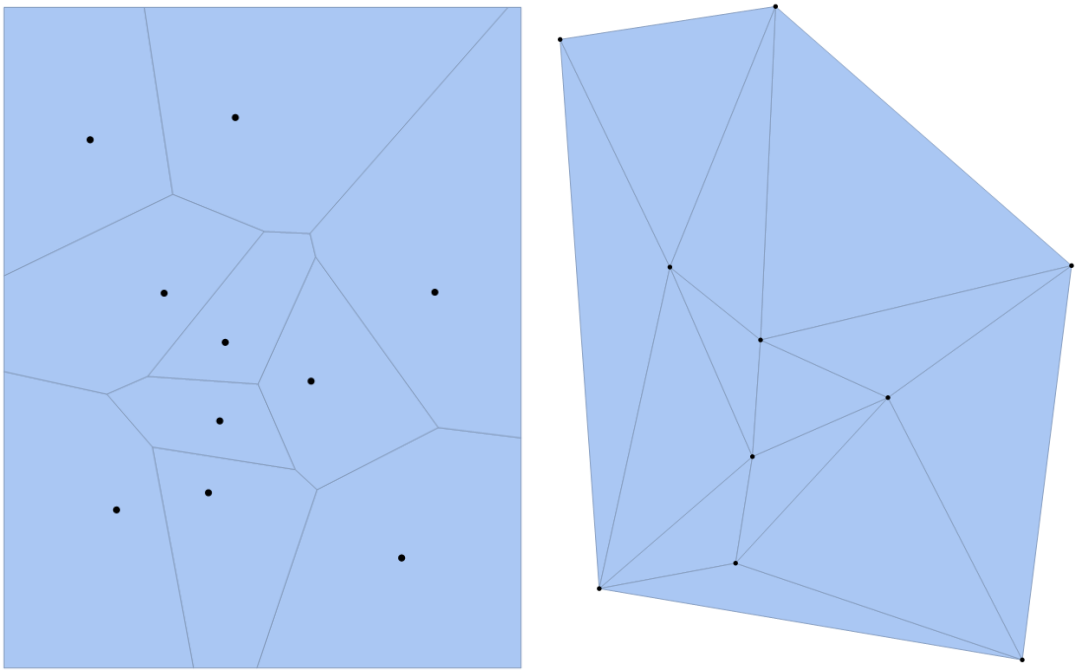

Рис. 1. Диаграмма Вороного и триангуляция Делоне

скольку его сложность $O(n \log n)$, а также он позволяет получить решение сразу, без «сшивания» отдельных частей как, например, алгоритм «Разделяй и властвуй» [2].

Целью данной работы была реализация алгоритма Форчуна для диаграммы Вороного в системе Wolfram Mathematica и получение с помощью нее триангуляции Делоне. Дополнительно требовалось экспериментально проверить скорость работы алгоритма.

2. Диаграмма Вороного и триангуляция Делоне

2.1. Преобразование *

Метод заметающей прямой предполагает ее движение снизу вверх. При этом в прямой постановке задачи получается так, что очередной многоугольник (область диаграммы Вороного) впервые пересекается этой прямой в произвольной точке, а не в той исходной точке, которая его определяет. Это усложняет работу с областью диаграммы Вороного и возникают проблемы с определением точки, относящейся к данному многоугольнику, поэтому имеет смысл применить специальное преобразование (обозначается *).

Преобразование *: $R^2 \to R^2$ определяется следующим образом: $^*(z) = (z_x, z_y +$

$d(z)$), где $d(z)$ — расстояние от точки z до ближайшей из точек множества S, и является центральным в алгоритме. Преобразование * отображает точку $z \in R^2$ в самую верхнюю точку круга Вороного в точке z. Более того, преобразование * отображает каждую вертикальную прямую в себя. Рассмотрим случай, когда l – не вертикальная линия, тогда * взаимооднозначна на l и $*_p(l)$ является гиперболой.

Пусть прямая l задана уравнением $y = mx + b$. Результат ее преобразования с помощью * относительно точки p следующий (иллюстрация приведена на рисунке 2):

$$*_p(l) = (x, z) : z = mx + b + ((mx + b - p_y)^2 + (x - p_x)^2)^{\frac{1}{2}}. \quad (1)$$

Таким образом получаем уравнение гиперболы

$$(y - mx - b)^2 = (mx + b - p_y)^2 + (x - p_x)^2$$

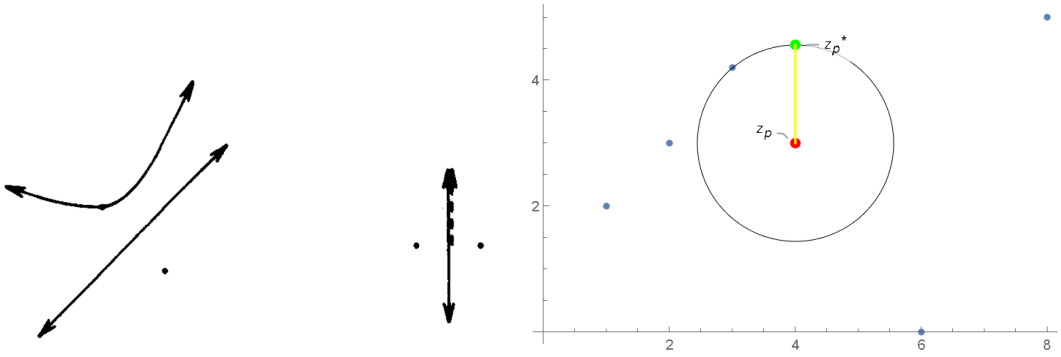

Рис. 2. Преобразование * для прямой и точки

Смысл преобразования * заключается в том, что исходная точка множества S остается на месте, а все остальные переходят выше нее с сохранением координаты x. Это приводит к тому, что в преобразованном пространстве заметающая прямая впервые пересечет область именно в исходной точке.

2.2. Основные понятия

Точки $p_1, p_2, \ldots, p_n$ – исходные точки (sites), относительно которых надо построить диаграмму Вороного и триангуляцию Делоне. В программе следует их записать в массив, отсортировав в лексикографическом порядке по возрастанию y-координаты. Серединный перпендикуляр (bisector) $B_{q,p}$ для точек p, q, где $q_y < p_y$, — прямая, которая перпендикулярна отрезку pq и проходит через его середину. Гиперболой $C_{q,p}$ назовем гиперболу, заданную уравнением (1), что является преобразованием * для серединного перпендикуляра $B_{q,p}$. Точка p, $p_y > q_y$ является вершиной соответствующей гиперболы и условно делит ее на две ветви: $C^-_{q,p}$, $C^+_{q,p}$, лежащие слева и справа

от точки p, соответственно (как показано на рисунке 3). Также в работе алгоритма используется очередь (queue) из точек, обрабатываемых программой, и список L, содержащий ветви гипербол и области пространства R_q.

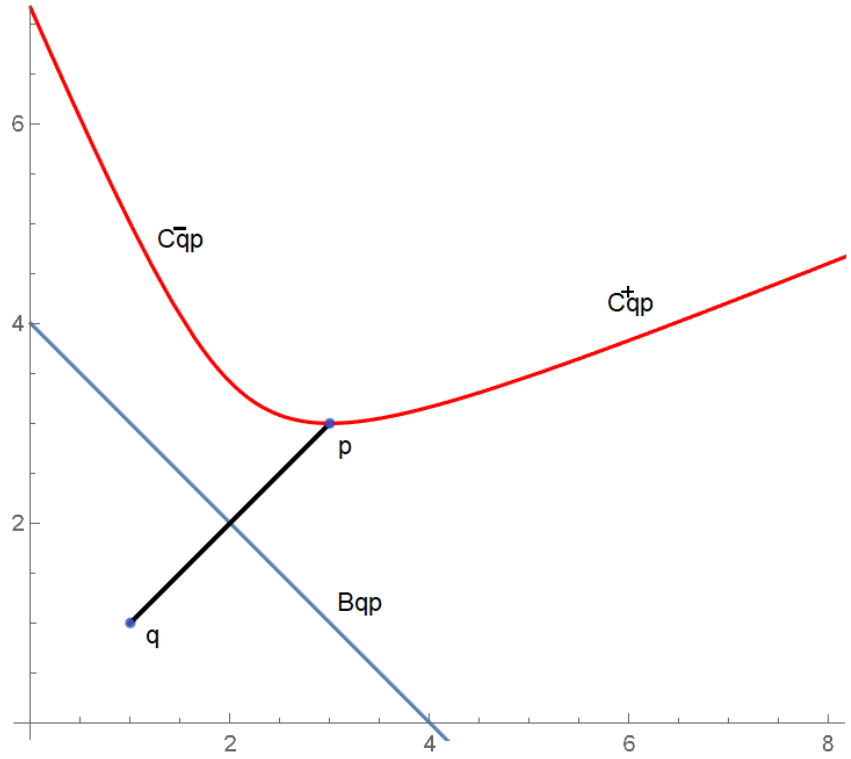

Рис. 3. Основные понятия

3. Реализация метода заметающей прямой

3.1. Поиск областей и точек пересечения

По мере движения заметающей прямой вверх обрабатываются точки исходного множества S, для каждой из них сначала следует найти области R_q, которой принадлежит эта точка. В системе Wolfram Mathematica программно это реализуется с помощью функции `Findregion[point]`, про которую будет написано ниже. В данном случае под областью понимается множество точек, лежащих между ветвями гиперболы. Первая гипербола будет построена относительно первой и второй точки. Второй индекс гиперболы – рассматриваемая точка, первый индекс – индекс точки, в области которой находится рассматриваемая точка (см. рисунок 4). В результате построенные гиперболы имеют точки пересечения друг с другом, которые будут записаны в очередь и также будут обработаны в программе.

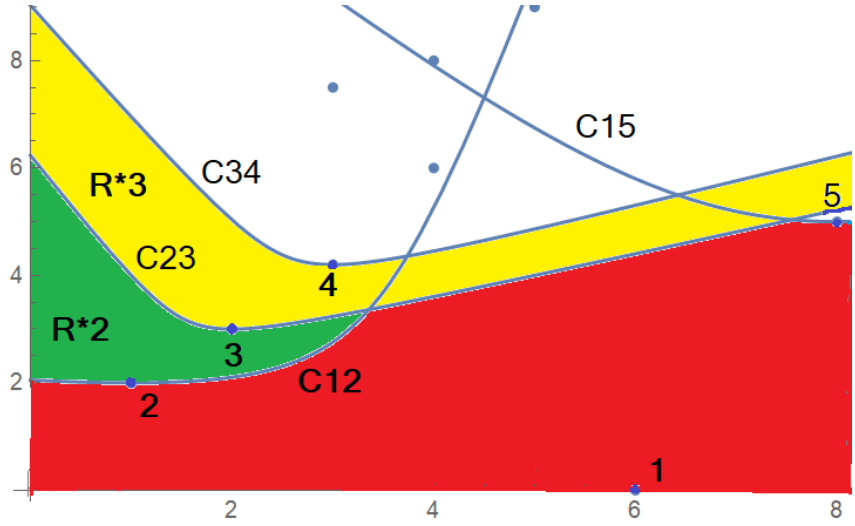

Рис. 4. Поиск областей

3.2. Обработка случая точки пересечения гипербол

Если у двух гипербол есть общий индекс ($C_{p,s}$ и $C_{s,q}$, например), то точка их пересечения является вершиной диаграммы Вороного в пространстве V^* и треугольник из точек $\{p, q, s\}$ будет добавлен в триангуляцию Делоне, то есть будет являться элементом массива `vertex` в программе, куда по результатам работы программы будут записаны вершины триангуляции Делоне для заданного множества точек. Далее строим гиперболу относительно несовпадающих индексов (рисунок 5), а две исходные гиперболы не продлеваем после точки пересечения. Если пересечение образовано ветвями разных знаков, то если $q_y < p_y$ и при этом $q_x < p_x$, оставляем у гиперболы $C_{p,q}$ только левую ветвь $C^-_{p,q}$, если $q_x > p_x$ — правую. При пересечении ветвей с одинаковыми знаками оставляем у новообразованной гиперболы ветвь того же знака. Далее продолжаем процесс нахождения точек пересечения с совпадающими индексами. Таким образом, каждая вершина диаграммы Вороного в пространстве V^* будет образована пересечением трех гипербол. С помощью этого можно получить список вершин треугольника Делоне, для этого нужно извлечь индексы пересекающихся гипербол с устранением повторений. Промежуточный результат построения приведен на рисунке 6.

3.3. Общий алгоритм метода заметающей прямой

1. Создаем массив `queue` и записываем туда отсортированные исходные точки.
2. Отбрасываем первую точку, начинаем обработку остальных элементов из `queue` до тех пор, пока список не станет пустым.

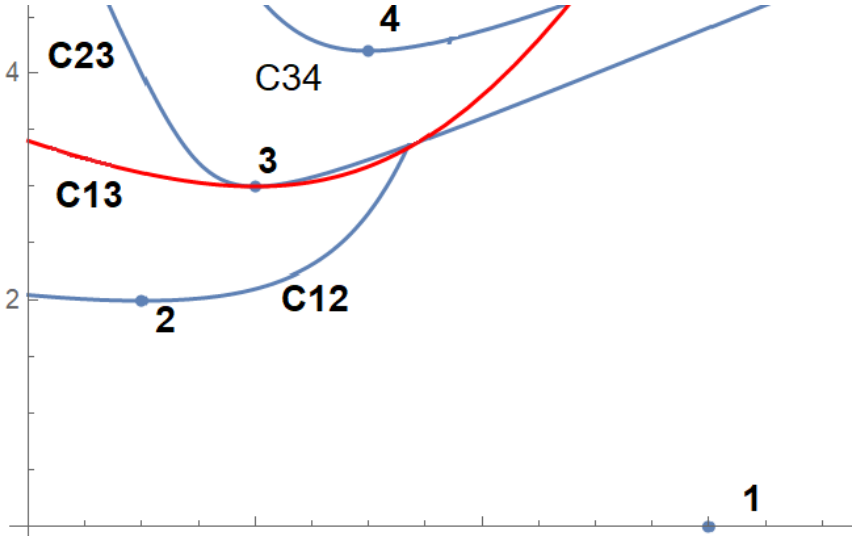

Рис. 5. Построение новой гиперболы

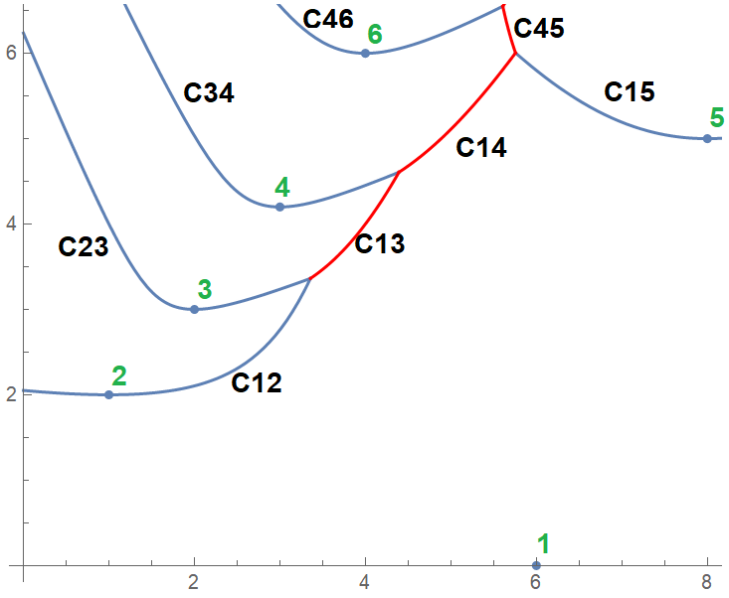

Рис. 6. Нахождение вершин диаграммы Вороного

3. Если элемент p_i из `queue` является точкой исходного множества, находим область R_q^*, к которой он относится.
4. После нахождения области строим гиперболу $C_{p,q}$.
5. Вместо R_q^* в список L записываем $R_q^*, C_{p,q}^-, R_p^*, C_{p,q}^+, R_q^*$.
6. Удаляем из `queue` пересечения между левой и правой границей R_q^*, если такие имеются.
7. Добавляем в `queue` пересечение между $C_{p,q}^-$ и его соседом в L слева и пересечение между $C_{p,q}^+$ и его соседом в L справа, если такие имеются.

8. Следующие шаги выполняются в том случае, если точка p — это пересечение $C_{q,r}$, $C_{r,s}$.
9. Строим гиперболу $C_{q,s}$.
10. Заменяем $C_{q,r}, R_r^*, C_{r,s}$ в L на $C_{q,s}^-$ или $C_{q,s}^+$ в зависимости от значений координат точек q и s и знаков у пересекающихся ветвей.
11. Удаляем из `queue` пересечения между $C_{q,r}$ и его соседом слева и между $C_{r,s}$ и его соседом справа.
12. Добавляем в `queue` пересечения $C_{q,s}$ с соседями справа и слева из L.
13. Отмечаем p — точку пересечения $C_{q,r}, C_{r,s}, C_{q,s}$ как вершину диаграммы Вороного, $\{q, r, s\}$ – точки, соответствующие треугольнику Делоне.

3.4. Основные функции и структуры в программе

При реализации программы в системе компьютерной алгебры были созданы некоторые функции, без которых реализация была бы трудно выполнима. Одними из самых главных функций являются следующие.

1) `Bisector[point1,point2]` — функция для построения гиперболы по формуле (1), соответствующей $B_{p1,p2}$. На вход передаем координаты точек, на выходе получаем уравнение нужной нам гиперболы.

2) `Findregion[point]` — функция для поиска области, содержащей данную точку, координаты которой подаем на вход. Для этого мы строим горизонтальную прямую (по сути — заметающую прямую в данный момент), соответствующую координате y данной точки, и находим пересечения с построенными гиперболами. Область можно найти путем выбора пересечения с максимальным значением координаты x из всех пересечений, образовавшихся слева от точки и пересечение с минимальным значением x, образовавшихся справа от точки.

3) `Sortfunction[sites]` — функция для сортировки точек в лексикографическом порядке по возрастанию y-координаты.

Также не обойтись и без определенных структур, центральной из которых является очередь `queue`, хранящая в себе исходные точки и, в дополнение к ним, получившиеся точки пересечения гепербол. После обработки каждого элемента этого массива, он удаляется из `queue` и программа выполняется до тех пор, пока список `queue` не пустой. Результатом работы программы станет массив `vertex`, каждый элемент которого состоит из номеров трех различных точек, которые входят в индексы пересекающихся гипербол. Каждый элемент данного массива является списком вершин одного треугольника в треангуляции Делоне.

3.5. Обратное преобразование

После построения диаграммы Вороного в пространстве V^* выполним обратное преобразование. Для этого проведем срединные перпендикуляры, соответствующие конкретным гиперболам ($C_{p,q} \to B_{p,q}$) до x-координаты вершины диаграммы Вороного в пространстве V^*, то есть до пересечения гипербол из V^* (рисунок 7). Диаграмма Вороного и триангуляция Делоне взаимно однозначны. В двумерном случае вершины Вороного соединены ребрами, которые могут быть получены из отношений смежности треугольников Делоне: если два треугольника имеют общее ребро в триангуляции Делоне, их окружности должны быть соединены с ребром в диаграмме Вороного. Таким образом, если многоугольники в диаграмме Вороного имеют общее ребро, то соединив исходные точки, соответствующие этим многоугольникам, получим триангуляцию Делоне для набора исходных точек (рисунок 8).

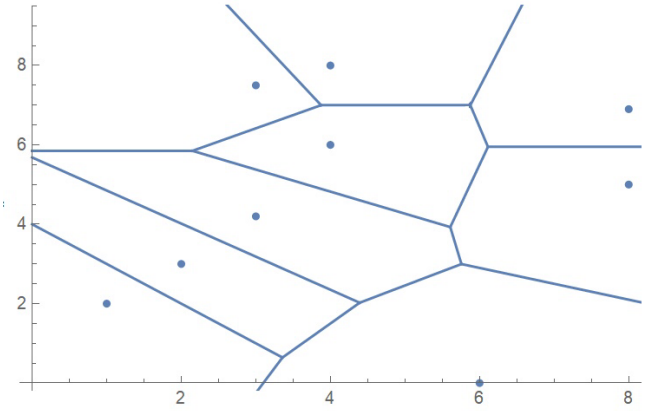

Рис. 7. Диаграмма Вороного

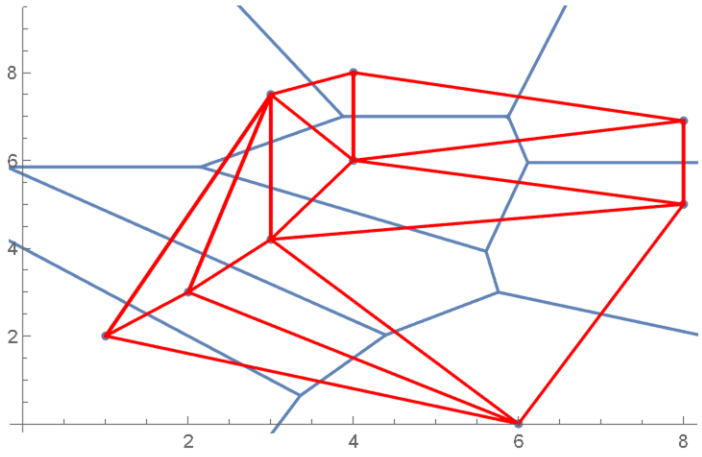

Рис. 8. Диаграмма Вороного и триангуляция Делоне

4. Экспериментальная проверка скорости работы алгоритма

Сложность алгоритма Форчуна для диаграммы Вороного и, соответственно, триангуляции Делоне — $O(n \log n)$. Проверим это экспериментально, для этого измерим время для набора задач с произвольным набором исходных данных для различного числа исходных точек и проведем интерполяцию по полученным результатам. Проведены замеры времени для набора задач с $n = 10, 20, 30, 40, 50, 100$, путем интерполяции получена зависимость $y = 0.37 - 0.041n - 0.053 \log n + 0.025n \log n$, график которой приведен на рисунке 9. По полученным результатам видим, что временная сложность алгоритма близка к $O(n \log n)$.

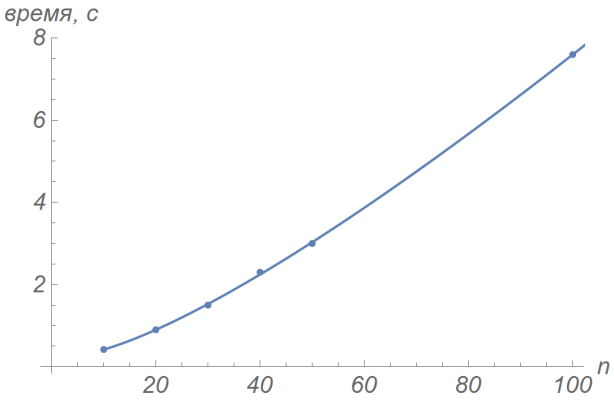

Рис. 9. Зависимость времени от входных данных

5. Заключение

Таким образом, в рамках данной работы в системе компьютерной алгебры Wolfram Mathematica был реализован метод заметающей прямой (алгоритм Форчуна) для построения триангуляции Делоне, которая в основном используется в сеточных методах решения задач. Также экспериментально проверена сложность его работы. Дополнительно реализована проверка, удовлетворяют ли треугольники условию Делоне: рассматриваются два треугольника ABD и BCD с общим ребром BD, если сумма углов α и γ (рисунок 10) меньше или равна 180°, треугольники удовлетворяют условию Делоне.

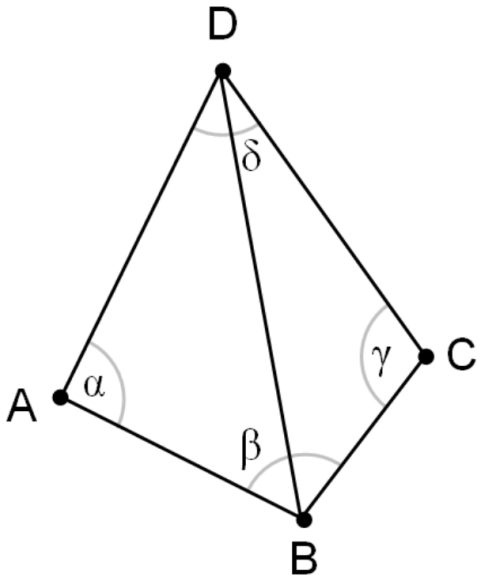

Рис. 10. Проверка триангуляции Делоне

Список использованных источников

1. Скворцов А.В. Триангуляция Делоне и её применение. Томск: Изд-во Томского университета, 2002. 128 с.
2. Wong K. An Efficient Implementation of Fortune's Plane-Sweep Algorithm for Voronoi Diagrams. URL: http://www.rigi.cs.uvic.ca/downloads/papers/pdf/cg.pdf (Дата обращения: 18.12.2021).
3. Fortune S. A sweepline algorithm for Voronoi diagrams // Algorithmica. 1987. P. 153–174.