Chapter 5

Recursive Geometry of the Flow Complex

Summary. The flow complex is a geometric structure, similar to the Delaunay tessellation, for organizing a set of (weighted) points in $\mathbb{R}^d$. Flow shapes are topological spaces corresponding to substructures of the flow complex. The flow complex and flow shapes have found applications in surface reconstruction, shape matching, image processing, and molecular modeling. In this chapter we give an algorithm for computing the flow complex of weighted points in any dimension. The algorithm reflects the recursive geometry of the flow complex.

Notes. The results in this and the following chapter were obtained under the supervision of Joachim Giesen. The algorithm was published in [25]. The analysis of the topology is an analysis of the results presented in [52] refined and generalized to arbitrary dimensions. The results of this and the following chapter will be published in [24].

The properties of the flow complex in Section 5.4 have been previously considered by Matthias John in his PhD thesis [82]. Lemmas 5.3, 5.4, 5.6 and 5.7 together with their proofs (except for minor changes) are from his thesis. For completeness we included these proofs here.

5.1 Introduction

In the previous chapters we discussed space-filling curves and Delaunay tessellations for organizing a set of points. Space-filling curves yield a linear order on the point set suitable for storing the points in a way that points which are close to each other in the linear order are also close in Euclidean space. We used the space-filling curve order of a point set to compute its Delaunay tessellation. Here we will use the Delaunay tessellation to compute a further geometric structure for organizing a point set, the flow complex.

The flow complex of a set of points has been successfully applied to surface reconstruction from a point cloud [69], to shape segmentation and matching [51], to gamut mapping [70] and to modeling properties of macromolecules in biogeometry [68]. It is a cell decomposition based on the flow in the direction of

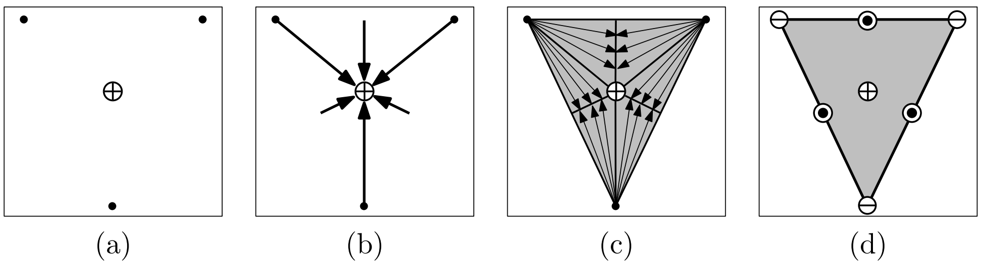


Figure 5.1: Illustration of the recursive structure.

the steepest ascent of the power distance function of a given set of weighted points. It is closely related to the Voronoi diagram and the Delaunay tessellation, in particular in the case of weighted points it is related to the power diagram and the regular triangulation (see e.g. [115]). The flow complex was introduced by Giesen and John [69]. Discrete flow was first used in 1995 for surface reconstruction by Edelsbrunner [60] (see also [61]) who defined a flow relation on the simplices of the Delaunay tessellation.

For computing the flow complex so far only algorithms specializing on the two and three dimensional complex are known. For instance in the case of unweighted, two-dimensional points the complex corresponds to the Gabriel graph which can easily be computed using the Delaunay tessellation. Here we want to obtain insight into the general structure of the complex in higher dimensions. The analysis of its geometric structure leads to an algorithm for computing the flow complex of (weighted) points in any dimension, which reveals a recursive geometry of the flow complex.

Example. In Section 5.5 we present an algorithm for constructing the flow complex of a set of points. The algorithm computes the inflow into any critical point of the distance function to a point set. Flow here is meant with respect to the direction of steepest ascent of this distance function and a critical point is a point with no unique direction of steepest ascent (later we will make these concepts more precise). The basic concept of the algorithm is to first compute the points that flow directly (on a straight line segment) into the critical point, then the direct flow into these points, and so on. We illustrate this by the simple example of Figure 5.1 with three input points (see Section 5.5.1 for a more complex example). By definition the flow at a point in the plane is in the direction in which the Euclidean distance to the closest input point (or points) increases most. Figure 5.1(a) shows the three input points (black dots) and a critical point ($\oplus$) of the flow. This critical point is a local maximum of the distance function. The flow into this critical point can be computed as follows: We start with the point itself. Direct flow into this point lies either on a line segments between one of the input points and the critical point or on one of the bisectors of two input points. This gives as direct flow the six (open) line segments shown in Figure 5.1(b). Next, we consider the direct flow into these line segments. There is only flow into the line segments on bisectors. Into each of these line segments there is flow from two (open) triangles shown in

Figure 5.1(c). There is no flow into these triangles. Thus the total flow into the critical point is the open triangle spanned by the input points. The flow in the example has six further critical points: the input points themselves (which are minima of the distance function and have themselves as inflow region) and the midpoints of the line segments between the input points (which are saddle points of the distance function and have the open line segments between the points as inflow region). The flow complex is shown in Figure 5.1(d).

5.2 Preliminaries

In this chapter we discuss geometric data structures on weighted points in $\mathbb{R}^d$.

Complexes. A *polyhedral complex* [155] $\mathcal{C}$ is a finite collection of polyhedra in $\mathbb{R}^d$ such that

- the empty polyhedron is in $\mathcal{C}$,
- if $C \in \mathcal{C}$ then all the faces of C are also in $\mathcal{C}$,
- the intersection $C_1 \cap C_2$ of two polyhedra $C_1, C_2 \in \mathcal{C}$ is a face both of C_1 and of C_2.

For polyhedra C_1, C_2 we will denote by $C_1 \leq C_2$ (and by $C_2 \geq C_1$) that C_1 is a face of C_2 (and C_2 a co-face of C_1) and by $C_1 < C_2$ (and by $C_2 > C_1$) that C_1 is a proper face of C_2 (and C_2 a proper co-face of C_1).

If the cells of a polyhedral complex $\mathcal{C}$ are all bounded then $\mathcal{C}$ is called *polytopal complex*. If the cells are all simplices then $\mathcal{C}$ is called *(geometric) simplicial complex*. The combinatorial structure of a simplicial complex is captured by an *abstract simplicial complex*, i.e., a family of subsets of $\{1, \ldots, n\}$ that is closed under taking subsets.

The Voronoi diagram is an example for a polyhedral complex. The Delaunay tessellation is a polytopal complex, and furthermore simplicial if the points (and for weighted points the corresponding balls) are in general position.

Power Distance. In the following the term *point* always refers to a weighted point and the terms *Delaunay tessellation* and *Voronoi diagram* always include also the weighted versions, i.e., the regular triangulation and the power diagram. Recall from Section 4.6 that the power distance of $x \in \mathbb{R}^d$ from a point p with weight w_p is defined as

$$\pi_p(x) = \|x - p\|^2 - w_p.$$

For two points p and q the set of all points with the same power distance to p and q is a hyperplane orthogonal to the line through p and q. This yields the following relation between a Voronoi face and its dual Delaunay face which we will use to derive properties of the flow complex.

Remark 5.1. *The affine hulls of a Voronoi face and its dual Delaunay face are orthogonal and intersect in exactly one point.*

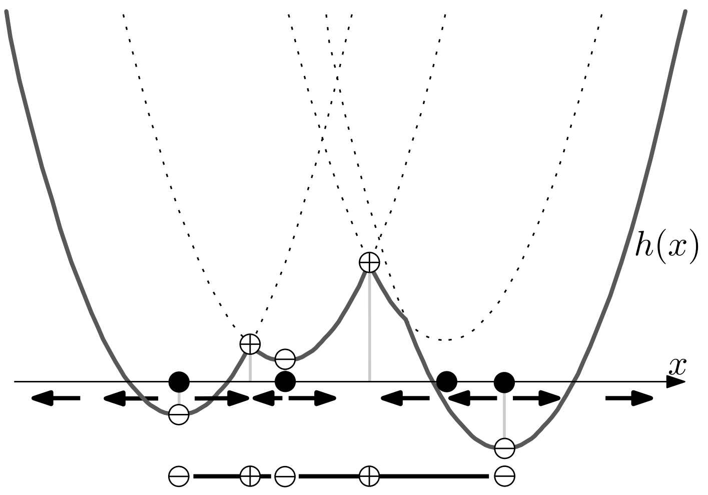


Figure 5.2: One-dimensional example of flow induced by a set of weighted points.

Relative Interior and Boundary. Let *aff*(C) denote the affine hull of C. The *relative interior* C° and the *relative boundary* ∂C of C are the interior and the boundary of C in *aff*(C). For instance the relative interior of a Delaunay edge contains all points in this edge besides the endpoints, i.e., the interior relative to the line through the edge. The relative interior of a vertex is the vertex itself. We always refer to the interior and to the boundary of convex sets with respect to their dimension, i.e., to their relative interior and boundary.

The closure of C will be denoted by $\overline{C}$.

For computing the flow complex the relative interiors and boundaries of (the underlying space of) Voronoi faces will play a central role. The relative boundary of a Voronoi face is (the underlying space of) the union of its proper faces.

5.3 Flow and the Flow Complex

5.3.1 The Distance Function and its Critical Points

Distance Function. The flow that we are going to study is the flow in direction of the steepest ascent of the *distance function*

$$h(x) = \min\{\,\pi_p(x)|\,p \in P\}$$

induced by the points in P.

Figure 5.2 shows a one-dimensional example with weighted points. The points in P are shown as black dots. The weight w_p of a point p is the signed distance from the point to the apex of the parabola below it (positive weight) or above it (negative weight) it. The graph of the distance function h is the lower envelope of the parabolas.

Critical and Regular Points. For a smooth function the direction of steepest ascent corresponds to the direction of the gradient. Here, h is not smooth on the boundary of Voronoi cells but the notion of *generalized gradients* is applicable. Generalized gradients have been defined in the context of critical point theory for distance functions on Riemannian manifolds [34, 74, 121] pioneered by Grove and Shiohama [75].

In the case of the Euclidean space $\mathbb{R}^d$ as Riemannian manifold and h as distance function, generalized gradients can be described as follows: Let $N(x) \subset P$ be the set of nearest neighbors of x in P, i.e., the set of all $p \in P$ with $\pi_p(x) = h(x)$. For a point $x \in \mathbb{R}^d$ the set of *generalized gradients* at x is the set of unit vectors $v \in \mathbb{R}^d$ pointing away from all points in $N(x)$, i.e., for which

$$\langle v, x - p \rangle > 0 \quad \text{for all } p \in N(x).$$

For instance, if $N(x)$ only contains one point then h is smooth in x and the generalized gradient is the open hemisphere with the unit vector in direction of the gradient as center.

A point is called *critical* if the set of generalized gradients is empty, otherwise it is called *regular.* Thus, a point $x \in \mathbb{R}^d$ is critical if and only if $x \in conv(N(x))$. If $x \in \partial conv(N(x))$, then it is a *degenerate* critical point. In the following we assume that all critical points are non-degenerate. This directly yields a description of the critical points in terms of the Voronoi diagram and the Delaunay tessellation. The convex hull $conv(N(x))$ is the Delaunay face dual to the Voronoi face corresponding to $N(x)$ in which x lies.

Remark 5.2. *The critical points of h are exactly the intersections of Voronoi faces and their dual Delaunay faces.*

In the one-dimensional example of Figure 5.2 the critical points are the local maxima ($\oplus$) and minima ($\ominus$) of the distance function h. In higher dimensions saddle points are further critical points. In the case of weighted points as in the example, points in P are not necessarily minima, thus not necessarily critical points. The arrows below the x-axis indicate the direction of the generalized gradient which is unique in the one-dimensional case.

In the following we define a vector field in accordance with this theory. We will prove later that the vector at a regular point corresponds to the direction of steepest ascent of the distance function.

5.3.2 A Vector Field of Generalized Gradients

Vector Field and Drivers. For $x \in \mathbb{R}^d$ the point

$$\delta(x) = \operatorname{argmin}_{y \in conv(N(x))} \|x - y\|^2$$

is called the *driver* of x.

We define a vector field $v : \mathbb{R}^d \to \mathbb{R}^d$ by

$$v(x) := \begin{cases} \frac{x-\delta(x)}{\|x-\delta(x)\|} & \text{if } x \text{ is a regular point of } h \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

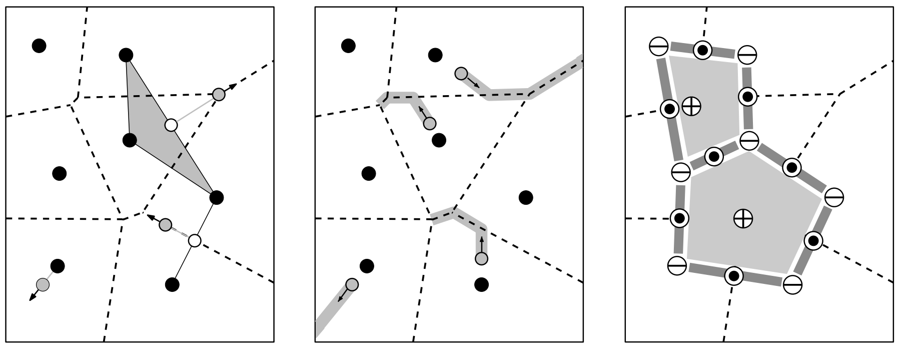

(a) drivers (b) orbits (c) stable manifolds

Figure 5.3: Two-dimensional example of flow induced by a set of unweighted points.

The drivers and vectors are illustrated in Figure 5.3(a) for a set P of unweighted points in $\mathbb{R}^2$. The points in P are shown as black dots and the 1-skeleton of their Voronoi diagram as dashed lines. For the three gray points the corresponding drivers are shown and the corresponding vectors are indicated as arrow. For the gray point in the interior of a Voronoi cell of a point p the driver is the point p. For the gray point on a Voronoi segment the driver shown as white point is the intersection of the Voronoi segment and the dual Delaunay segment. For the gray point at a Voronoi vertex the driver shown as white point is the closest point on the dual Delaunay triangle.

Flow. The *flow* $\phi : [0,\infty) \times \mathbb{R}^d \to \mathbb{R}^d$ is defined by the equations

$$\begin{aligned} \phi(0,x) &= x \\ \lim_{t \downarrow t_0} \frac{\phi(t,x)-\phi(t_0,x)}{t-t_0} &= v(\phi(t_0,x)). \end{aligned}$$

If the first parameter is interpreted as time then $\phi(t,x_0)$ can be interpreted as the point reached at time t by starting at time 0 at x_0 and following the direction given by v. For $x_0 \in \mathbb{R}^d$

$$\phi_{x_0}(t) := \phi(t,x_0)$$

is called the *orbit* of x_0. Figure 5.3(b) shows the orbits for some points for the two-dimensional example used before.

If $\phi(t,x_0) = x_0$ for all $t \geq 0$ then x_0 is a *fixed point* of the flow. The fixed points of the flow are exactly the critical points of the distance function h.

5.3.3 The Flow Complex

Stable Manifolds. Points in $\mathbb{R}^d$ flow either into a critical point or to infinity as Figure 5.3(b) indicates. We are interested in a decomposition of $\mathbb{R}^d$ into regions $S(c)$ such that all points in a region flow into the same critical point c, i.e., the *stable manifold* of c

$$S(c) = \left\{ x \in \mathbb{R}^d \,\middle|\, (\exists t \geq 0)\, \phi(t,y) = c \right\}.$$

Note that the points that flow to infinity do not lie in the stable manifold of a critical point. To obtain a decomposition of $\mathbb{R}^d$ we could add a designated critical point at infinity.

Flow Complex. The *flow complex* is the collection of closures of the stable manifolds of all critical points (not including a critical point at infinity).

For the one-dimensional example of Figure 5.2 the stable manifolds are line segments and points. They are shown below the graph. The stable manifold of a minimum is the minimum itself. The stable manifold of a maximum is a full-dimensional open set, i.e., a segment in the one-dimensional example. The flow complex in this example contains the three minima and the two (closed) segments between them.

For the two-dimensional example the stable manifolds are shown in Figure 5.3(c). The distance function h has two maxima, and their stable manifolds are the interiors of the two polygons depicted. The open segments on their boundary are the stable manifolds of the saddle points of h and the vertices are the stable manifolds of the minima, i.e., the points in P. The flow complex is the closure of these stable manifolds, i.e., the two polygons and the segments and vertices on their boundaries.

5.4 Properties of the Flow

Most of the properties proved in this section are needed to prove the correctness of the algorithm in the next section. Furthermore, we prove that the vector $v(x)$ at a regular point $x \in \mathbb{R}^d$ corresponds to the direction of steepest ascent of $h(x)$.

The following lemma yields a description of drivers in terms of Voronoi diagrams. In particular it yields that the number of Voronoi faces is an upper bound on the number of drivers. In the algorithm for computing the flow complex we use it to compute drivers and to treat the flow within a Voronoi face uniformly.

Lemma 5.3 ([24, 82]). *Let V be an arbitrary Voronoi face and D its dual Delaunay face. Then all points x in the interior of V have the same driver d which is the point in D closest to $\mathit{aff}(D) \cap \mathit{aff}(V)$.*

Proof. The proof is illustrated in Figure 5.4. Note that for this example the weight of the left vertex of D must be small compared to the weight of the right vertex.

Let D be the Delaunay face dual to V and let z be the intersection point of the affine hulls $\mathit{aff}(V)$ and $\mathit{aff}(D)$ (see Remark 5.1). Let $\delta \in D$ be the point closest to z. Note that it is possible that $\delta = z$.

Let y be any point in D. Since $y - z$ is orthogonal to $x - z$ we have by Pythagorean theorem (denoted in the following by P.t.)

$$\begin{aligned}
\|x-y\|^2 &\overset{\text{P.t.}}{=} \|x-z\|^2 + \|y-z\|^2 \\
&\geq \|x-z\|^2 + \|\delta-z\|^2 \\
&= \|\delta-x\|^2.
\end{aligned}$$

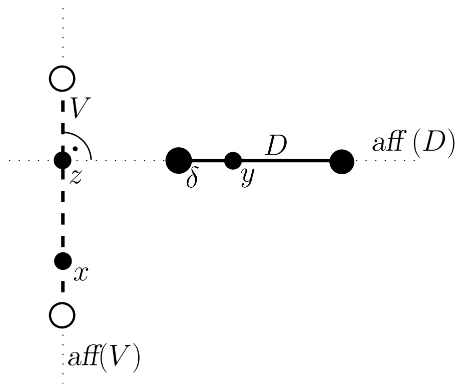


Figure 5.4: Illustration of the proof of Lemma 5.3.

Thus δ is the closest point in $D = conv(N(x))$ to x and therefore its driver. The argument holds for any point having V as lowest dimensional Voronoi face containing it, i.e., for all points in V°. $\square$

For a point x in the Voronoi cell of p we have $h(x) = \pi_p(x)$, thus h is determined by the power distance to its driver. To generalize this we assign a power to every driver. We show in the following lemma that with these powers $h(x)$ is determined by $\pi_{d(x)}$ for all x. The power of a driver will be used as a tool to prove the acyclicity of the flow.

For a driver d let D be the lowest dimensional Delaunay face that contains d, and let p be a vertex of D. We define

$$w_d = -\pi_p(d).$$

Observe that the value w_d is independent of the choice of p, as follows from the following lemma. The intuition behind this definition is that the graphs of the distance functions π_p of the vertices p of D intersect in a lower-dimensional paraboloid with apex d and weight w_d.

Lemma 5.4 ([24, 82]). *For a Voronoi face V and the driver δ of V°*

$$h(x) = \pi_\delta(x) \text{ for all } x \in V.$$

Proof. Let $p' \in P$ be a vertex of the lowest dimensional Delaunay face D' containing δ. In particular, p' is a point defining V, thus

$$h(x) = \pi_{p'}(x)$$

for any point $x \in V$. Since D' is a convex polytope and δ is the closest point on D' to x, $x - \delta$ and $p' - \delta$ are orthogonal.

Therefore,

$$\begin{aligned} h(x) &= \pi_{p'}(x) \\ &= \|p'-x\|^2 - w_{p'} \\ &\overset{\text{P.t.}}{=} \|\delta - x\|^2 + \|p'-\delta\|^2 - w_{p'} \\ &= \|\delta - x\|^2 + \pi_{p'}(\delta) \\ &= \|\delta - x\|^2 - w_\delta \\ &= \pi_\delta(x). \end{aligned}$$

□

Like π_p for $p \in P$ the power distance π_δ of a driver has the property that it is never below h.

Lemma 5.5. *For a driver δ*

$$h(x) \le \pi_\delta(x) \text{ for all } x \in \mathbb{R}^d.$$

If D is the lowest dimensional Delaunay face containing δ and V its dual Voronoi face then

$$h(x) < \pi_\delta(x) \text{ for all } x \notin \mathit{aff}(V).$$

Proof. Let p be a vertex of D. By definition of w_δ we have

$$\pi_\delta(x) = \|x-\delta\|^2 + \|\delta - p\|^2 - w_p.$$

Thus, $\pi_\delta(x) \ge \pi_p(x)$ if and only if the angle $\measuredangle(p,\delta,x) \le \pi/2$ by the law of cosines. Since $\delta \in D$ and D is convex this holds for at least one of the vertices $p \in D$. If $x \notin \mathit{aff}(V)$ then there is a vertex $p \in D$ with $\measuredangle(p,\delta,x) < \pi/2$. □

The following lemma provides a condition under which a Voronoi face V shares its driver with a co-face V' of V.

Lemma 5.6 ([24, 82])**.** *Let V be a Voronoi face with co-face V'. Let δ and δ' be the drivers of V° and V'°, respectively. If no line segment connecting a point of V with δ' intersects V'° then $\delta = \delta'$. Otherwise, $\delta \ne \delta'$ and $w_\delta < w_{\delta'}$.*

Proof. Let D and D' be the Delaunay faces dual to V and V', respectively. Since $V < V'$, we have $D > D'$. Let y be any point in the interior of V. Let z be the intersection point of the affine hull of D and V. As $y \in V$ and $\delta, \delta' \in D$, $\delta - z$ and $\delta' - z$ are orthogonal to $y - z$.

First we prove

$$w_{\delta'} - w_\delta = \|z-\delta'\|^2 - \|z-\delta\|^2.$$

which is equivalent to $\pi_{\delta'}(z) = \pi_\delta(z)$.

We have

$$\begin{aligned} \pi_\delta(z) &= \|\delta - z\|^2 - w_\delta \\ &\overset{\text{P.t.}}{=} \|\delta - y\|^2 - \|y-z\|^2 - w_\delta \\ &= \pi_\delta(y) - \|y-z\|^2 \\ &= h(y) - \|y-z\|^2 \end{aligned}$$

and by the same argument $\pi_{\delta'}(z) = h(y) - \|y - z\|^2$. Thus, $\pi_{\delta'}(z) = \pi_\delta(z)$.

Now we show that if the line segment $L_{y,\delta'}$ from y to δ' intersects the interior of V' we have $\|z - \delta'\| > \|z - \delta\|$ which implies $w_{\delta'} > w_\delta$ and that we get $\delta = \delta'$ otherwise. This part of the proof is illustrated in Figure 5.5. Note that in the example of Figure 5.5(b) the weight of w_q must be large relative to w_p.

Let H_1 be the affine hull of D. Let H_2 be the hyperplane through δ' orthogonal to $L_{y,\delta'}$. H_1 is subdivided by H_2 in two half-subspaces. D' is in the closed half-subspace not containing y since D' is convex and δ' is the closest point of D' to y.

Let q be any Delaunay vertex of D which is not a vertex of D'. The point q lies in the same half-subspace of H_1 as the convex hull of $D' \cup \{q\} \subset D$ and the point δ' lies in D'. So if q lies in the same half-subspace as z as in the case of Figure 5.5(a) there is a point of D closer to z than δ' and therefore $\|z - \delta'\| > \|z - \delta\|$. If any such q lies in a different half-subspace than z as in the case of Figure 5.5(b) then δ' is the closest point of D to z and thus $\|z - \delta'\| = \|z - \delta\|$.

If $L_{y,\delta'}$ intersects V' then q lies in the same half-subspace as z. Otherwise they lie in different half-subspaces. This proves the lemma. $\square$

A consequence of this lemma is the following.

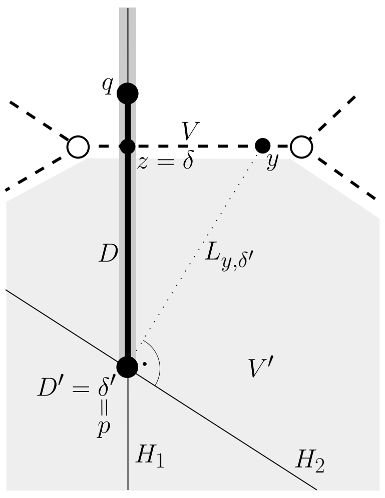
Lemma 5.7 ([24, 82]). *For all $y \in \mathbb{R}^d$ there is an $\varepsilon_0 > 0$ such that*

$$\delta(y + \varepsilon v(y)) = \delta(y) \text{ for all } 0 < \varepsilon < \varepsilon_0.$$

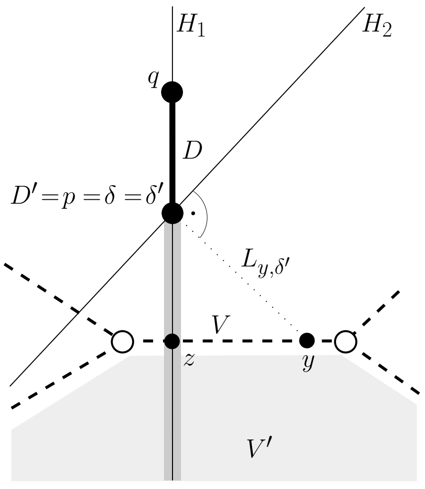
Proof. Let V be the Voronoi face with $y \in V^\circ$. For a sufficiently small $\varepsilon > 0$

$$y' := y + \varepsilon v(y) \in V'^\circ$$

with V' a Voronoi face fulfilling $V \leq V'$.



(a) $L_{y,\delta'}$ intersects V' and $\delta \neq \delta'$.



(b) $L_{y,\delta'}$ does not intersect V' and $\delta = \delta'$.

Figure 5.5: The two possible cases in Lemma 5.6.

For $V = V'$ we have $\delta(y) = \delta(y')$. Assume $V < V'$ and $\delta(y) \neq \delta(y')$. Thus, by Lemma 5.6 $L_{\delta(y'),y}$ intersects the interior of V'. For an illustration refer again to Figure 5.5 but now with the driver δ placed beyond z. Let H be the hyperplane orthogonal to $L_{\delta(y),\delta(y')}$ containing y. Since the vector from $\delta(y)$ to y points into V' and the vector from $\delta(y')$ to y out of V', $\delta(y)$ and $\delta(y')$ lie on different sides of H. But then the intersection of H with $L_{\delta(y),\delta(y')}$ is closer to y than $\delta(y)$ and lies inside D, contradicting that $\delta(y)$ is the driver of y. □

Theorem 5.8 (Orbits)**.** *The orbit of a point is a polygonal chain (possibly ending with a ray to infinity). The orbit passes through each Voronoi face at most once. The weights of the drivers increase along an orbit.*

Proof. For Voronoi faces V, V' with $V < V'$ Lemma 5.6 yields that the power of the driver along the orbit stays the same if the orbit goes from V° to V'° and strictly increases if it goes from V'° to V°. In the second case the driver changes and a line segment of the polygonal chain ends. Thus, we have an increasing sequence of drivers along an orbit and the orbit cannot enter a Voronoi face multiple times. □

Theorem 5.9 (Steepest Ascent)**.** *For all regular points $x \in \mathbb{R}^d$, the vector $v(x)$ points in the direction of steepest ascent.*

Proof. At a point $x \in \mathbb{R}^d$ with driver δ by Lemma 5.4, $h(x) = \pi_\delta(x)$. For a sufficiently small $\varepsilon_0 > 0$, we have by Lemma 5.7 that δ is the driver of all points $x + \varepsilon v(x)$ for all $0 \leq \varepsilon < \varepsilon_0$ and therefore

$$h(x + \varepsilon v(x)) = \pi_\delta(x + \varepsilon v(x)).$$

For all unit vectors v' we have

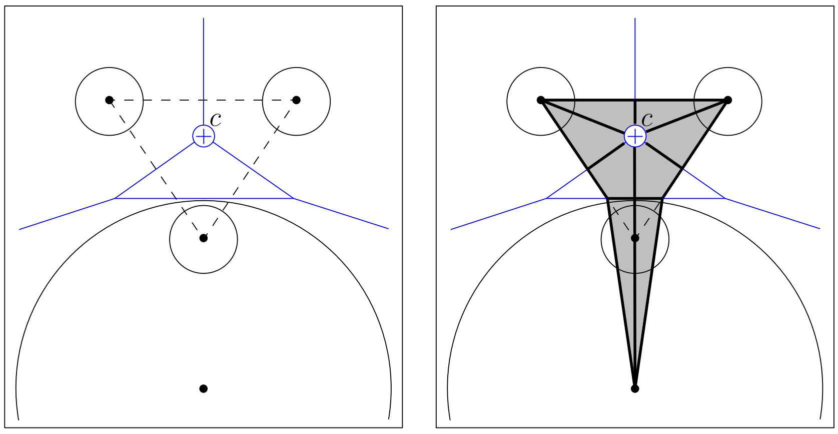
$$h(x + \varepsilon v') \leq \pi_\delta(x + \varepsilon v') \leq \pi_\delta(x + \varepsilon v(x)) = h(x + \varepsilon v(x)),$$

where the first inequality follows from Lemma 5.5, the second inequality from the definition of π_δ and v, and the last equality from the argument above. If $v' \neq v(x)$ the second inequality holds strictly, thus, $v(x)$ is the direction of steepest ascent at x. □

5.5 Recursive Geometry

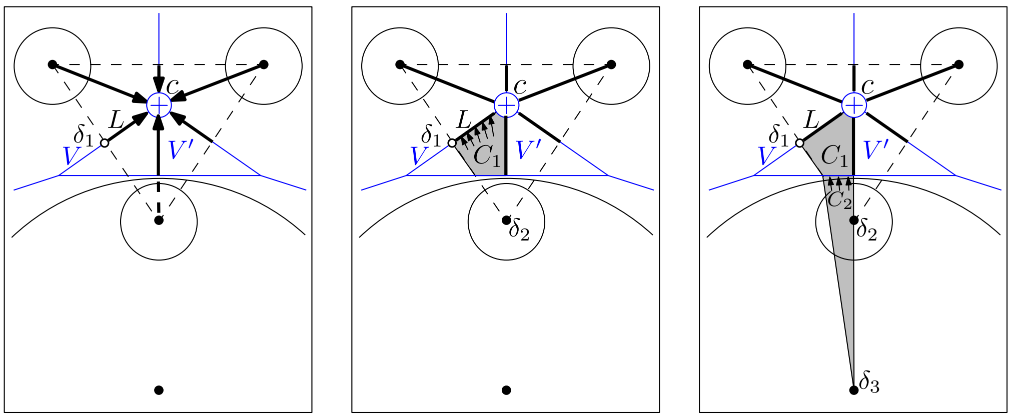
Outline. The flow complex consists of the closures of the stable manifolds of the critical points of the distance function. Thus, in order to compute the flow complex we need to compute these closures.

We present an algorithm that computes the stable manifold for a given critical point. The algorithm makes use of the close relationship of the flow complex to the Delaunay tessellation and the Voronoi diagram of P. The critical points can be determined using their description as intersections of Voronoi faces and their dual Delaunay faces (Remark 5.2). The algorithm yields a description of the closure of a stable manifold as a polytopal complex.



(a) weighted point set (b) stable manifold of c

Figure 5.6: A decomposition of a stable manifold as computed by the algorithm.



(a) inflow into c (b) inflow into L from V' (c) inflow into C_1

Figure 5.7: Several steps in the computation of the stable manifold of c.

5.5.1 Example

Before presenting the algorithm we illustrate the recursive nature of the flow by the example of Figures 5.6 and 5.7. Figure 5.6(a) shows a set of four weighted points together with its Voronoi diagram and Figure 5.6(b) shows the stable manifold of the local maximum c. The points are depicted as black dots. Their weights are depicted by the circles, where the weight of a point corresponds to the squared radius of a circle. To demonstrate effects that do not appear in the unweighted two-dimensional case, one of the points has a very large weight. As a consequence one of the points does not lie in its Voronoi cell and the closure of the stable manifold has non-simplicial cells.

We want to compute the stable manifold of the local maximum c of h depicted as $\oplus$. Thus, we want to compute the *inflow* into c, i.e., all points eventually flowing into c. The orbits of points flowing into c are polygonal chains ending at c (Theorem 5.8). Therefore, any point flowing into c either lies on a line segment flowing into c or flows into a line segment flowing into c. To such a line segment we have a corresponding driver, thus we have only a finite number of such line segments (Lemma 5.3). In terms of inflow we have that the inflow into c is the union of these line segments and their inflow. This already indicates the recursive nature of the flow.

Figure 5.7(a) shows the line segments flowing into c. Consider the line segment L and the driver δ_1 of the Voronoi segment V in the figure. Thus all points between δ_1 and c are driven into c by δ_1. In general any line segment flowing directly into c must flow through a co-face of the Voronoi face containing c and is driven by the driver of the interior of this co-face. In the example of Figure 5.7(a) the Voronoi face containing c is c itself, the co-faces are the Voronoi edges and cells containing this vertex, and the line segments flowing directly into c are drawn in bold.

Next consider the inflow into L. Since L (without its endpoints) lies in the interior of the Voronoi segment V, inflow into L must again come from co-faces of V. In the example there is inflow coming from both neighboring Voronoi cells. Consider the inflow coming from V' as depicted in Figure 5.7(b). The driver of V'° is δ_2. All points of V' between L and δ_2 are driven by δ_2 into L. Since δ_2 lies outside of V' the resulting set of points C_1 is $conv\{L, \delta_2\}^\circ \cap V'$.

Now V' is a Voronoi cell, so there is no inflow from co-faces of V'. But since δ_2 was cut-off from C_1 by the boundary of V', inflow coming from this boundary must be considered. The inflow into C_1 comes from C_2. The driver δ_3 lies in its Voronoi cell and is therefore not cut-off.

Computing all inflow into c in this way yields the stable manifold of c as depicted in Figure 5.6(b).

5.5.2 Algorithm

Next we formulate the algorithm to compute the stable manifold of a critical point c, i.e., the inflow into c. The recursive nature of the flow as illustrated in the example above yields that the inflow into the relative interior C° of a convex set C is the set of all points flowing along a straight line into C° plus the inflow into these points. We therefore formulate the algorithm Inflow with a parameter C. To compute the stable manifold of c we call the algorithm with $C = \{c\}$.

The Voronoi faces from which points might flow directly into C° can be easily determined if we know that C° lies in V° for a Voronoi face V. We therefore keep track of this face as a second parameter of the algorithm. To compute the stable manifold of c this parameter is set to the lowest dimensional Voronoi face V containing c. Relative interiors are used to get a unique decomposition of the Voronoi diagram and the stable manifolds.

We can now formulate the algorithm (Algorithm 6).

Algorithm 6: INFLOW(Convex polytope C, Voronoi face V with $C \subset V$)

```
1  $\mathcal{I} := \{C\}$
2  for each Voronoi face $V'$ with $V < V'$ do
3      $\delta :=$ driver of $V'^{\circ}$
4      $C' := conv(C, \delta)$
5      if $C'^{\circ} \cap V' \not\subseteq C$ do
6          $\mathcal{V} := \{ V'' \leq V' \mid C'^{\circ} \cap (V'')^{\circ} \not\subseteq V \}$
7          for each $V'' \in \mathcal{V}$ do
8              $\mathcal{I} := \mathcal{I} \cup$ INFLOW$(C' \cap V'', V'')$
9          end for
10     end if
11 end for
12 return $\mathcal{I}$
```

We first describe how the algorithm works and then prove its correctness in Theorem 5.10. We do not claim the algorithm to be optimal and do not analyze the complexity of the algorithm beyond proving that it terminates. Nonetheless, its complexity would be of interest in particular as an upper bound for the complexity of the flow complex. The algorithm is polynomial in the number of *combinatorially different* orbits, where we call two orbits combinatorially different if they traverse a different sequence of Voronoi faces.

Since the inflow has to contain C we add C to it in line 1 of the algorithm. We have to take care of the inflow into the relative interior of C that comes through the boundary of V or through higher dimensional Voronoi faces that contain V in their boundary. Since the algorithm INFLOW only takes care of the higher dimensional Voronoi faces we need to guarantee that any flow coming through the boundary of V has been handled when INFLOW is called.

Note that in the special case of $C = \{c\}$ there cannot be any inflow from within the Voronoi face V (and thus from the boundary of V) since in this case c is the unique driver of the relative interior of V that pushes away all other points in this relative interior (Lemma 5.3).

In the loop in lines 2 to 11 we take care of the inflow via all Voronoi faces V' that contain V in their boundary. The relative interior of any Voronoi face V' has a unique driver δ that we determine in line 3 (using Lemma 5.3). All points that flow via V' into the relative interior of C have to be contained in the intersection of V' with the relative interior of the pyramid C' whose apex is δ and whose base is C. If V' does not contain its driver δ, i.e., if δ is not a critical point, then the whole pyramid cannot be contained in V' but is cut off at the boundary of V'. This can result in a non-simplicial cell as the example of C_1 in Figure 5.7(b) shows.

In lines 5 to 10 we take care of the inflow coming from V' and its boundary. By definition of $\mathcal{V}$ in line 6 $V' \in \mathcal{V}$ therefore the inflow into the cut-off pyramid coming from higher-dimensional Voronoi faces is computed by a recursive call of the algorithm in line 8. By construction there is no additional flow from the relative interior of V' (Lemma 5.3). We now handle flow into the cut-off pyramid coming from the boundary of V' by considering all possible cases. The polytope C is not driven into V'. We therefore do not add faces of V' to $\mathcal{V}$ if their intersection with C'° lies in V. If the driver δ of V' lies on the boundary of V' then it has to be a critical point and is therefore not driven into V'. In the algorithm δ will in this case not be added to the inflow since it is not in the interior of C'. Any further point that is in C' but outside C'° is driven past it by the common driver δ (Lemma 5.6). Therefore, any flow coming from the boundary of V' must come from points in C'° which are taken care of in line 6.

The recursion stops when there is no more inflow through higher dimensional Voronoi faces or through the boundary of a Voronoi face to consider.

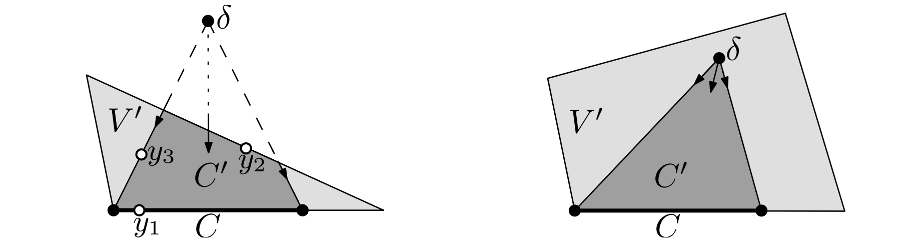
Theorem 5.10. *Let c be a critical point of the distance function h and let V be the lowest dimensional Voronoi face containing c. Then* Inflow$(\{c\}, V)$ *computes the stable manifold of c.*

Proof. First we show that the algorithm terminates. For this we need to prove that the depth of recursion can be bounded. If the recursion depth is at least m we get a sequence $(C_1, V_1), (C_2, V_2), \ldots, (C_{m-1}, V_{m-1}), (C_m, V_m) = (c, V)$ of pairs of polytopes and Voronoi faces, and a point $x \in C_1 \setminus C_2$ which flows through all the polytopes starting at C_1. By Theorem 5.8, x flows at most once through any Voronoi face. Therefore m is bounded by the complexity of the Voronoi diagram and the algorithm terminates.

Next we need to prove that the computed face S' is indeed the stable manifold S of c. From the description of the algorithm above it is clear that $S' \subseteq S$. If we assume $S \nsubseteq S'$ then there would be a point x which flows into c but is outside of S'. Since c is in S' the point x eventually must flow into S'. Assume the polytope C' is the polytope of S' into which x flows first. We may assume $C' \neq \{c\}$ and therefore C' has the form $C' = conv(C, \delta) \cap V$ with C a convex polytope, V a Voronoi face with $C^{\circ} \subset V^{\circ}$, and δ the driver of V°.

Then x cannot flow into C' through the relative interior of V because it would be driven away by δ. But x can also not come from a lower-dimensional Voronoi face because then it would either come through C which is part of S' or would be driven into V by δ. But then the algorithm would have added the corresponding polytope to $\mathcal{I}$ in line 8 at the same time it added C'. The algorithm checks for flow from higher-dimensional Voronoi faces in the main loop, so x can also not flow into C' by a higher-dimensional face. Therefore x must already have been in S' and therefore $S \subset S'$. In total we have $S = S'$, i.e., the algorithm computes the stable manifold of c. □

We have not considered the running time of the algorithm beyond observing that it terminates. Theorem 5.8 yields an exponential bound on the running time but it is open whether the running time is polynomial. The complexity of the algorithm would give insight into the complexity of the flow complex which also is an open problem. Along the lines of the previous chapters, it would be interesting to bound the complexities (of the flow complex and the algorithm) under assumptions on the point distribution. In particular for uniformly distributed points we expect that the recursion depth in the algorithm drastically decreases.



(a) $C' = conv(C, \delta) \cap V'$ with $\delta \notin V'$ (b) $C' = conv(C, \delta)$ and $\delta \in V'$

Figure 5.8: Illustration of the proof of Lemma 5.11: a polytopal face C' of a stable manifold.

5.5.3 Properties of the Stable Manifolds

The algorithm gives insight into the structure of the stable manifolds of critical points and the relationship between neighboring stable manifolds. Most importantly, the algorithm directly gives a polytopal decomposition of the closures of stable manifolds and therefore of the flow complex.

In the following we discuss the relationship between neighboring stable manifolds. Let S_1 and S_2 be the stable manifolds of the critical points c_1 and c_2, respectively. Let C_1 and C_2 be maximal polytopal faces of the closures of S_1 and S_2, respectively, and let $C_1 < C_2$, thus C_1 lies on the boundary of S_2. We denote this situation by $S_1 < S_2$.

We prove that in this case the critical point c_1 lies on the boundary of S_2 and c_1 has a smaller distance function value than c_2.

Lemma 5.11. *Let S_1 and S_2 be the stable manifolds of critical points c_1 and c_2, respectively. If $S_1 < S_2$, then c_1 lies on the boundary of S_2 and $h(c_1) < h(c_2)$.*

Proof. Suppose c_1 does not lie on the boundary of S_2. Then the orbit of any point in S_1 must eventually leave the boundary of S_2. Assume an orbit leaves the boundary at a point y. Then y is on the boundary of a polytopal face computed by the algorithm. Let C' be the first polytopal face of $\overline{S_2}$ computed by the algorithm with y on its boundary. We have $C' \neq \{c_2\}$. Thus C' is of the form $C' = conv(C, \delta) \cap V'$ with C a previously computed polytopal face of $\overline{S_2}$, V' the Voronoi face with $C'^{\circ} \subset V'^{\circ}$, and δ the driver of V'°. The situation is illustrated in Figure 5.8, where Figure 5.8(a) shows the case where $\delta \notin V'$ and Figure 5.8(b) the case where $\delta \in V'$.

We consider the different cases for the position of y on the boundary of C'. The case $y \in C$, e.g., y_1 in Figure 5.8(a), is not possible by the assumption that C' was the first polytope found by the algorithm with y on its boundary. If y is in V'° (as y_3 in the example of Figure 5.8(a) or δ in the example of Figure 5.8(b)) then y has δ as a driver and the orbit of y does not leave C' at y. Now, consider the case that y is on the boundary of V' and the line segment $L_{\delta,y}$ does not intersect the interior of V' (as y_2 in Figure 5.8(a)). Then by Lemma 5.6 the driver of y is δ, thus y cannot flow away from C'.

Therefore, the orbit of a point in S_1 on the boundary of S_2 cannot leave the boundary of S_2, and the critical point c_1 of S_1 must lie on the boundary of S_2.

If we reconsider the case distinction for the position of a point y on the boundary of S_2 (Figure 5.8), we see that $y \in \partial C' \setminus C$ can only be a critical point if it is actually the driver δ of C'°. Thus the critical point c_1 is the driver δ of a point $x \in S_2$ with $x \neq \delta$. By Lemmas 5.4 and 5.5 we have

$$h(c_1) \leq \pi_{c_1}(c_1) < \pi_{c_1}(x) = h(x).$$

The point x flows into c_2 and h increases in the direction of the flow, thus $h(x) \leq h(c_2)$ which proves the lemma. $\square$

Conclusion

We presented an algorithm for computing the flow complex of a weighted point set in any dimension. The algorithm reveals the recursive geometry of the flow complex which we will use in the following chapter to analyze the topology of the flow complex and its sub-complex.

A main open problem is the complexity of the flow complex. The complexity of the algorithm would give insight into the complexity of the flow complex but this again is an open problem. Theorem 5.8 yields an exponential bound on the running time but it is open whether the running time is polynomial.

Along the lines of the previous chapters, it would be interesting to bound the complexities (of the flow complex and the algorithm) under assumptions on the point distribution. In particular for uniformly distributed points we expect that the recursion depth in the algorithm drastically decreases.