Computing Minimum Complexity 1D Curve Simplifications under the Fréchet Distance

On Computing 1D Curve Simplifications of Minimum Complexity and Fréchet Distance

Thijs van der Horst[1,2] and Tim Ophelders[1,2]

1 **Department of Information and Computing Sciences, Utrecht University, the Netherlands**
{t.w.j.vanderhorst|t.a.e.ophelders}@uu.nl

2 **Department of Mathematics and Computer Science, TU Eindhoven, the Netherlands**

Abstract

We consider the problem of simplifying curves in one dimension under the Fréchet distance. In particular, we consider the *minimum complexity* and *minimum error* simplifications. We present a continuous one-parameter family of simplifications for curves in one dimension, that contains both these simplifications. We can in linear time build a data structure that can be queried for this simplification at any parameter, and it will answer the query in linear output-sensitive time.

1 Introduction

Curve simplification is a widely studied topic in computational geometry, due to its applications in, for example, computer graphics. The main idea behind curve simplification is often to reduce the size of the curve, without affecting the overall shape of the curve too much. There are other types of simplification as well, such as computing a curve of some special class of curves that sufficiently resembles the original curve. For example, like plane graphs are a special class of graphs, plane (or simple) curves are a special class of curves. In this work we focus on the former type of simplification, which reduces the complexity of a curve.

Let P be a curve with n vertices. There are many variations for simplifying P into lower-complexity curves. These range from using different similarity measures, such as the *Hausdorff distance* or *Fréchet distance*, to constraining the shape of simplifications P' of P, for example by restricting vertices of P' to be vertices of P as well. Bereg *et al.* [1] give algorithms for simplifying a polygonal curve in $\mathbb{R}^3$ to one with the minimum number of vertices, where the discrete Fréchet distance is used to measure the similarity between the original curve and its simplification. If the vertices of the simplification are restricted to be vertices of the original curve, their algorithm runs in $O(n^2)$ time. If there are no restrictions, their algorithm runs in $O(n \log n)$ time instead. Under the continuous Fréchet distance in general dimensions, Bringmann and Chaudhury give an $O(n^3)$ time algorithm for the case where vertices are restricted to vertices of P, and give a matching conditional lower bound. Under the Hausdorff distance, van Kreveld *et al.* [7] show that the problem is in fact NP-hard if vertices are again restricted. The problem remains NP-hard in the unrestricted case [6].

Considered simplifications. In this work we study curve simplification in one dimension under the Fréchet distance, without restrictions on the vertices. We consider computing two types of simplifications: *min-#* simplifications and *closest k-curve* simplifications. A *min-# ε-simplification* of P is a curve P' within Fréchet distance ε of P and the minimum number of vertices. A *closest k-curve* of P is a curve P' with at most k vertices and the minimum Fréchet distance to P.

In one dimension, a linear-time algorithm for computing a *slightly* suboptimal min-# simplification is known due to Driemel *et al.* [3]. Their simplification takes the form of a *signature*, which uses only vertices of the original curve. Signatures have at most two vertices more than the minimum number. The class of signatures also contains a 2-approximation for the closest k-curve, in that it contains a curve with k vertices that is at most twice as far as the closest k-curve. Driemel *et al.* [3] give an $O(k \log k)$ time algorithm for computing such a curve, after $O(n \log n)$ time preprocessing.

39th European Workshop on Computational Geometry, Barcelona, Spain, March 29–31, 2023.
This is an extended abstract of a presentation given at EuroCG'23. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

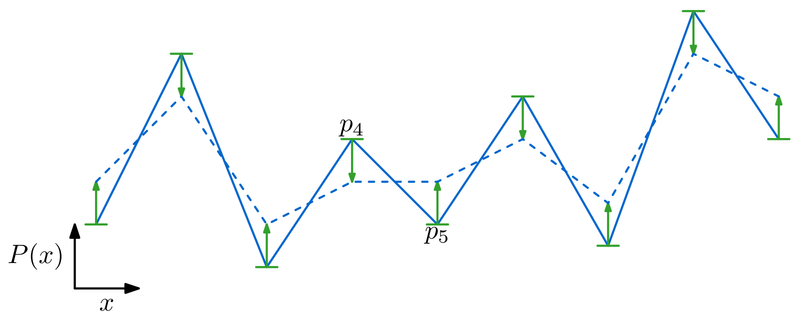

Figure 1 An illustration of smoothings. The curve P (non-dashed) is drawn as a plot of the underlying function for clarity. The minimum edge length of P is realized by $\overline{p_i p_{i+1}}$ The dashed curve is the result of smoothing. The vertices p_i and p_{i+1} have become degenerate and are not considered vertices in the smoothing.

Results and organization. In Section 2 we present *smoothings*, a method of curve simplification for curves in one dimension that is based on *truncated smoothings* for Reeb graphs [2]. We show that the ε-*smoothing* P^ε of a curve P is a min-# ε-simplification of P. We further show that for every positive integer k, there is a smoothing of P with at most k vertices that is a closest k-curve for P. In Section 3 we give a data structure for computing P^ε for any $\varepsilon \geq 0$. After $O(n)$ time preprocessing, we can compute P^ε in $O(k)$ time, where k is the complexity of P^ε. This data structure is extended to our main contributions: a data structure for constructing min-# and closest k-curve simplifications in $O(k)$ time.

Preliminaries. A (polygonal) n-*curve* is a piecewise-linear function $P\colon [0,1] \to \mathbb{R}^d$ connecting a sequence $p_1, \dots, p_n$ of d-dimensional points, which we refer to as *vertices*. A vertex p_i is *degenerate* if $2 \leq i \leq n-1$ and $p_i \in \overline{p_{i-1}p_{i+1}}$. An *edge* of P is a directed line segment connecting consecutive vertices p_i, p_{i+1}.

A *reparameterization* is a non-decreasing, continuous surjection $f\colon [0,1] \to [0,1]$ where $f(0) = 0$ and $f(1) = 1$. Two reparameterizations f and g describe a *matching* (f, g) between two curves P and Q, where $P(f(t))$ is matched with $Q(g(t))$. Given a norm $\|\cdot\|$, a matching (f, g) between P and Q is said to have *cost* $\max_t \|P(f(t)) - Q(g(t))\|$. The (continuous) Fréchet matching between P and Q is the minimum cost over all matchings.

2 Smoothings

Throughout this work we consider a polygonal n-curve P in one dimension, without degenerate vertices. In this section we present the notion of *smoothings* of P and show that among these smoothings are both min-# and closest k-curve simplifications of P.

Let $\varepsilon \geq 0$ be at most half the minimum edge length of P. The ε-*smoothing* P^ε of P is the curve obtained by truncating every edge of P by ε on either side and removing any degenerate vertex that is created. See Figure 1 for an example. For technical reasons, if vertex p_2 or p_{n-1} becomes degenerate, we remove p_1 or p_n instead of p_2 or p_{n-1}. This ensures that local minima (resp. maxima) on P^ε correspond to local minima (resp. maxima) on P. We extend the smoothing definition to all non-negative values $\varepsilon \geq 0$ by recursively defining the ε-smoothing of P for ε greater than half the minimum edge length ε' of P to be the $(\varepsilon - \varepsilon')$-smoothing of $P^{\varepsilon'}$ if $\varepsilon' > 0$ (that is, if P has at least one edge), and simply as P otherwise.

▶ **Theorem 1.** *The Fréchet distance between P and its ε-smoothing is at most ε.*

Proof. Let $\varepsilon \geq 0$. If ε is at most half the minimum edge length of P, then there is a natural matching between P and P^ε induced by the truncating operation performed for the smoothing. See

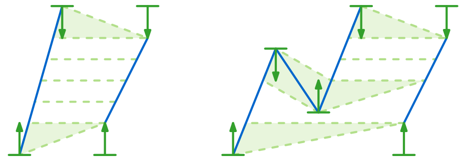

Figure 2 The matching induced by smoothings. (left) Smoothing (truncating) a single edge. Dashed segments indicate point to point matchings, dashed areas indicate subsegments matching to a single point. (right) Smoothing a more complex curve by half its minimum edge length.

Figure 2 for an illustration of this matching. This matching trivially has cost at most ε, since points are moved by distance at most ε during truncation. By the triangle inequality and the recursive definition of smoothings, it follows that $d_F(P, P^\varepsilon) \leq \varepsilon$. ◀

We proceed to show that the ε-smoothing of P is a min-# ε-simplification of P. An important consequence is that certain smoothings are closest k-curves as well for P.

▶ **Theorem 2.** *Let P be a curve in one dimension and let $\varepsilon \geq 0$. The ε-smoothing P^ε of P is a min-# ε-simplification of P.*

Proof. Let $p_1^\varepsilon, \dots, p_k^\varepsilon$ be the vertices of P^ε. For every p_j^ε there is a vertex p_{i_j} of P with value $p_j^\varepsilon - \varepsilon$ if p_j^ε is a local minimum and $p_j^\varepsilon + \varepsilon$ if p_j^ε is a local maximum. Let Q be a polygonal curve within Fréchet distance ε of P. Let $\phi = (f, g)$ be a matching between P and Q of cost at most ε. There is a sequence of values $0 \leq x_1 \leq \cdots \leq x_m \leq 1$ such that ϕ matches p_{i_j} to $Q(x_j)$ for all j. We argue that the edges of Q containing the points $Q(x_j)$ contain at least k different vertices.

Let p_j^ε be a local minimum of P^ε. Then $p_{i_j} = p_j^\varepsilon - \varepsilon$. Therefore $Q(x_j) \leq p_{i_j} + \varepsilon = p_j^\varepsilon$. The edge containing $Q(x_j)$ hence has a vertex with value at most p_j^ε. By a symmetric argument, for every local maximum p_j^ε of P^ε the edge containing $Q(x_j)$ has a vertex with value at least p_j^ε. Consecutive vertices are unique, as P^ε has no degenerate vertices. As the vertices are ordered along Q, this implies that the above vertices are all unique. Hence Q has at least k vertices. ◀

▶ **Theorem 3.** *Let P be a curve in one dimension and let $k \geq 1$ be an integer. Let $\varepsilon \geq 0$ be the smallest value for which P^ε has at most k vertices. Then P^ε is a closest k-curve for P.*

Proof. Let Q be a curve with at most k vertices. Let $\varepsilon' = d_F(P, Q)$. By Theorem 2, the ε'-smoothing of P has at most k vertices. Thus we obtain that $\varepsilon \leq \varepsilon' = d_F(P, Q)$. It follows from Theorem 1 that $d_F(P, P^\varepsilon) \leq \varepsilon \leq d_F(P, Q)$. ◀

3 Constructing smoothings in linear time

In this section we present a data structure for computing smoothings of a curve. The data structure relies on computing the *death times* of the vertices of P. We say that a vertex is *not present* in a smoothing P^ε if it has no corresponding vertex in P^ε. That is, during smoothing, it has become degenerate. We define the death time of a vertex p_i of P to be the smallest value $\varepsilon \geq 0$ for which p_i is not present in P^ε.

We proceed to give a precise expression for the death time of a vertex. To this end, define the *sublevel curve* of a vertex p_i of P to be the maximal subcurve of P that contains p_i and is bounded to the right by p_i. Define the *superlevel curve* of p_i analogously. These definitions mimic the notion of sublevel and superlevel sets of functions, but whereas for functions these sets can be disconnected, we require them to be subcurves of P. This makes sublevel and superlevel curves subsets of the respective sublevel and superlevel sets.

For a local maximum p_i of P, let P^- be its sublevel curve. We define the points ℓ_i and r_i as (global) minima on the prefix and suffix curves of P^- that end and start at p_i, respectively. We let $m_i := \min\{|p_i - \ell_i|, |p_i - r_i|\}$. See Figure 3 for an illustration. We analogously define the points ℓ_i

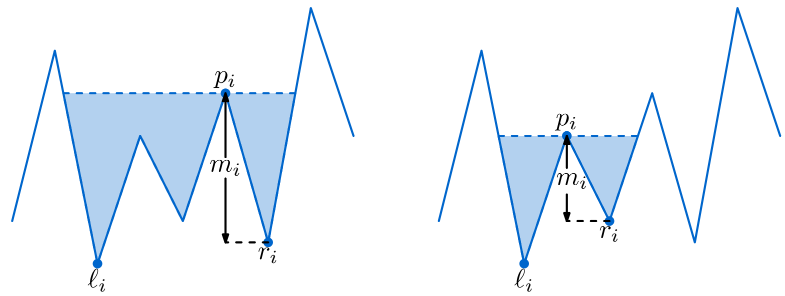

Figure 3 (left) The sublevel curve of p_i, below the dashed line segment. Points ℓ_i and r_i are the minima of the left and right parts of this sublevel curve. (right) If p_i is incident to a shortest edge of P then m_i is the length of this edge.

and r_i for a local maximum p_i of P, and again let $m_i := \min\{|p_i - \ell_i|, |p_i - r_i|\}$. In the following we show that the death time of an interior vertex p_i is equal to $m_i/2$.

▶ **Lemma 4.** *For any $2 \le i \le n-1$, the death time of vertex p_i is equal to $m_i/2$.*

Proof. Let p_i be a vertex of P for some $2 \le i \le n-1$ and assume without loss of generality that p_i is a local maximum. We distinguish between the case where p_i is incident to a shortest edge of P and the case where no incident edge is a shortest edge.

First assume that p_i is incident to a shortest edge e of P and assume without loss of generality that $e = \overline{p_{i-1}p_i}$. The death time of p_i is equal to $\|e\|/2$, since truncating e by half the minimum edge length truncates e into a point and p_i is not an endpoint of P. Observe that $m_i = \|e\|$. Indeed, because e is a shortest edge of P, we have that $p_{i-2} \ge p_i \ge p_{i-1}$ (if p_{i-2} exists) and $p_{i+1} \le p_{i-1}$. Thus we obtain that $\ell_i = p_{i-1}$ and $r_i \le \ell_i$, and hence $m_i = |p_i - p_{i-1}| = \|e\|$. See Figure 3. This proves that the death time of p_i is $m_i/2$ if p_i is incident to a shortest edge of P.

Next assume that p_i is not incident to a shortest edge of P. Let ε be equal to half the minimum edge length of P. Note that ℓ_i and r_i are both local minima of P and therefore vertices of P. As every local minimum of P gets increased by ε during the smoothing process, every local maximum gets decreased by ε, and the minimum edge length of P is 2ε, we obtain that the points $\ell_i^\varepsilon := \ell_i + \varepsilon$ and $r_i^\varepsilon := r_i + \varepsilon$ are the analogues of ℓ_i and r_i for the point p_i^ε, with respect to P^ε. It follows that m_i^ε, the analogue of m_i, is equal to $\min\{|p_i^\varepsilon - \ell_i^\varepsilon|, |p_i^\varepsilon - r_i^\varepsilon|\} = m_i - 2\varepsilon$. Applying the above recursively on the point p_i^ε, curve P^ε and value m_i^ε shows that the death time of p_i is $m_i/2$. ◀

With the expression m_i for the death times of vertices, we are able to compute the death time of every vertex in linear time. To this end we use *Cartesian trees*, introduced by Vuillemin [8]. A Cartesian tree is a type of binary max- or min-heap. We call a Cartesian tree a *max-Cartesian tree* if it represents a max-heap and a *min-Cartesian tree* if it represents a min-heap. A max-Cartesian tree T for a sequence of values $x_1, \ldots, x_n$ is recursively defined as follows. The root of T contains the maximum value x_j in the sequence. The subtree left of the root node is a max-Cartesian tree for the sequence $x_1, \ldots, x_{j-1}$, and the right subtree is a max-Cartesian tree for the sequence $x_{j+1}, \ldots, x_n$. See Figure 4. Max-Cartesian trees are defined symmetrically.

▶ **Lemma 5.** *We can compute the death time of every vertex of P in $O(n)$ time.*

Proof. To compute the death times of the vertices, we build two Cartesian trees; a max-Cartesian tree $T_{\max}$ and a min-Cartesian tree $T_{\min}$, both built on the sequence of vertices $p_1, \ldots p_n$ of P. These trees can be constructed in $O(n)$ time [5].

For a given node v of $T_{\max}$ storing vertex p_i, the vertices stored in the subtree rooted at v are precisely those of the sublevel curve of p_i. Thus if p_i is a local maximum, the values ℓ_i and r_i are precisely the minimum values stored in the left and right subtrees of v, respectively. We can therefore compute the death times of the local maxima of P with a bottom-up traversal of $T_{\max}$,

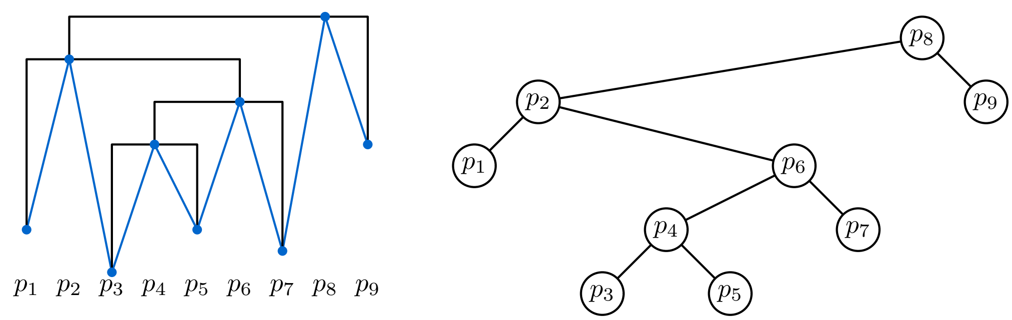

Figure 4 A max-Cartesian tree.

taking $O(n)$ time. Repeating the above process for $T_{\min}$, we compute the death times of the local minima of P in $O(n)$ time as well. ◀

Having computed the death times of the vertices, computing the smoothing P^ε of P is merely a matter of removing vertices of P with a death time at most ε, decreasing the leftover local maxima by ε, and increasing the leftover local minima by ε. To identify the vertices present in the smoothing, we store the vertices of P in another max-Cartesian tree, storing the vertices based on death time. The vertices with a death time greater than ε can be found in linear output-sensitive time by traversing the tree from the root. We obtain the following result.

▶ **Theorem 6.** *We can preprocess an n-curve P in $\mathbb{R}$ in $O(n)$ time, after which we can query it for the ε-smoothing of P in $O(k)$ time for any $\varepsilon \geq 0$, where k is the output complexity.*

▶ **Corollary 7.** *We can preprocess an n-curve P in $\mathbb{R}$ in $O(n)$ time, after which we can query it for a min-# ε-simplification of P in $O(k)$ time for any $\varepsilon \geq 0$, where k is the output complexity.*

Using death times, we can in linear time build a data structure that supports output-sensitive queries for closest k-curves as well.

▶ **Theorem 8.** *We can preprocess an n-curve P in $\mathbb{R}$ in $O(n)$ time, after which we can query it for a closest k-curve for P in $O(k)$ time for any $k \geq 1$.*

Proof. We store the death times of P in a max-heap in $O(n)$ time. To compute a closest k-curve we proceed as follows. Let ε be the $(k+1)$-st greatest death time. We can compute ε in $O(k)$ time using the algorithm for selection in binary heaps by Frederickson [4]. The ε-smoothing of P has at most k vertices and for any $\varepsilon' < \varepsilon$, any ε'-smoothing has more than k vertices. Thus by Corollary 3, P^ε is a closest k-curve for P. We report P^ε in $O(k)$ time using Theorem 6. ◀

References

1 Sergey Bereg, Minghui Jiang, Wencheng Wang, Boting Yang, and Binhai Zhu. Simplifying 3d polygonal chains under the discrete Fréchet distance. In *Proc. 8th Latin American Symposium on Theoretical Informatics (LATIN)*, volume 4957, pages 630–641, 2008. doi:10.1007/978-3-540-78773-0_54.

2 Erin Wolf Chambers, Elizabeth Munch, and Tim Ophelders. A family of metrics from the truncated smoothing of reeb graphs. In *Proc. 37th International Symposium on Computational Geometry (SoCG)*, volume 189, pages 22:1–22:17, 2021. doi:10.4230/LIPIcs.SoCG.2021.22.

3 Anne Driemel, Amer Krivosija, and Christian Sohler. Clustering time series under the Fréchet distance. In *Proc. 27th Annual Symposium on Discrete Algorithms (SODA)*, pages 766–785, 2016. doi:10.1137/1.9781611974331.ch55.

4 Greg N. Frederickson. An optimal algorithm for selection in a min-heap. *Information and Computation*, 104(2):197–214, 1993. doi:10.1006/inco.1993.1030.
5 Harold N. Gabow, Jon Louis Bentley, and Robert Endre Tarjan. Scaling and related techniques for geometry problems. In *Proc. 16th Annual ACM Symposium on Theory of Computing*, pages 135–143, 1984. doi:10.1145/800057.808675.
6 Mees van de Kerkhof, Irina Kostitsyna, Maarten Löffler, Majid Mirzanezhad, and Carola Wenk. Global curve simplification. In *Proc. 27th Annual European Symposium on Algorithms*, volume 144, pages 67:1–67:14, 2019. doi:10.4230/LIPIcs.ESA.2019.67.
7 Marc J. van Kreveld, Maarten Löffler, and Lionov Wiratma. On optimal polyline simplification using the Hausdorff and Fréchet distance. *Journal of Computational Geometry*, 11(1):1–25, 2020. doi:10.20382/jocg.v11i1a1.
8 Jean Vuillemin. A unifying look at data structures. *Communications of the ACM*, 23(4):229–239, 1980. doi:10.1145/358841.358852.