Contents

Chapter 1

Voronoi Diagrams

1.1 Introduction

Let $\|\cdot\| : \mathbb{R}^2 \to \mathbb{R}$ be a norm. Then we define the distance function as

$$\operatorname{dist}(p, q) = \|p - q\| . \tag{1.1}$$

For $1 \leq p < \infty$ we define the L^p norm by

$$\|(x, y)\|_p = \left(|x|^p + |y|^p \right)^{1/p}, \tag{1.2}$$

and we note that $\|\cdot\|_2$ is the well-known Euclidean distance. For $p = 1$, the above reduces to

$$\|(x, y)\|_1 = |x| + |y| . \tag{1.3}$$

Letting $p \to \infty$, we also obtain the norm

$$\|(x, y)\|_\infty = \max \left(|x| , |y| \right). \tag{1.4}$$

Definition 1.1 (Voronoi diagram)**.** Let $P = \{p_1, p_2, \ldots, p_n\} \subset \mathbb{R}^2$. The cells corresponding to each point are denoted by

$$\mathcal{V}(p_i) = \{q \in \mathbb{R}^2 \mid \operatorname{dist}(q, p_i) < \operatorname{dist}(q, p_j) \text{ for all } i \neq j\}.$$

The Voronoi diagram of P, denoted $\operatorname{Vor}(P)$, is the subdivision of $\mathbb{R}^2$ consisting of the cells $\mathcal{V}(p_1), \mathcal{V}(p_2), \ldots, \mathcal{V}(p_n)$.

The following figure shows how the Voronoi diagram for 9 random points looks like with regards to some different L^p norms:

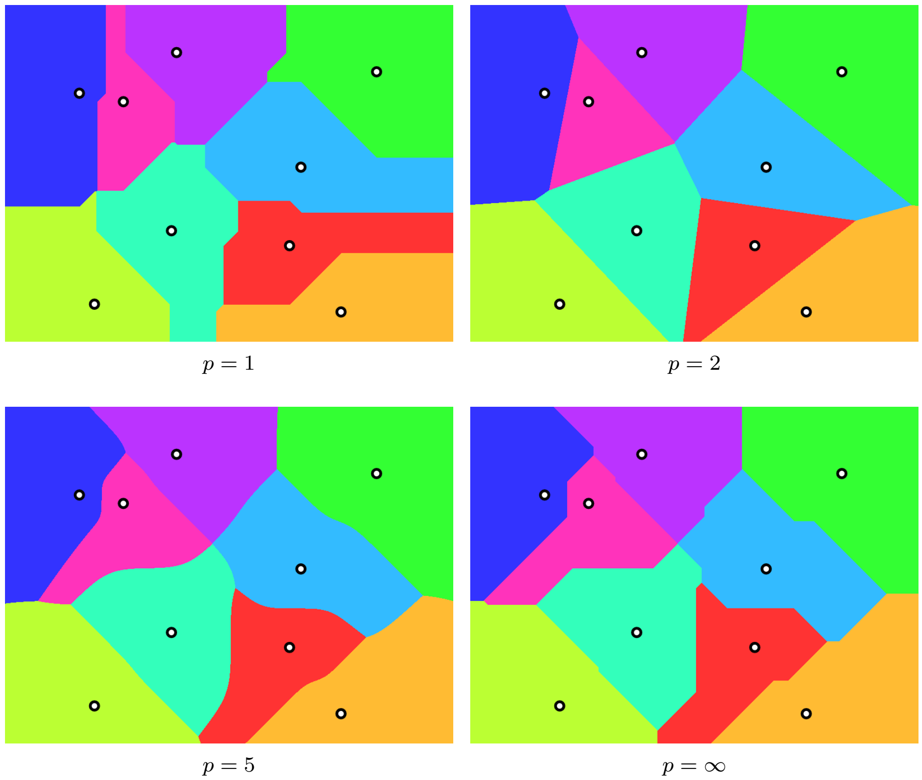


Figure 1.1: Voronoi diagrams of 9 random points using different $\|\cdot\|_p$

The above figures were generated using a very naive algorithm, which for each each pixel determinates which of the 9 points is the closest with regards to the chosen norm. A demo is available in demos/pixel-voronoi-naive.

Note that some of the cells may be unbounded, for example the bottom left green cell in the above figure. For $p = 1$ and $p = \infty$ the boundaries of the cells $\mathcal{V}(p_i)$ are characterised by lines, rays and segments that can only point in the 8 compass directions. For $p = 2$ the boundaries consist of lines, rays and segments which can point in any direction. Interestingly, for $2 < p < \infty$ it seems that the boundary consists of smooth curves that are not necessarily part of a line.

We now want to look at the graph structure of the Voronoi diagram. For $P = \{p_1, p_2, \dots, p_n\} \subset \mathbb{R}^2$ the set

$$G_P = \mathbb{R}^2 - \bigcup_{i=1}^{n} \mathcal{V}(p_i) = \{q \in \mathbb{R}^2 \mid \operatorname{dist}(q, p_i) = \operatorname{dist}(q, p_j) \text{ for some } i \neq j\}$$

turns out to be an embedding of a graph, where some of the edges are infinite, here's a visualization:

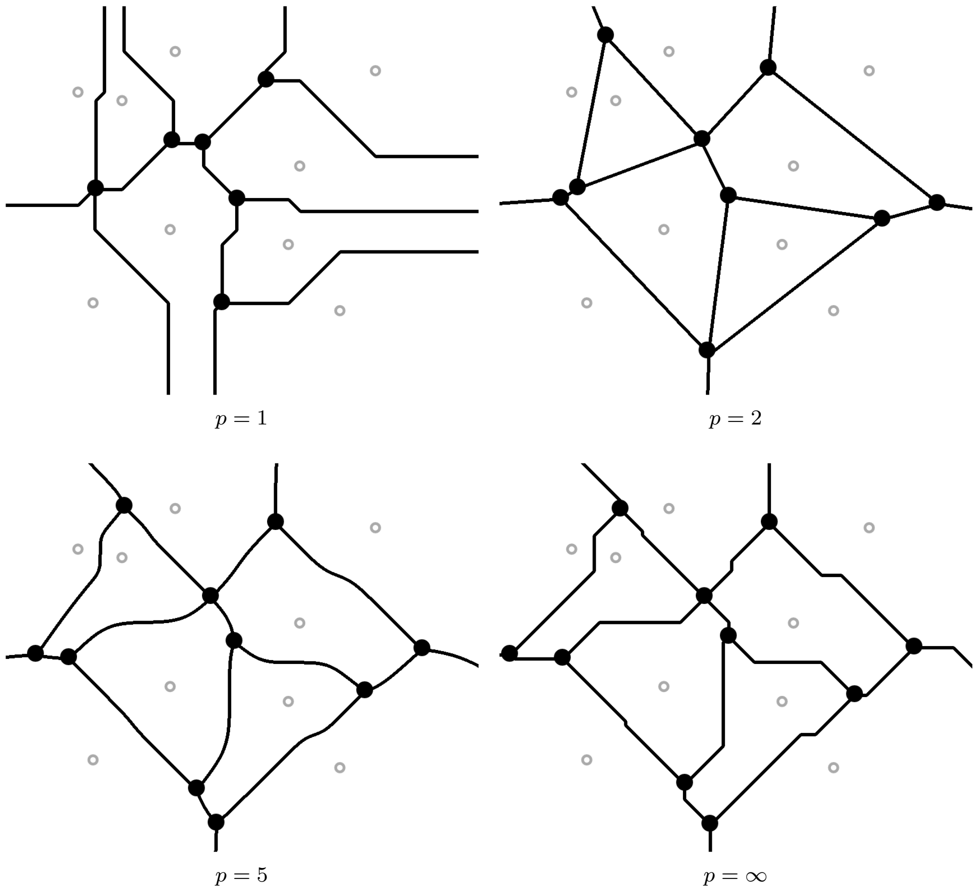


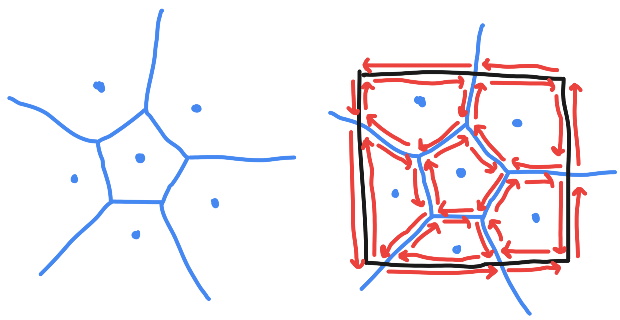
Figure 1.2: G_P of the 9 random points using different $\|\cdot\|_p$.

Note that it's the black vertices and edges which make up the graph, the gray points from P are just there for visualization. Rather than computing $\mathrm{Vor}(P)$, our algorithms will actually compute G_P, and from there be able to compute $\mathrm{Vor}(P)$. For this reason we may sometimes actually mean G_P when we write $\mathrm{Vor}(P)$, this will clear from the context.

Now, a natural question arises: how do we store Voronoi diagrams? We'll need the following geometric data structure:

Definition 1.2 (DCEL)**.** (TODO: Define the DCEL.)

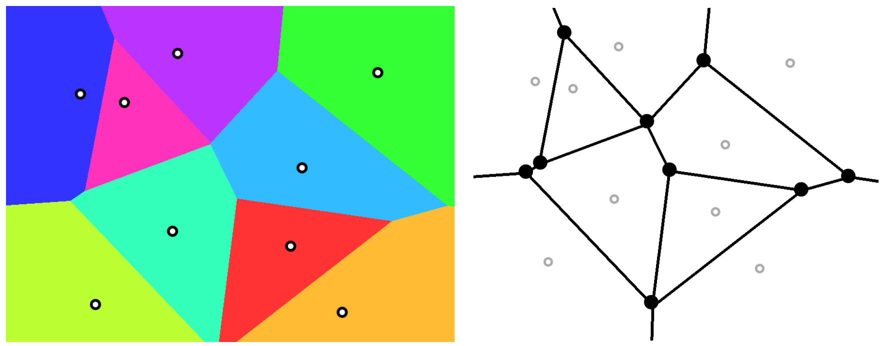
Note that the DCEL does not support infinite edges, so what we do is put a bounding box B with some padding around the vertices of $\mathrm{Vor}(P)$, and then intersect the infinite edges and faces with the boundary of B and only keep the part inside the bounding box.



The aim of our algorithms will then be to calculate the DCEL in the right figure.

1.2 Euclidean Voronoi Diagrams

In this section we focus on the Voronoi diagram when the norm is the Euclidean norm, that is $\|\cdot\|_2$. Here is the example from earlier:



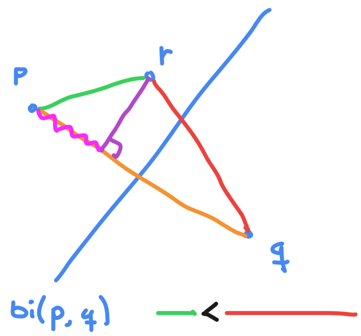
From linear algebra we know that $\|v\|_2 = \sqrt{\langle v, v\rangle}$, where $\langle\,\cdot\,,\,\cdot\,\rangle$ is the usual dot product on $\mathbb{R}^2$. Given two points $p, q \in \mathbb{R}^2$ then the **bisector** of p and q is denoted by $\mathrm{bi}(p, q) \subset \mathbb{R}^2$ and denotes the set of points on a line ℓ which passes through the midpoint of p and q and is orthogonal (w.r.t. $\langle\,\cdot\,,\,\cdot\,\rangle$) to the vector $p - q$.

A bisector $\mathrm{bi}(p, q)$ splits the plane into two **half-planes** H_p and H_q such that $p \in H_p$ and $q \in H_q$. We define $h(p, q)$ to be the open half-plane which contains p, that is the interior of H_p. So we have that

$$\mathbb{R}^2 = h(p, q) \cup \mathrm{bi}(p, q) \cup h(q, p).$$

Proposition 1.3. $r \in h(p, q)$ if and only if $\mathrm{dist}(r, p) < \mathrm{dist}(r, q)$.

Proof.



(TODO: Formalize) Proof sketch: We want to project r onto the orange line. As long as $r \in H_p$ then the squiggly pink segment is shorter than the orange segment, which will make the green segment shorter than the red segment (which is what we want to show). □

Proposition 1.4. For every Voronoi cell we have

$$\mathcal{V}(p_i) = \bigcap_{\substack{1 \leq j \leq n \\ j \neq i}} h(p_i, p_j).$$

Proof. "$\subset$": Let $r \in \mathcal{V}(p_i)$. Then $\mathrm{dist}(r, p_i) < \mathrm{dist}(r, p_j)$ for all $i \neq j$. Prop 1.3 then gives us that this is equivalent to $r \in h(p_i, p_j)$ for all $i \neq j$.

"$\supset$": This argument is symmetrical to the above argument. □

A Voronoi cell is thus the intersection of convex sets and is therefore convex. We conclude that the Voronoi cells are open and convex (possibly unbounded) polygons with at most $n-1$ vertices and $n-1$ edges.

We now look at the shape of the entire Voronoi diagram:

Theorem 1.5. If the points in P are collinear then $\mathrm{Vor}(P)$ consists of $n-1$ parallel lines. Otherwise, $\mathrm{Vor}(P)$ is connected and its edges are either segments or half-lines.

Proof. (TODO: .) □

Finally, we show that that the complexity of the vertices and edges is $\mathcal{O}(n)$:

Theorem 1.6. For $n \geq 3$, the number of vertices in $\mathrm{Vor}(P)$ is at most $2n-5$ and the number of edges is at most $3n-6$.

Proof. (TODO: .) □

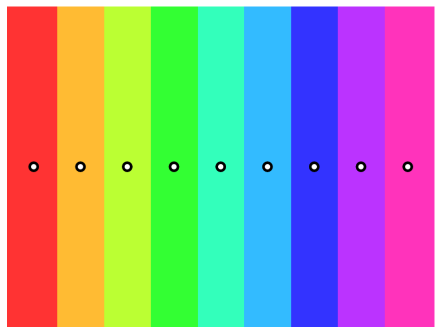
In the next section we will present an algorithm which computes $\mathrm{Vor}(P)$ in $\mathcal{O}(n \log n)$ time. This is actually optimal, as we can use a Voronoi diagram for sorting:

Theorem 1.7. We can't do better than $\mathcal{O}(n \log n)$.

Proof. Let $A = \{a_1, a_2, \ldots, a_n\} \subset \mathbb{R}$. Now assume we have used an algorithm to compute a Voronoi diagram of the points

$$P = \{(a_1, 0), (a_2, 0), \ldots, (a_n, 0)\}.$$

By Theorem 1.5 we obtain a diagram which looks similar to this:



We assume without loss of generality that the algorithm outputs a DCEL Δ of $\mathrm{Vor}(P)$. Assume that the edge pointer of every face of Δ points to the edge to the right of the face, and that the face pointer of every edge of Δ points to the face to the right. Let F_i be the face in Δ which contains the point $(0, a_i)$. Let $\ell \in \mathbb{N}$ such that $a_\ell < a_i$ for all $i \neq \ell$. Let $b_1 = a_\ell$ and if $b_i = a_j$ and $i < n$ then define $b_{i+1} = a_k$ where $F_j.\mathsf{edge}.\mathsf{face} = F_k$. Then $(b_1, b_2, \ldots, b_n)$ is the elements of A in sorted order. This means that we can use the Voronoi diagram to sort, which proves the theorem. $\square$

(TODO: The statement of the above theorem is temporary. I originally phrased it like so: "The optimal worst-case running time for computing $\mathrm{Vor}(P)$ is $\Omega(n \log n)$." What is the proper terminology here?)

1.3 Fortune's algorithm

Hello world.