α-Functions
Piecewise-linear Approximation from Noisy and Hermite Data

MARC ALEXA, TU Berlin, Germany

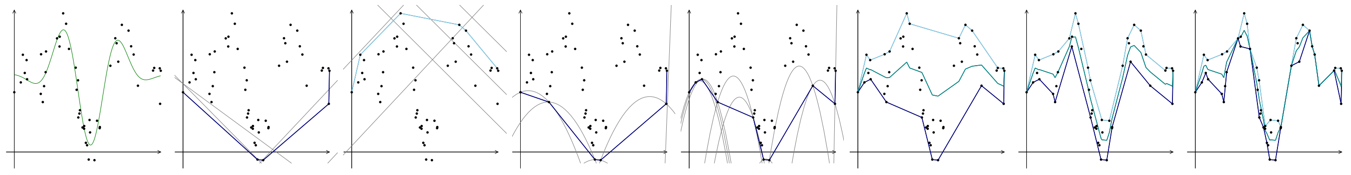


Fig. 1. A function is represented with noisy data points. The lower and upper convex hull crudely approximate the data by fitting lines. Replacing the lines with parabolas of the form $-\frac{1}{2}\alpha x^2$ creates more detailed lower hulls as α increases (from left to right). The function can be approximated as the mid-contour of the lower and upper α-hull

We introduce α-functions, providing piecewise linear approximation to given data as the difference of two convex functions. The parameter α controls the shape of a paraboloid that is probing the data and may be used to filter out noise in the data. The use of convex functions enables tools for efficient approximation to the data, adding robustness to outliers, and dealing with gradient information. It also allows using the approach in higher dimension. We show that α-functions can be efficiently computed and demonstrate their versatility at the example of surface reconstruction from noisy surface samples.

CCS Concepts: • **Computing methodologies → Mesh models**.

Additional Key Words and Phrases: piecewise-linear approximation, convex-concave split, Legendre transformation, weighted Delaunay triangulation

ACM Reference Format:
Marc Alexa. 2022. α-Functions Piecewise-linear Approximation from Noisy and Hermite Data. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Proceedings (SIGGRAPH '22 Conference Proceedings), August 7–11, 2022, Vancouver, BC, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3528233.3530743

1 INTRODUCTION

Piecewise-linear (PL) functions are a convenient and commonplace data representation in virtually all engineering disciplines, and particularly relevant for two and three-dimensional signals. PL functions are often generated from scattered data samples: locations $\mathbf{x}_i$ in a Euclidean parameter space associated with function values f_i. It is common to assume that the data has been sampled from an unknown, possibly (piecewise) smooth function $f : \mathbb{R}^d \mapsto \mathbb{R}$ and that the sampling process might have introduced noise.

Seeking a PL approximation to f, all we can do is ask that it is close to the given data, i.e. $f(\mathbf{x}_i) \approx f_i$. Our central idea to approach the approximation problem is to generalize the notion of *lower/upper convex hull* of the point set (see Def. 3.1). Inspired by α-shapes [Edelsbrunner et al. 1983], we interpret the lower convex hull as the result of 'probing' from below using planes. Generalizing the probing to paraboloids, more points become 'reachable' (Figure 1). In Section 3 we introduce this concept formally. It turns out that the set of simplices touched by paraboloids is the lower convex hull of an appropriately lifted point set. This implies that the set of simplices, generically, is a weighted Delaunay triangulation , i.e., it forms a triangulation of (a subset) of the given points. We call this weighted Delaunay triangulation of the data points the *(lower) α-hull*. Using the analogously defined upper α-hull, we can approximate given data by a PL envelope consisting of the lower and upper α-hull, which encloses all points, progressively tighter with increasing α. Using any convex combination of the two hulls yields a PL approximation.

This view not only allows us to show several nice properties of lower α-hulls, including how they can be constructed efficiently for all possible values of α (Section 3). It also leads to an alternative approximation of the sought-after function f as a *convex-concave* decomposition or, equivalently, the difference of convex functions (Section 4): we call the lower convex hull of the lifted point set the α-lift of the data. The α-function is then (half of) the difference of the lower and upper lift. Representation in terms of convex functions is useful, as a wide variety of algorithms and data structures become available. A noteworthy concept are α/τ-lifts, which are based on leaving some data points out when constructing the lower convex hull. Another useful feature is that all algorithms can be implemented without explicitly constructing the simplicial complex, so they might be used in higher dimension as well. Convex function also have a dual representation, the so-called *Legendre transformation.* The decomposition into two convex function allows to apply this to any function. This view makes it possible to also apply PL function approximation to data that carries gradient information, sometimes called Hermite data. We show in Section 5 how this can be used to generate PL signed distance functions from surface samples with normals. Reconstructions from real world data using robust filtering are on par with the state of the art despite not being specifically designed to exploit the characteristics of the input.

Summarizing, the concepts of α-functions have the following features:

SIGGRAPH '22 Conference Proceedings, August 7–11, 2022, Vancouver, BC, Canada
© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9337-9/22/08…$15.00
https://doi.org/10.1145/3528233.3530743

- Combinatorial algorithms for representing scattered data as PL functions that deal well with noise.
- Representation of scattered data as the difference of PL convex functions enables efficient algorithms and data structures for approximation and representation that also scales to higher dimension.
- Handling of data with gradient information, including noise, resulting in a Voronoi-type reconstruction algorithms for noisy surface samples.

2 RELATED WORK

While the particular construction of α-functions is inspired by α-shapes [Edelsbrunner et al. 1983; Edelsbrunner and Mücke 1994] the motivation for coming up with convex/concave decompositions is that convolutional neural networks with ReLU activation can be interpreted in this way: such networks have been characterized as *tropical* rational polynomials [Zhang et al. 2018]. Tropical geometry is based on a min, +-algebra, so a polynomial is the minimum of linear functions and the quotient turns into a difference [Maclagan and Sturmfels 2015], and rational tropical polynomials are representations in terms of a convex and a concave PL linear function. Such decompositions have been used actively in optimization [Yuille and Rangarajan 2003], which is natural as the decompositions reduces non-convex optimization to the easier case of convex optimization [Boyd et al. 2004].

This work is an attempt at exploiting this idea for the case of scattered data interpolation and approximation [Wendland 2004] using PL functions. The motivation for restricting to PL functions is their simplicity, resulting in efficient representation and computation. The convex/concave decomposition gives rise to handling noisy data as well as data with gradients, the latter being quite unusual for PL data approximation methods. We cannot possibly discuss the many potential application areas in graphics and geometry processing beyond pointing to surveys covering the areas of curve [Ohrhallinger et al. 2021] and surface [Berger et al. 2014] reconstruction from unstructured samples.

Our approach, at least combinatorially, can be interpreted as computing with arrangements of paraboloids. Edelsbrunner and Seidel [1986] mention in passing that "To our knowledge paraboloid arrangements per se have not been studied in the literature" and then go on to show that computing arrangements of paraboloids is computationally equivalent to computing arrangements of planes so "there is really no need to do so [study them]". We exploit this computational equivalence and show that it can be very fruitful.

Weighted Delaunay Triangulations and Bistellar Flips. The lower convex hull of points lifted from $\mathbb{R}^d$ to $\mathbb{R}^{d+1}$ is a *regular* or *weighted Delaunay* triangulation of the points [Aurenhammer 1987; Aurenhammer et al. 2013; De Loera et al. 2010]. The weights w_i of weighted Delaunay triangulations are connected to the function values by $w_i = \mathbf{x}_i^\mathsf{T}\mathbf{x}_i - f_i$. The Delaunay triangulations arises from setting $w_i = 0$ (see Alexa [2020, Sec. 2] for a more detailed explanation using similar notation).

An important concept for the construction of weighted Delaunay triangulations are *bistellar flips* [Pachner 1991; Santos 2006]. A flip is defined by considering a *circuit* [De Loera et al. 2010, Sec. 2.4.1], a set c of $d+2$ points in $\mathbb{R}^d$. Every circuit has exactly two triangulations. These two triangulations are the lower and upper hull of the points lifted into $\mathbb{R}^{d+1}$ (assuming they are not affinely dependent in the lift). In fact, these two triangulations are the Delaunay and 'furthest point' Delaunay triangulation (i.e. the dual of the Euclidean furthest point Voronoi diagram [Edelsbrunner and Seidel 1986]) of the $d+2$ points. The bistellar flip changes from one triangulations to the other [De Loera et al. 2010, Sec. 4.4]. The sum of the simplices in the two triangulations is always $d+2$. For example, in $\mathbb{R}^2$, we may flip the diagonal of 4 points in convex position, the (2-2) flip. In 3 dimensions, we typically consider the (2-3) and (3-2) flips. If the points are not in convex position, one of the two triangulations is a single full simplex and the other configuration results from connecting the point inside the simplex to all vertices, resulting in $d+1$ full simplices - these are the (1-($d+1$)) and (($d+1$)-1) flips, changing the number of vertices.

With this general notion of flipping, it is possible to move between any two weighted Delaunay triangulations by a sequence of bistellar flips [De Loera et al. 2010, Thm. 5.3.7]. In fact, incremental algorithms for constructing the (weighted) Delaunay triangulation [Edelsbrunner and Shah 1992; Joe 1989] can be interpreted in this setting as first lifting all points to $+\infty$ and then moving them downwards to their desired height ($\mathbf{x}_i^\mathsf{T}\mathbf{x}_i$ in the case unweighted Delaunay) one after the other (c.f. [De Loera et al. 2010, Cor. 5.3.14]. Throughout the downwards movement, the changing lower convex hull dictates the necessary flips.

Legendre transformation. Using convex functions enables the Legendre transformation [Fenchel 1949]. In the continuous case, it maps a convex function f to another convex function defined over the domain of its gradients ∇f. Notice that the gradients are unique if f is convex. The Legendre transformation is a dual transformation, meaning if we transform twice, we get the original (convex) function back.

If $f(\mathbf{x})$ is *strictly* convex and continuous, then its Legendre transform is

$$f^*(\mathbf{y}) = \mathbf{x}^\mathsf{T}\mathbf{y} - f(\mathbf{x}), \qquad \mathbf{y} = \nabla f(\mathbf{x}). \tag{1}$$

The Legendre transformation has a meaningful definition also for PL convex functions, albeit such functions not being strictly convex [Chynoweth and Sewell 1990]. A fact not widely known in the graphics community is that the Legendre transformation is yet another way to connect the Voronoi diagram to the Delaunay triangulation (using the quadratic lift) or, more generally, weighted Delaunay triangulations and dual power diagrams [Boissonnat et al. 2010; Springborn 2008].

3 LOWER HULLS

In order to avoid unnecessary complication in the presentation, we make several assumptions on the genericity of the data – in the real world data will fail to conform to these assumptions but it is possible to (symbolically) perturb the data [Edelsbrunner and Mücke 1990] to deal with these situations. The parametric locations of the data points $\mathbf{x}_i \in \mathbb{R}^d$ are in *general position*, i.e. no more $d+1$ points are on a common sphere. Moreover, no more than $d+1$ points $(\mathbf{x}_i, f_i)$ are on a common affine function, i.e. co-planar when viewed as points in $\mathbb{R}^{d+1}$.

The *lower convex hull* of the data $\mathbf{X}, \mathbf{f} = \{(\mathbf{x}_i, f_i)\}$ may be defined as either the minimum among all convex combinations of the data points or, as we prefer here, as the maximum among the planes defining empty half-spaces:

Definition 3.1. The *lower convex hull* of the data $(\mathbf{X}, \mathbf{f})$ is

$$h_{\mathbf{X},\mathbf{f}}(\mathbf{x}) = \sup_{\mathbf{a}\in\mathbb{R}^d, b\in\mathbb{R}} \mathbf{a}^\mathsf{T}\mathbf{x} + b,\ \forall_i\ \mathbf{a}^\mathsf{T}\mathbf{x_i} + b \le f_i. \tag{2}$$

We will drop either of the two subscripts of $h_{\mathbf{X},\mathbf{f}}$ if they are irrelevant or clear from the context. Note that $h(\mathbf{x})$ is a PL function on the convex hull of the set $\{\mathbf{x}_i\}$. Under our genericity assumptions the planar pieces are simplices. A simple conceptual (albeit not practical) way to construct $h(\mathbf{x})$ is to consider each simplex, construct the affine function $\mathbf{a}^\mathsf{T}\mathbf{x} + b$ that interpolates its vertices, and check if all other points are in the upper half space. Practical algorithms avoid inspecting all simplices [Avis et al. 1997; Barber et al. 1996; Fukuda and Prodon 1996].

Our central idea is to make the condition for inclusion of a simplex less restrictive. Rather than asking that a plane through the vertices of a simplex separates the data points from negative infinite, we consider a paraboloid. Making the paraboloid concave results in being able to 'touch' points that are not on the lower convex hull. We do this by introducing the factor $-\frac{1}{2}\alpha$ for the quadratic monomial $\mathbf{x}^\mathsf{T}\mathbf{x}$. Generically, a paraboloid with fixed α interpolates d+1 points or, in other words, the vertices of a simplex. Let t identify the vertices of a simplex, then we write the paraboloid interpolating these vertices as

$$f_i = -\frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i + \mathbf{a}_t^\mathsf{T}\mathbf{x_i} + b_t,\ i \in t, \tag{3}$$

i.e. the subscript t indicating that $\mathbf{a}_t, b_t$ have been chosen appropriately. With this notation, the selection of simplices by probing with paraboloids are defined as follows:

Definition 3.2. The set of simplices comprising the *lower α-hull* of the data $(\mathbf{x}_i, f_i)$ are

$$\mathcal{T}^\alpha = \left\{t \mid -\frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i + \mathbf{a}_t^\mathsf{T}\mathbf{x_i} + b_t \le f_i, \forall i\right\} \tag{4}$$

Elementary properties. One may wonder if the set of selected simplices defines anything useful. Transforming the condition

$$-\frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i + \mathbf{b}_t^\mathsf{T}\mathbf{x}_i + c_t \le f_i \quad\Longleftrightarrow\quad \mathbf{b}_t^\mathsf{T}\mathbf{x}_i + c_t \le f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i \tag{5}$$

shows that it defines the triangulation $\mathcal{T}_\alpha$ as the lower convex hull of the 'lifted' data points $(\mathbf{x}_i, f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i)$. This means the simplices form a simplical complex - a triangulation in 2D, a tetrahedralization in 3D - of the convex hull of the $\{\mathbf{x}_i\}$. This suggests that we can define a PL surface, the lower α-hull h^α by assigning the affine function to each simplex.

Definition 3.3. Let t be a simplex that contains $\mathbf{x}$ in the parameter domain. Then the function h^α at $\mathbf{x}$ is defined as

$$h^\alpha(\mathbf{x}) = \sum_{k\in t}\mu_k f_k, \quad \text{with} \quad \mathbf{x} = \sum_{j\in t}\mu_j\mathbf{x}_j, \quad \sum_j \mu_j = 1. \tag{6}$$

The name 'lower' α-hull seems to imply that no data point is located below h^α. This is indeed the case:

Proposition 3.4. *Let $\alpha > 0$ and h^α be constructed from data $(\mathbf{x}_i, f_i)$. Then $h(\mathbf{x}_i) \le f_i$.*

Proof. If $(\mathbf{x}_i, f_i)$ is a vertex of any of the simplices in $\mathcal{T}^\alpha$ then clearly $h(\mathbf{x}_i) = f_i$. So assume this is not the case and let t be a simplex containing $\mathbf{x}_i$ in $\mathbb{R}^d$, and $\mathbf{a}_t, b_t$ defining the paraboloid interpolating its vertices. By construction of h^α, we have $-\frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i + \mathbf{a}_t^\mathsf{T}\mathbf{x_i} + b_t \le f_i$. It suffices to show the paraboloid through the vertices of t is above the plane through the vertices of t in the simplex. Note that the intersection of the paraboloid and the plane projected to the parameter domain is a sphere [Aurenhammer et al. 2013], interpolating the parameter locations of the vertices – in other words it is the circumsphere of the simplex in the parameter domain. Inside this sphere the paraboloid has larger function values than the plane. The claim follows because the simplex is contained in its circumsphere. □

Other properties of lower α-hulls can be proved elementary, but we find the most illuminating picture arises from exploiting the fact that they are weighted Delaunay triangulations and can be modified through bistellar flips (see Section 2).

Bistellar flips. Flips in Delaunay algorithms are determined by evaluating determinants [Guibas and Stolfi 1985]. The determinant amounts to computing the *signed volume* of the lifted circuit interpreted as a simplex in $\mathbb{R}^d\times\mathbb{R}$. For computing the lower alpha hull we need to consider the lifted points $(\mathbf{x}_i, f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i)$. The volume V_c of the circuit $c = (0, 1, \dots, d+1)$ depends on α and is up to constant factors

$$\begin{aligned} V_c(\alpha) &= \det\begin{pmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \dots \\ f_0 + \frac{\alpha}{2}\mathbf{x}_0^\mathsf{T}\mathbf{x}_0 & f_1 + \frac{\alpha}{2}\mathbf{x}_1^\mathsf{T}\mathbf{x}_1 & \dots \\ 1 & 1 & \dots \end{pmatrix} \\ &= \det\begin{pmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \dots \\ f_0 & f_1 & \dots \\ 1 & 1 & \dots \end{pmatrix} + \frac{\alpha}{2}\det\begin{pmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \dots \\ \mathbf{x}_0^\mathsf{T}\mathbf{x}_0 & \mathbf{x}_1^\mathsf{T}\mathbf{x}_1 & \dots \\ 1 & 1 & \dots \end{pmatrix} \\ &= V_c(0) + \frac{\alpha}{2}V_c^{\text{del}}. \end{aligned} \tag{7}$$

We see that the volume in the lift depends on the volume in the original configuration $V_c(0)$ and the volume V_c^{del} of the Delaunay configuration. A flip occurs on the circuit when the sign of the volume changes. Since V_c is a linear function of α it has exactly one zero and the critical value is

$$\alpha_c = -2\frac{V_c(0)}{V_c^{\text{del}}}. \tag{8}$$

We have the following result.

Lemma 3.5. *For the $d+2$ points in a circuit c the lower alpha hull $\mathcal{T}^\alpha$ is the Delaunay triangulation for $\alpha > \alpha_c$.*

Proof. Arrange the index set in c so that V_c^{del} is positive. If $V_c(\alpha)$ is positive, the corresponding triangulation is the Delaunay triangulation. The condition $0 < V_c(\alpha) = V_c(0) + \frac{\alpha}{2}V_c^{\text{del}}$ leads to $\alpha > \alpha_c$, as claimed. □

This local result yields characterization of the 'endpoints' of the spectrum of triangulations $\mathcal{T}^\alpha$ over varying α.

Corollary 3.6. *For given data $(\mathbf{x}_i, f_i)$ there exist constants $\alpha^\pm$ so that for all $\alpha > \alpha^+$ the lower alpha hull $\mathcal{T}^\alpha$ is a Delaunay triangulation, and for all $\alpha < \alpha^-$ the lower alpha hull $\mathcal{T}^\alpha$ is the furthest point Delaunay triangulation.*

Proof. We only discuss the case of α^+, the other case is analogous. A triangulation is Delaunay if all of its simplices are Delaunay. A simplex is not Delaunay if and only if it is part of a flippable circuit that is not Delaunay. Consider the Delaunay configuration of the points $\{\mathbf{x}_i\}$. Compute α_c for all flippable circuits and let $\hat{\alpha_c}$ be their maximum. Set $\alpha^+ = \hat{\alpha}_c + \epsilon$ for $\epsilon > 0$. Increasing α starting from $\mathcal{T}^{\alpha^+}$ cannot result in any flips. Scaling the function values has no effect on the combinatorics, so we consider the lifted points $(\mathbf{x}_i, \frac{1}{\alpha} f_i + \mathbf{x}^\mathsf{T}\mathbf{x})$, which converge to the Delaunay configuration for $\alpha \to \infty$. □

The general position assumption implies that the Delaunay triangulation is unique. If it is not, α^+ identifies one of the Delaunay triangulations.

How are the triangulations evolving as α varies in $[\alpha^-, \alpha^+]$? Notice that the furthest point Delaunay triangulation contains only the boundary vertices of the convex hull in the parameter domain, while the Delaunay triangulation contains all the points. One may expect that the number of vertices increases with α, and this is indeed the case, because flips for increasing α are Delaunay.

Corollary 3.7. *The number of vertices in $\mathcal{T}^\alpha$ is a non-decreasing function of α.*

Proof. It suffices to show that a flip occurring due to the increase of α can never remove a vertex. This follows from flips for increasing α being Delaunay flips, which either leave the number of vertices unchanged or are (1-$(d+1)$) flips. □

For $d = 1, 2$ this completely characterizes the behavior of the number of elements, as the number of edges and triangles is determined by the number of vertices for fixed boundary. Beyond two dimensions there is not much we can say about the number of the other simplices, as Delaunay flips may decrease the number of simplices.

Construction. The observations so far show that moving α from $-\infty$ to $+\infty$ creates a sequence of bistellar flips starting at the furthest point Delaunay triangulation and ending in the Delaunay triangulation of the point set. The choice of flip for a triangulation $\mathcal{T}^\alpha$ is governed by the flippable circuit with the smallest value α_c. This observation immediately leads to an algorithm for computing all different triangulations $\mathcal{T}^\alpha$, similar to algorithms for incremental Delaunay triangulation.

As it makes asymptotically no difference and Delaunay triangulation is fast in theory and practice [Amenta et al. 2007; Attali et al. 2003; Jamin et al. 2018] we start from the Delaunay triangulation. This is more convenient because decreasing α only reduces the number of vertices, avoiding point location for adding vertices, which is the most costly part of the algorithm [Amenta et al. 2003]. The algorithm is then straightforward in principle: maintain a priority queue of circuits to be flipped, prioritized based on α_c; after each flip, update α_c for the boundary facets of the circuit. An efficient implementation of identifying flippable circuits in arbitrary dimension is non-trivial, and there are established libraries for the task. For $d = 2$, (3-1) flips are identified as vertices with degree 3, and (2-2) flips are recognized as interior edges that intersect the flipped edge. These conditions extend to higher dimension in a natural way, but how to maintain efficient data structures is non-obvious. The complexity of this algorithm is similar to incremental Delaunay algorithms.

The result of the algorithm can be compactly stored as the sequence of flips together with the corresponding value α_c. This allows browsing through the different triangulations with constant time per flip, meaning time linear in the number different triangulations for accessing a triangulation based on α.

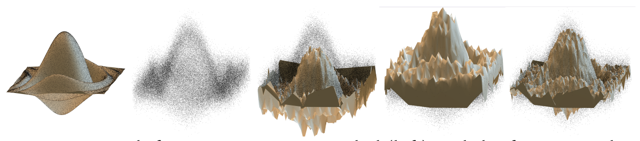


Fig. 2. A smooth function is point sampled (left) and the function values are contaminated with Gaussian noise. The lower and upper α-hull are poor reconstructions of the original data, the mid-contour (right) performs well.

4 CONVEX/CONCAVE DECOMPOSITION

We can compute upper hulls $\mathcal{U}^\alpha$ in a completely analogous fashion. In fact, we do so by computing lower α-hulls of the data $(\mathbf{x}, -f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i)$, which gives the negative of the upper hull. In other words, the upper hull is given as $-h^\alpha_{-\mathbf{f}}(\mathbf{x})$.

Assuming the data contains noise, it may be reasonable to approximate it as the mid-contour of the lower and upper PL functions, i.e. as $m^\alpha_\mathbf{f} = \frac{1}{2}(h^\alpha_\mathbf{f} + (-h^\alpha_{-\mathbf{f}}))$ (see Figure 2). It seems we could also take the mid-contour of the lifted triangulations and get the same result, because the lifting by $\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i$ cancels, similar to an observation of Biswis et al. [2022] for the mid-contour of the lifts for a weighted Delaunay triangulations and its dual power diagram. Such a representation of the mid-contour would be a *convex/concave* decomposition, because the lifts are convex. We will now analyze this idea

To make this concrete we define the lifted PL function as the lower hull of the lifted points:

Definition 4.1. Let data $(\mathbf{X}, \mathbf{f})$ be given and write $\mathbf{f}^\alpha = \{f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i)\}$ for the lifted data. Then the (lower) *α-lift* is

$$l^\alpha_{\mathbf{X},\mathbf{f}}(\mathbf{x}) = h_{\mathbf{X},\mathbf{f}^\alpha}(\mathbf{x}) = \sup_{\mathbf{a}\in\mathbb{R}^d, b\in\mathbb{R}} \mathbf{a}^\mathsf{T}\mathbf{x} + b,\ \forall_i\ \mathbf{a}^\mathsf{T}\mathbf{x_i} + b \le f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i. \tag{9}$$

Note that l^α is a convex PL function by construction. The corresponding lifted upper hull is given as $l^\alpha_{-\mathbf{f}}$. Taking the mid-contour of the two lifts yields the

$$\alpha\text{–function}: \quad f^\alpha_\mathbf{f} = \frac{1}{2}\left(l^\alpha_\mathbf{f} - l^\alpha_{-\mathbf{f}}\right) \tag{10}$$

and shows that it is represented as the difference of two convex PL functions. As mentioned, this approximation is identical to the mid-contour based on the original data, if the lower and upper α-hull have the same combinatorics.

Proposition 4.2. *For $\mathcal{T}^\alpha = \mathcal{U}^\alpha$ we have $f^\alpha_\mathbf{f} = m^\alpha_\mathbf{f}$.*

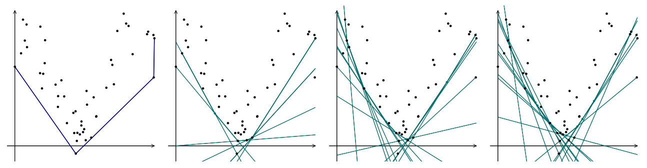


Fig. 3. The α/τ lift: the data points are equipped with confidence measures τ_i, here we assume $\tau_i = 1$. Rather than considering the lower convex hull of the (lifted) data, we consider the intersection of half spaces that are above data points so that $\sum \tau_i \leq \tau$. The illustrations shows this for $\alpha = 0$ (the original function values) and, from left to right, $\tau = \{0, 1, 2, 3\}$.

Proof. The function is determined by the function values for the vertices and the triangulation. As both triangulations are the same, it suffices to inspect the function values. Here we have

$$\begin{aligned} 2m^\alpha_{\mathbf{f}}(\mathbf{x}_i) &= l^\alpha_{\mathbf{f}}(\mathbf{x}_i) - l^\alpha_{-\mathbf{f}}(\mathbf{x}_i) \\ &= f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i - (-f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i) \\ &= f_i - (-f_i) = h^\alpha_{\mathbf{f}}(\mathbf{x}_i) - h^\alpha_{-\mathbf{f}}(\mathbf{x}_i) = 2f^\alpha_{\mathbf{f}}. \end{aligned} \tag{11}$$

□

For α large enough, every set of function values will be the Delaunay triangulation. This immediately implies the following result:

Corollary 4.3. *A PL function interpolating the data* $(\mathbf{X}, \mathbf{f})$ *defined by the Delaunay triangulation of* $\mathbf{X}$ *can be represented as the difference of two convex functions* $\mathbf{f}^\alpha$ *and* $-\mathbf{f}^\alpha$*.*

We want to stress that while every PL function has a PL convex/concave decomposition [Aleksandrov 2012], such decomposition generally induces additional vertices.

Moreover, if the combinatorics of the lower and upper α-hulls are different then, in general, averaging the lower and upper hulls is different from averaging the lower and upper α-lifts, i.e. $m^\alpha_{\mathbf{f}} \neq f^\alpha_{\mathbf{f}}$. In particular, while $m^\alpha_{\mathbf{f}}$ is guaranteed to be enclosed in the α-hulls, the α-function $f^\alpha_{\mathbf{f}}$ is not.

Why is it interesting to represent a PL function as the difference of two convex PL functions (rather than the difference of arbitrary PL functions such as the α-hulls)? A number of algorithms for approximation or, more generally, processing such data exist that require or exploit convexity of the input. Importantly, the function can be represented exactly or approximated coarsely by a set of planes. Function values can then be computed in time linear in the number of planes by taking the maximum. This is possible even in high dimensions, where generating the complete combinatorial information is impractical. We cannot give a comprehensive account of the many algorithms that would apply and highlight the potential by combining a robust redefinition of the α-lift and subsampling.

Robust α-lifts. If the data is contaminated with outliers, using hulls will make the outliers prominent features in the approximation and the mid-contour $m^\alpha_{\mathbf{f}}$ degenerates to noise (see Figure 4). We introduce a simple way to account for outliers, made possible by the convex representation. Recall that we defined the convex hull as the intersection of half-spaces, for which *all* points are on the positive side. Following Joswig et al. [2020], we generalize this idea and consider the intersection of half-spaces, for which *almost all* points are on the positive side (see Figure 3). In order to deal with *confidence* information τ_i connected to the points, we restrict to half-spaces, for which the sum of the confidence values on the negative side is below a threshold.

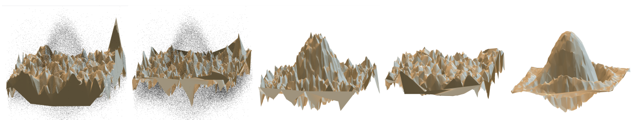


Fig. 4. Pairs of samples from the function in Figure 2 are swapped to create outliers. The mid-contour of the α-hulls fails to reconstruct the function (left). Constructing the α-function from α-lifts based on random planes (middle left) makes no difference, but enables using the α/τ-lift to filter the outliers (middle). Using subsampling on a grid for the α-lifts likewise results in a poor α-function approximation (middle right), but adding α/τ-filtering for the lifts yields a good result.

Definition 4.4. The *α/τ-lift* of the data $\mathbf{X}, \mathbf{f}$ is

$$l^{\alpha/\tau}_{\mathbf{X},\mathbf{f}}(\mathbf{x}) = \sup_{\mathbf{a}\in\mathbb{R}^d, b\in\mathbb{R}} \mathbf{a}^\mathsf{T}\mathbf{x} + b, \quad \sum_{j\in\mathcal{P}(\mathbf{a},b)} \tau_j < \tau, \tag{12}$$

where $\mathcal{P}(\mathbf{a}, b)$ is the set of points below plane $\mathbf{a}, b$, i.e. satisfying

$$\mathbf{a}^\mathsf{T}\mathbf{x_j} + b > f_j + \frac{1}{2}\alpha\mathbf{x}_j^\mathsf{T}\mathbf{x}_j. \tag{13}$$

While the α/τ-lift is, by construction, a convex PL function, unlike the hulls and lifts so far, it is not necessarily a triangulation, and its vertices are not necessarily elements of $\mathbf{X}$. This is illustrated in Figure 3 for $\tau_i = 1$. We suggest to approximate the α/τ-lift by a finite set of (random) planes. Assuming a lower bound $\tilde{\tau}$ on the τ_i at most $k = \tilde{\tau}^{-1}$ points may be outside the α/τ-hull in any direction. This means for a given gradient $\mathbf{a}$ we can compute b without additional data structures in expected $O(n + k \log k)$ time using efficient computation of the k-th smallest element and then sorting the k elements. Spatial data structures (such as k-d trees) help in reducing the linear dependence on n, however, the effect will depend on the distribution of the data and the dimension d [Samet 1990].

Sub-sampling. An alternative to random sampling in lower dimensions is sub-sampling the data points. We suggest to use *coresets* [Agarwal et al. 2005], which allow approximating a geometric structure within a factor of $1 + \epsilon$. Bentley et al. [1982] show that coresets of size $O(1/\epsilon)$ for convex hulls can be found by taking extremal values from the data sorted into a grid. This is not optimal but attractive as it leads to a simple algorithm: the locations $\mathbf{x}_i$ are sorted into a grid, and the minimal/maximal value is chosen for the lower/upper α-lift. This approach works well on data with moderate noise and can be easily combined with α/τ filtering to handle outliers. Figure 4 shows the resulting α-functions.

5 HERMITE DATA AND THE LEGENDRE TRANSFORM

In many applications, the data $(\mathbf{x}_i, f_i)$ also carry information about gradients $\mathbf{g}_i$. One may think of this data as small (hyper-)planes around the sample points. It is uncommon to approximate such data with PL functions – the data will generally be contradictory (i.e., not integrable) for $d > 1$.

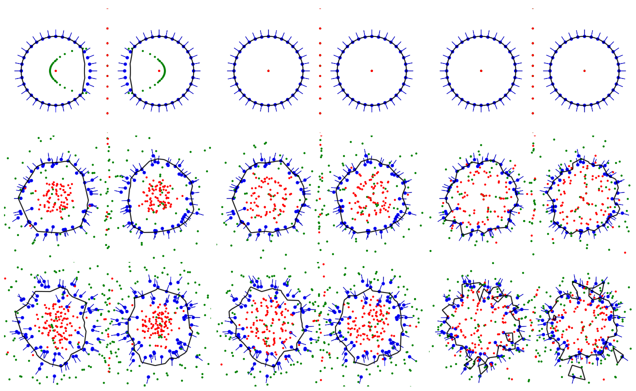


Fig. 5. Approximating a signed distance function from point samples with normals (blue). Points are sampled from two circles with varying Gaussian noise in the positions and normals (rows). Vertices of the lower and upper α-lift are shown in red and green. The reconstructed curve (black) is the zero level-set of the α-function for $\alpha = \frac{1}{2}, 1, 2$. (left to right).

One prime example in geometry processing is the reconstruction of a surface from sample points. If we want to generate a (PL) signed distance function from the data so that the surface is represented as a level set, the samples all provide the same function value. This shows not only that an important part of the information about the surface is encoded in the given gradients, but also that additional vertices are needed supporting a PL function that approximates the signed distance function.

If the sampled function were convex, we could interpret the samples as small planes, and compute their Legendre transformation. As before, we achieve convexity by lifting the (unknown) signed distance function f by $\frac{1}{2}\alpha\mathbf{x}^\mathsf{T}\mathbf{x}$. Notice that the gradient of the lift at $\mathbf{x}_i$ is $\alpha\mathbf{x}_i$ so the lifted samples are

$$(\mathbf{x}_i, f_i, \mathbf{g}_i) \xrightarrow{\alpha-\text{lift}} \left(\mathbf{x}_i, f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i, \mathbf{g}_i + \alpha\mathbf{x}_i\right). \tag{14}$$

The Legendre transformation applied to the samples yields

$$f_i^{\alpha^*}(\mathbf{y}_i) = f_i^*(\mathbf{g}_i+\alpha\mathbf{x}) = \mathbf{x}_i^\mathsf{T}\mathbf{y}_i - f_i - \frac{\alpha}{2}\mathbf{x}_i^\mathsf{T}\mathbf{x}_i, \qquad \mathbf{y}_i = \mathbf{g}_i+\alpha\mathbf{x}_i. \tag{15}$$

We use the variable $\mathbf{y}$ to denote points in the Legendre transform domain, i.e. the domain of the gradients of the lifted function. Plugging in $\mathbf{y}_i$ provides the samples

$$(\mathbf{y}_i, f_i^*) = \left(\mathbf{g}_i + \alpha\mathbf{x}_i, \mathbf{x}_i^\mathsf{T}\mathbf{g}_i - f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i\right). \tag{16}$$

Similar to the situation before, we can always choose α large enough to make sure the transformed samples are in convex position. If α is chosen smaller, we can then still take the lower convex hull, similar to the motivation for α-hulls, ignoring the information contained in some of the points. The lower hull is in any case a convex PL function, to which we can apply the Legendre transformation to generate an approximate PL function that respects the function values f_i as well as the gradients $\mathbf{g}_i$.

Computing function values. Transforming the weighted Delaunay triangulation of the Legendre data yields a power diagram that approximates the α-lift $l^\alpha_{\mathbf{X},\mathbf{f}}$ of the original data while respecting the gradients. Transforming both lower and upper hull allows approximating the signed distance function as an α-function, i.e., the difference of the two lifts, which is our goal given the data.

It is not necessary to explicitly compute the weighted Delaunay triangulation or the power diagram for evaluating the function. Consider what evaluation of l^α at $\mathbf{x}$ amounts to in the Legendre transform domain. Notice that all possible function values l at $\mathbf{x}$ form the plane $p^*(\mathbf{y}) = \mathbf{x}^\mathsf{T}\mathbf{y} - l$ in the Legendre transform domain. This means we can evaluate l^α at $\mathbf{x}$ by making the plane $p^*(\mathbf{y})$ tangent to $f^*(\mathbf{y})$. Generically, we have to identify the vertex $(\mathbf{y}_i, f_i^*)$ that minimizes $p^*(\mathbf{y}_i)$. Since $\mathbf{x}$ is fixed we have to maximize l:

$$l^\alpha(\mathbf{x}) = \arg\max_i \mathbf{x}^\mathsf{T}\mathbf{y}_i - f_i^*. \tag{17}$$

It is instructive to inspect what this maximization means in terms of the original data. Plugging in how the points $\mathbf{y}_i, f_i^*$ have been constructed reveals

$$\begin{aligned} l^\alpha(\mathbf{x}) &= \arg\max_i \ \mathbf{x}^\mathsf{T}(\mathbf{g}_i + \alpha\mathbf{x}_i) - \left(\mathbf{x}_i^\mathsf{T}\mathbf{g}_i - f_i + \frac{\alpha}{2}\mathbf{x}_i^\mathsf{T}\mathbf{x}_i\right) \\ &= \arg\max_i \ (\mathbf{x}-\mathbf{x}_i)^\mathsf{T}\mathbf{g}_i + f_i \ + \ \frac{\alpha}{2}(2\mathbf{x}-\mathbf{x}_i)^\mathsf{T}\mathbf{x}_i. \end{aligned} \tag{18}$$

This shows that $l^\alpha(\mathbf{x})$ results from evaluating the plane through data point $(\mathbf{x}_i, f_i)$ with gradient $\mathbf{g}_i$ at $\mathbf{x}$, the evaluation point in the original space. Lifting adds another plane of the form $(2\mathbf{x}-\mathbf{x}_i)^\mathsf{T}\mathbf{x}_i$. Evaluating this plane at $\mathbf{x} = \mathbf{x}_i$ yields $\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i$, the lifting for point i.

This means, the value $l^\alpha(\mathbf{x})$ depends on only on a single data point. Which point is selected? This depends on the function value of the plane given by $\mathbf{x}_i, f_i$, as well as $\mathbf{g}_i$ and the term $(2\mathbf{x}-\mathbf{x}_i)^\mathsf{T}\mathbf{x}_i$, both of which are maximized, with the second part weighted by $\frac{\alpha}{2}$. The first part selects the plane with maximal height at $\mathbf{x}$. If the original data was convex, this would be the correct assignment. Inspecting the second part, we find

$$\begin{aligned} \arg\max_i \ (2\mathbf{x}-\mathbf{x}_i)^\mathsf{T}\mathbf{x}_i &= \arg\max_i \ -(\mathbf{x}-\mathbf{x}_i)^\mathsf{T}(\mathbf{x}-\mathbf{x}_i) + \mathbf{x}^\mathsf{T}\mathbf{x} \\ &= \arg\min_i \|\mathbf{x}-\mathbf{x}_i\|. \end{aligned} \tag{19}$$

In other words, the term weighted by $\frac{\alpha}{2}$ minimizes the distance of $\mathbf{x}_i$ to the evaluation point $\mathbf{x}$. This connects to approaches using the Voronoi diagram of the surface samples for reconstruction. Hoppe's approach [1992] is recovered explicitly by choosing α large enough and then recovering $f(\mathbf{x})$ from the maximization as $f = f^\alpha - \alpha(2\mathbf{x}-\mathbf{x}_i)^\mathsf{T}\mathbf{x}_i$, i.e. by only considering the plane through the point $\mathbf{x}_i$. Doing this to remove the effect of lifting, however, gives up continuity of f. Using the α-function based on the lower and upper α lifts guarantees a continuous PL approximation of the signed distance field.

The overall evaluation procedure leads to a simple algorithm for approximating the signed distance function based on surface samples with gradient information, summarized in Alg. 1. It is crucial to use spatial data structures for computing the Legendre sample that maximizes Eq. 17, lines marked OPT in Alg. 1. While k-d trees [Friedman et al. 1977] work generally well in small dimension, we have found that for $d = 3$ computing the convex hull explicitly and then 'walking the triangulation' [Devillers et al. 2001; Mücke et al. 1999] is more efficient. For using the α/τ-lift, the algorithm needs to be extended to first compute the top $1/\min \tau_i$ values and then sort them to find the index according to the τ cutoff. This extension works

ALGORITHM 1: Computing the α-function from Hermite data

```
Preprocess lift($\alpha$, Positions $\mathbf{x}_i$, function values $f_i$, gradients $\mathbf{g}_i$)
    for $i \in \{0, \dots, n-1\}$ do
        $(\mathbf{y}_i^+, f_i^{*+}) \leftarrow (\mathbf{g}_i + \alpha\mathbf{x}_i, \mathbf{x}_i^\mathsf{T}\mathbf{g}_i - f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i)$
        $(\mathbf{y}_i^-, f_i^{*-}) \leftarrow (-\mathbf{g}_i + \alpha\mathbf{x}_i, -\mathbf{x}_i^\mathsf{T}\mathbf{g}_i + f_i + \frac{1}{2}\alpha\mathbf{x}_i^\mathsf{T}\mathbf{x}_i)$

Function alpha(x)
    Data: Legendre samples computed in preprocess
OPT $l^+ \leftarrow \arg\max_i \mathbf{x}^\mathsf{T}\mathbf{y}_i^+ - f_i^{*+}$
OPT $l^- \leftarrow \arg\max_i \mathbf{x}^\mathsf{T}\mathbf{y}_i^- - f_i^{*-}$
    return $\frac{1}{2}(l^+ - l^-)$
```

well with the k-d trees and the combinatorial data of the convex hull.

Results. We have used this algorithm to approximate the PL signed distance function generated from the raw samples of the Stanford Bunny. We evaluate the function on a grid and then use Marching Cubes [Lorensen and Cline 1987] to extract a surface. The data contains unreliable samples, particularly the normals degrade in quality towards the boundary of the scans. Kazhdan et al. [2006, Fig. 4] show how this leads to artifacts in many surface reconstruction methods. We use confidence values [Curless and Levoy 1996; Turk and Levoy 1994] for an α/τ-lift to increase robustness. Figure 6, left column shows the reconstruction without using the robust lift, the right column with α/τ-lifting (for two values of α and $\tau = 1$). The result for $\alpha = 64$ appears to match the state of the art Poisson Surface Reconstruction [Kazhdan et al. 2006], while our approach has not been designed for this task. It certainly outperforms other combinatorial approaches [Amenta and Bern 1998; Amenta et al. 2001; Bernardini et al. 1999; Dey and Goswami 2003].

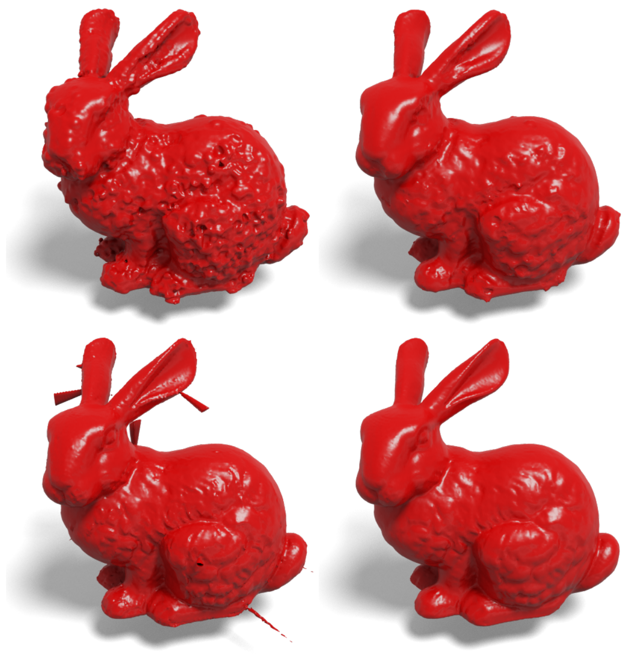
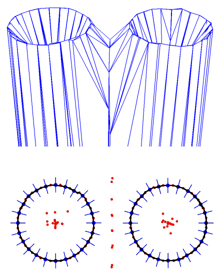


Fig. 6. The Stanford Bunny reconstructed from the raw range scans based on the α-functions for $\alpha = 16, 64$ (top, bottom). Left column is based on the standard α-lift, right column uses the robust α/τ-lift with $\tau = 1$ and the vertex-wise confidence values [Curless and Levoy 1996; Turk and Levoy 1994] for the raw samples as τ_i.

6 DISCUSSION

We have presented the concept of α-functions for PL linear approximation of scattered data, possibly carrying gradient information. The main ingredient is a convex/concave decomposition that enables exploiting a range of existing data structures, algorithms for efficient handling of large amounts of data, data in high dimension, and dealing with noise and outliers. Handling data with gradients via the Lagendre transformation, while perhaps conceptually non-trivial, ultimately leads to a simple algorithm. Despite its simplicity, for the well-researched task of surface reconstruction from 3D samples it provides competitive results.



We could exploit the possibility to work with noisy inputs and explicitly specify contradicting gradients for the purpose of modeling functions with discontinuous derivatives, for example to model *unsigned* distance fields. Each surface sample turns into two copies, with identical position and opposing gradients. The inset shows the resulting lower α-hull (upper image, upside down) for the samples shown below. The curve implied by the samples is no longer encoded as a level set. We argue that it is now given as a *ravine* or *valley* [Belyaev et al. 1997], the union of edge paths connecting saddles to local minima (lower image in the inset), and leave algorithms for its extraction as future work.

There are many ideas and avenues that were left for follow-up reports, partly addressing several limitations. For one, α is currently handled as global parameter. If the data (discounting noise) contains large gradients, then a large value α is needed to represent all features faithfully. Approximating the α-lift using techniques for PL surfaces leads to approximating mostly the paraboloid and not the data. This problem can be addressed with multi-level or hierarchical representations, similar to how convolutional networks model the function. In some applications the value of the PL approximation can only be exploited if a representation based on vertices and incident simplices has been generated. As α-functions are the difference of simplicial meshes, an overlay is required. While this problem is well understood in principle in computational geometry [Amato et al. 1995; Edelsbrunner and Seidel 1986], efficient and robust implementations exist only in 2D [Wein et al. 2021]. Likewise, efficient computational tools for flipping among weighted Delaunay triangulations in dimension $d > 3$ are missing as well.

We have not yet reaped the benefit of topological filtering and persistence enabled through the parameter α. Similar to α-shapes, changes to the topology of the graph of critical points of the function can be tracked locally [Edelsbrunner et al. 2002]. This allows computing the persistence of vertices in the sample set, with a variety of applications in data analysis [Carlsson et al. 2005].

ACKNOWLEDGMENTS

The author would like to thank Ugo Finnendahl, Max Kohlbrenner, Boris Springborn, and the anonymous reviewers for valuable comments; and Herbert Edelsbrunner for consulting on the terminology. The work has been supported by the Deutsche Forschungsgemeinschaft (EXC-2046/1 – MATH+).

REFERENCES

Pankaj K Agarwal, Sariel Har-Peled, et al. 2005. Geometric approximation via coresets. *Combinatorial and computational geometry* 52, 1-30 (2005), 3.

Aleksandr Danilovich Aleksandrov. 2012. On the surfaces representable as difference of convex functions. *Sibirskie Elektronnye Matematicheskie Izvestiia* 9 (2012), 360–3376. English translation of original 1949 article..

Marc Alexa. 2020. Conforming Weighted Delaunay Triangulations. *ACM Trans. Graph.* 39, 6, Article 248 (nov 2020), 16 pages. https://doi.org/10.1145/3414685.3417776

Nancy M. Amato, Michael T. Goodrich, and Edgar A. Ramos. 1995. Computing Faces in Segment and Simplex Arrangements. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing* (Las Vegas, Nevada, USA) *(STOC '95)*. Association for Computing Machinery, New York, NY, USA, 672–682. https://doi.org/10.1145/225058.225285

Nina Amenta, Dominique Attali, and Olivier Devillers. 2007. Complexity of Delaunay Triangulation for Points on Lower-Dimensional Polyhedra. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, USA, 1106?1113.

Nina Amenta and Marshall Bern. 1998. Surface Reconstruction by Voronoi Filtering. In *Proceedings of the Fourteenth Annual Symposium on Computational Geometry (SCG '98)*. Association for Computing Machinery, New York, NY, USA, 39–48. https://doi.org/10.1145/276884.276889

Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. 2001. The Power Crust. In *Proceedings of the Sixth ACM Symposium on Solid Modeling and Applications (SMA '01)*. Association for Computing Machinery, New York, NY, USA, 249–266. https://doi.org/10.1145/376957.376986

Nina Amenta, Sunghee Choi, and Günter Rote. 2003. Incremental Constructions Con BRIO. In *Proceedings of the Nineteenth Annual Symposium on Computational Geometry* (San Diego, California, USA) *(SCG '03)*. Association for Computing Machinery, New York, NY, USA, 211–219. https://doi.org/10.1145/777792.777824

Dominique Attali, Jean-Daniel Boissonnat, and André Lieutier. 2003. Complexity of the Delaunay Triangulation of Points on Surfaces the Smooth Case. In *Proceedings of the Nineteenth Annual Symposium on Computational Geometry*. Association for Computing Machinery, New York, NY, USA, 201–210. https://doi.org/10.1145/777792.777823

Franz Aurenhammer. 1987. Power diagrams: properties, algorithms and applications. *SIAM J. Comput.* 16, 1 (1987), 78–96.

Franz Aurenhammer, Rolf Klein, and Der-Tsai Lee. 2013. *Voronoi Diagrams and Delaunay Triangulations*. WORLD SCIENTIFIC. https://doi.org/10.1142/8685

David Avis, David Bremner, and Raimund Seidel. 1997. How good are convex hull algorithms? *Computational Geometry* 7, 5 (1997), 265–301. https://doi.org/10.1016/S0925-7721(96)00023-5 11th ACM Symposium on Computational Geometry.

C. Bradford Barber, David P. Dobkin, and Hannu Huhdanpaa. 1996. The Quickhull Algorithm for Convex Hulls. *ACM Trans. Math. Softw.* 22, 4 (dec 1996), 469–483. https://doi.org/10.1145/235815.235821

A. G. Belyaev, E. V. Anoshkina, and T. L. Kunii. 1997. *Ridges, Ravines and Singularities*. Springer Japan, Tokyo, 375–383. https://doi.org/10.1007/978-4-431-66956-2_18

Jon Louis Bentley, Franco P. Preparata, and Mark G. Faust. 1982. Approximation Algorithms for Convex Hulls. *Commun. ACM* 25, 1 (jan 1982), 64–68. https://doi.org/10.1145/358315.358392

Matthew Berger, Andrea Tagliasacchi, Lee M. Seversky, Pierre Alliez, Joshua A. Levine, Andrei Sharf, and Claudio T. Silva. 2014. State of the Art in Surface Reconstruction from Point Clouds. In *Eurographics 2014 - State of the Art Reports*, Sylvain Lefebvre and Michela Spagnuolo (Eds.). The Eurographics Association, 161–185. https://doi.org/10.2312/egst.20141040

Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. 1999. The Ball-Pivoting Algorithm for Surface Reconstruction. *IEEE Transactions on Visualization and Computer Graphics* 5, 4 (oct 1999), 349–359. https://doi.org/10.1109/2945.817351

Ranita Biswas, Sebastiano Cultrera di Montesano, Herbert Edelsbrunner, and Morteza Saghafian. 2022. Continuous and Discrete Radius Functions on Voronoi Tessellations and Delaunay Mosaics. *Discrete Comput. Geom.* 67, 3 (apr 2022), 811–842. https://doi.org/10.1007/s00454-022-00371-2

Jean-Daniel Boissonnat, Frank Nielsen, and Richard Nock. 2010. Bregman Voronoi Diagrams. *Discrete & Computational Geometry* 44, 2 (2010), 281–307. https://doi.org/10.1007/s00454-010-9256-1

S. Boyd, S.P. Boyd, L. Vandenberghe, and Cambridge University Press. 2004. *Convex Optimization.* Number pt. 1 in Berichte über verteilte messysteme. Cambridge University Press. https://books.google.de/books?id=mYm0bLd3fcoC

Gunnar Carlsson, Afra Zomorodian, Anne Collins, and Leonidas J. Guibas. 2005. Persistence Barcodes for Shapes. *International Journal of Shape Modeling* 11, 02 (2005), 149–187. https://doi.org/10.1142/S0218654305000761

S. Chynoweth and M. J. Sewell. 1990. Mesh Duality and Legendre Duality. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences* 428, 1875 (1990), 351–377. http://www.jstor.org/stable/51804

Brian Curless and Marc Levoy. 1996. A Volumetric Method for Building Complex Models from Range Images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*. Association for Computing Machinery, New York, NY, USA, 303–312. https://doi.org/10.1145/237170.237269

Jesus A. De Loera, Jorg Rambau, and Francisco Santos. 2010. *Triangulations: Structures for Algorithms and Applications* (1st ed.). Springer Publishing Company, Incorporated.

Olivier Devillers, Sylvain Pion, and Monique Teillaud. 2001. *Walking in a triangulation*. Technical Report RR-4120. INRIA.

Tamal K. Dey and Samrat Goswami. 2003. Tight Cocone: A Water-Tight Surface Reconstructor. In *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications* (Seattle, Washington, USA) *(SM '03)*. Association for Computing Machinery, New York, NY, USA, 127–134. https://doi.org/10.1145/781606.781627

Herbert Edelsbrunner, David Kirkpatrick, and Raimund Seidel. 1983. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory* 29, 4 (1983), 551–559. https://doi.org/10.1109/TIT.1983.1056714

Herbert Edelsbrunner, David Letscher, and Afra Zomorodian. 2002. Topological Persistence and Simplification. *Discrete & Computational Geometry* 28, 4 (2002), 511–533. https://doi.org/10.1007/s00454-002-2885-2

Herbert Edelsbrunner and Ernst Peter Mücke. 1990. Simulation of Simplicity: A Technique to Cope with Degenerate Cases in Geometric Algorithms. *ACM Trans. Graph.* 9, 1 (jan 1990), 66–104. https://doi.org/10.1145/77635.77639

Herbert Edelsbrunner and Ernst P. Mücke. 1994. Three-Dimensional Alpha Shapes. *ACM Trans. Graph.* 13, 1 (jan 1994), 43–72. https://doi.org/10.1145/174462.156635

Herbert Edelsbrunner and Raimund Seidel. 1986. Voronoi diagrams and arrangements. *Discrete & Computational Geometry* 1, 1 (1986), 25–44. https://doi.org/10.1007/BF02187681

Herbert Edelsbrunner and Nimish R. Shah. 1992. Incremental Topological Flipping Works for Regular Triangulations. In *Proceedings of the Eighth Annual Symposium on Computational Geometry (SCG '92)*. ACM, New York, NY, USA, 43–52. https://doi.org/10.1145/142675.142688

W. Fenchel. 1949. On Conjugate Convex Functions. *Canadian Journal of Mathematics* 1, 1 (1949), 73–77. https://doi.org/10.4153/CJM-1949-007-x

Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. 1977. An Algorithm for Finding Best Matches in Logarithmic Expected Time. *ACM Trans. Math. Softw.* 3, 3 (sep 1977), 209–226. https://doi.org/10.1145/355744.355745

Komei Fukuda and Alain Prodon. 1996. Double description method revisited. In *Combinatorics and Computer Science*, Michel Deza, Reinhardt Euler, and Ioannis Manoussakis (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 91–111.

Leonidas Guibas and Jorge Stolfi. 1985. Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi. *ACM Trans. Graph.* 4, 2 (apr 1985), 74–123. https://doi.org/10.1145/282918.282923

Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. 1992. Surface Reconstruction from Unorganized Points. *SIGGRAPH Comput. Graph.* 26, 2 (jul 1992), 71–78. https://doi.org/10.1145/142920.134011

Clément Jamin, Sylvain Pion, and Monique Teillaud. 2018. 3D Triangulations. In *CGAL User and Reference Manual* (4.13 ed.). CGAL Editorial Board.

Barry Joe. 1989. Three-Dimensional Triangulations from Local Transformations. *SIAM J. Sci. Stat. Comput.* 10, 4 (July 1989), 718–741. https://doi.org/10.1137/0910044

Michael Joswig, Marek Kaluba, and Lukas Ruff. 2020. Geometric Disentanglement by Random Convex Polytopes. https://doi.org/10.48550/ARXIV.2009.13987

Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. 2006. Poisson Surface Reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing* (Cagliari, Sardinia, Italy) *(SGP '06)*. Eurographics Association, Goslar, DEU, 61–70.

William E. Lorensen and Harvey E. Cline. 1987. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *SIGGRAPH Comput. Graph.* 21, 4 (aug 1987), 163–169. https://doi.org/10.1145/37402.37422

Diane Maclagan and Bernd Sturmfels. 2015. *Introduction to Tropical Geometry*. Graduate Studies in Mathematics, Vol. 161. American Mathematical Society, Providence, RI. vii+359 pages.

Ernst P. Mücke, Isaac Saias, and Binhai Zhu. 1999. Fast randomized point location without preprocessing in two- and three-dimensional Delaunay triangulations. *Computational Geometry* 12, 1 (1999), 63 – 83. https://doi.org/10.1016/S0925-7721(98)00035-2

Stefan Ohrhallinger, Jiju Peethambaran, Amal Dev Parakkat, Tamal K Dey, and Ramanathan Muthuganapathy. 2021. 2D Points Curve Reconstruction Survey and

Benchmark. *Computer Graphics Forum* 1 (March 2021), 1–1. https://www.cg.tuwien.ac.at/research/publications/2021/ohrhallinger-2021-egs/

Udo Pachner. 1991. P.L. Homeomorphic Manifolds are Equivalent by Elementary Shellings. *European Journal of Combinatorics* 12, 2 (1991), 129–145. https://doi.org/10.1016/S0195-6698(13)80080-7

H. Samet. 1990. *The Design and Analysis of Spatial Data Structures.* Addison-Wesley.

Francisco Santos. 2006. Geometric bistellar flips. The setting, the context and a construction. (2006). https://doi.org/10.48550/ARXIV.MATH/0601746

Boris A. Springborn. 2008. A variational principle for weighted Delaunay triangulations and hyperideal polyhedra. *Journal of Differential Geometry* 78, 2 (2008), 333 – 367. https://doi.org/10.4310/jdg/1203000270

Greg Turk and Marc Levoy. 1994. Zippered Polygon Meshes from Range Images. In *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '94).* Association for Computing Machinery, New York, NY, USA, 311–318. https://doi.org/10.1145/192161.192241

Ron Wein, Eric Berberich, Efi Fogel, Dan Halperin, Michael Hemmer, Oren Salzman, and Baruch Zukerman. 2021. 2D Arrangements. In *CGAL User and Reference Manual* (5.3.1 ed.). CGAL Editorial Board. https://doc.cgal.org/5.3.1/Manual/packages.html#PkgArrangementOnSurface2

Holger Wendland. 2004. *Scattered Data Approximation.* Cambridge University Press. https://doi.org/10.1017/CBO9780511617539

A. L. Yuille and Anand Rangarajan. 2003. The Concave-Convex Procedure. *Neural Comput.* 15, 4 (apr 2003), 915–936. https://doi.org/10.1162/08997660360581958

Liwen Zhang, Gregory Naitzat, and Lek-Heng Lim. 2018. Tropical Geometry of Deep Neural Networks. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 5824–5832. https://proceedings.mlr.press/v80/zhang18i.html